Efficient and Generic Transformations for Chosen-Ciphertext Secure Predicate Encryption

Marloes Venema(✉) and Leon Botros

Radboud University, Nijmegen
{m.venema,l.botros}@cs.ru.nl

Abstract. Predicate encryption (PE) is a type of public-key encryption that captures many useful primitives such as attribute-based encryption (ABE). Although much progress has been made to generically achieve security against chosen-plaintext attacks (CPA) efficiently, in practice, we also require security against chosen-ciphertext attacks (CCA). Because achieving CCA-security on a case-by-case basis is a complicated task, several generic conversion methods have been proposed. However, these conversion methods may incur a significant efficiency trade-off. Notably, for ciphertext-policy ABE, all generic conversion methods provide a significant overhead in the key generation, encryption or decryption algorithm. Additionally, many generic conversion techniques use one-time signatures to achieve authenticity, which are also known to significantly impact the efficiency.

In this work, we present a new approach to achieving CCA-security as generically and efficiently as possible, by splitting the CCA-conversion in two steps. The predicate of the scheme is first extended in a certain way, which is then used to achieve CCA-security generically e.g., by combining it with a hash function. To facilitate the first step efficiently, we also propose a novel predicate-extension transformation for a large class of pairing-based PE—covered by the pair and the predicate encodings frameworks—which incurs only a small constant overhead for all algorithms. In particular, this yields the most efficient generic CCA-conversion for ciphertext-policy ABE.

Keywords: predicate encryption · chosen-ciphertext security · generic transformation · identity-based encryption · attribute-based encryption

1 Introduction

Predicate encryption (PE) [36] is a paradigm that generalizes multiple powerful cryptographic primitives[1], such as identity-based encryption (IBE) [45,18] and attribute-based encryption (ABE) [44]. In contrast to traditional public-key encryption, PE allows for the fine-grained access control on data [33,13]. In PE, the

[1] Our definition of predicate encryption is in line with [4], which is more general than in some other works [39,10]. In those works, predicate encryption requires x to be hidden. We will use the notion of attribute hiding (e.g., in Appendix E) to refer to this additional property.

ciphertexts and secret keys are associated with "entities" x and y, respectively, for which a predicate P determines the ability of the secret key to decrypt the ciphertext. In particular, a secret key for y can decrypt a ciphertext for x if and only if $P(x, y) = 1$ (i.e., "the predicate is true"). For example, in ciphertext-policy ABE [13], x constitutes an access policy and y a set of attributes, and $P(x, y) = 1$ holds if the set of attributes satisfies the policy[2].

Over the years, many works have systematized and improved the techniques to achieve full[3] security of pairing-based PE against chosen-plaintext attacks (CPA) [48,8,26,9,3,4]. By formalizing the notions of pair encodings [8] and predicate encodings [48], a PE scheme can be abstracted to analyze only "what happens in the exponent". This simplifies the effort of proving security to information-theoretic and computational notions of security for vectors of polynomials. These frameworks are incredibly powerful: many PE schemes can be captured in them, and hence, these schemes are fully CPA-secure.

While these frameworks support CPA-security, in practice, it is often recommended or required that the scheme also provides security against chosen-ciphertext attacks (CCA) [41,46]. To this end, many works have proposed CCA-secure PE schemes, e.g., [18,37,30,38,47]. Moreover, to achieve CCA-security generically, any of the proposed transformations can be used, e.g.,

- using non-interactive zero-knowledge (NIZK) proofs of well-formedness [15];
- Fujisaki-Okamoto (FO) [29,35];
- Canetti-Halevi-Katz (CHK) [25];
- Boneh-Katz (BK) [19];
- Abe et al. (ACIK) [1];
- Yamada et al. (YA(SS)HK) [49,50], which consists of two transformations: one for delegatable ABE and one for verifiable ABE;
- Blömer-Liske (BL) [14];
- Koppula-Waters (KW) [39].

However, each of these generic transformations has a drawback. First, the transformation may be restricted to e.g., hierarchical IBE (HIBE) [25,19,1] or ABE [49,50]. Second, the FO-transform, the NIZK approach, and the transformations for verifiable schemes [49,50,14] incur an additional cost during decryption that is linear in the sizes of x and y, which is a significant cost for many ABE or inner-product encryption [36] schemes. Alternatively, these additional costs may be linear in the security parameter[4], such as in KW [39] and the transformations

[2] In this example, x could be called a predicate. However, in its dual variant, key-policy ABE, the keys are associated with policies and the ciphertexts with sets of attributes, and thus, the predicate is then associated with the keys, and not the ciphertexts. Similarly, dual-policy ABE [11] may specify policies for the keys and ciphertexts. Hence, we refer to both x and y as predicates (or attributes) throughout this work.

[3] As opposed to "selective" security, which restricts the security model in that the attacker needs to commit to the challenge x before the setup is run [24,25].

[4] Typically, the security parameter is fixed, e.g., equal to 128. Nevertheless, the additional costs are large.

Table 1: Comparison of the properties of the several CCA-transformations and our new transformations. For our CCA-transformations, we also consider alternative pathways based on the existing transformations to perform the two steps.

Variant	**Primitives used**	**Applicable to**	**Requirements**
FO [29,35]	hash	All PE	-
CHK [25]	OTS	(H)IBE	-
BK [19]	encapsulation, MAC, PRG	(H)IBE	-
ACIK [1, §7.2]	RPC	(H)IBE	partitioned KEM
YA(SS)HK [49,50]	OTS	ABE	delegatable or verifiable ABE
BL [14]	hash	All PE	verifiable pair encodings
KW [39]	PRG, OTS	All PE	-
Step 1 with CHK/BK	-	(H)IBE	-
Step 1 with YA(SS)HK	-	ABE	delegatable ABE
Step 2 with BK	encapsulation, MAC, PRG	All PE	-
Step 1 (new)	-	All PE	pair and predicate encodings
Step 2 (new)	RPC	All PE	decomposable ciphertexts

Note: PRG = pseudo-random generator, RPC = random-prefix collision-resistant hash

for delegatable CP-ABE [49]. Notably, for CP-ABE, no CCA-transformations yield a small and constant overhead.

In addition, most of these transformations—except for the NIZK, FO, BK and ACIK-transformations—use one-time signatures (OTS) to achieve authenticity of the ciphertexts. OTSs incur a considerable trade-off in storage and computational efficiency: either signing is efficient but the keys and signatures are large, or the keys and signatures are short but signing is inefficient. The BK-transformation improves on the CHK-transformation by replacing the OTS by a message authentication code (MAC) and a primitive called "encapsulation", which can be constructed from a hash and yields no such efficiency trade-off [19]. Encapsulation allows the encrypting user to commit to a secret value, which is later used to compute a MAC on the ciphertext to attain ciphertext authenticity. Subsequently, the ACIK-transformation improves on the BK-transformation by applying a primitive called a "random-prefix collision-resistant" hash directly to the ciphertext. The encrypting user is therefore not required to commit to a secret value (which also needs to be encrypted), and thus, minimizes the storage overhead.

1.1 Our contribution

In this work, we focus on generically achieving CCA-security for any PE as efficiently as possible. To this end, we propose a new high-level approach in the design of CCA-security transformations, by splitting any such transformations in two explicit steps. In the first step, the predicate of the scheme is extended. In the second step, the predicate-extended scheme is used to achieve CCA-security. Although several existing transformations take these steps implicitly, explicitly considering them as two steps may lead to more efficient (yet generic) constructions than previous methods allowed. To illustrate that, we propose two novel transformations that perform these two steps efficiently.

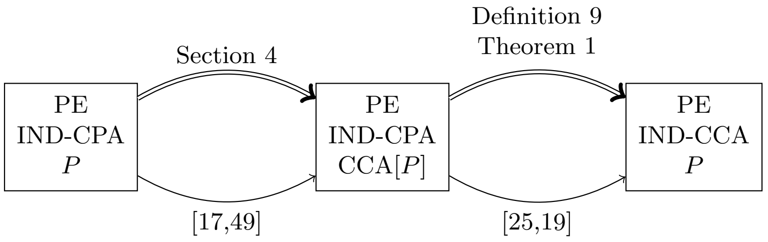
Warm-up: existing transformations. Apart from the NIZK, FO and KW transformations, all aforementioned generic transformations exploit the structure of the predicate to efficiently achieve CCA-security. Roughly, they all follow a similar approach: during encryption, the encrypting user commits to some value, which is then embedded in the predicate in addition to the original predicate. For example, for (H)IBE [25,19,1], this value is embedded in the (additional "layer" of the) identity, and for delegatable CP-ABE, the bit-representation of the value is encoded as an AND-policy (and is taken in conjunction with the original policy). Because these transformations exploit the specific structures of the predicates, they are therefore only applicable to those predicates. Furthermore, depending on the technique, the value to which is commited is either generated independently of the ciphertext [25,19,49] or by applying a hash with a specific property to the ciphertext [1,14]. Although the latter requires that the ciphertext is of a specific structure (which many pairing-based schemes satisfy), it relies on fewer primitives and yields less storage overhead in the ciphertext.

Our transformations. On a high level, our approach consists of two steps with varying "levels of genericness" (of which an overview is shown in Figure 1). First, we transform a CPA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CPA},P}$ for predicate P into a CPA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CPA},P'}$ for extended predicate P'. For this step, we propose novel generic constructions in the pair and predicate encodings frameworks. (We also show that our predicate-encoding transformation preserves the attribute-hiding property in [26].) As a result, many pairing-based PE schemes can be transformed using this construction. Second, we transform any CPA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CPA},P'}$ for extended predicate P' into a CCA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CCA},P}$ for the original predicate P. This step can be done by using similar approaches as CHK and BK. We also give a new transformation based on the ACIK-approach. This new transformation applies to any PE scheme for which the ciphertexts are "decomposable" (which is a similar notion to that of partitioned in [1]).

Although our transformation is less generic than fully generic transformations such as FO and KW, ours is more generic than most of the other transformations (Table 1). In fact, our approach can be seen as an efficient generalization of the transformations that exploit the specific structures of the predicate, i.e., CHK, BK, ACIK, and YA(SS)HK. However, by performing the transformation in two steps, we also allow for more efficient (yet generic) constructions. We show that this is especially beneficial for CP-ABE, for which existing such transformations always induce a linear computational overhead in at least one of the algorithms.

Step one: securely extending the predicate. We first extend the predicate P to some predicate $P' = \text{CCA}[P]$. The idea behind this is similar to the approach for hierarchical IBE [25,19,17] and delegatable KP-ABE by Yamada et

Fig. 1: A high-level overview of the transformations and our associated definitions and theorems that prove security of the given transformations. The double-edged arrows indicate that we give a novel provably secure generic transformation in this work, while the normal-edged arrows provide transformations that have been given in other works.



al. (YAHK) [49], and is later also applied using wildcards by Tomida et al. [47]. Roughly, the secret key predicate y is extended to $y \wedge y'$, where y' is either an attribute or a wildcard $*$, and the ciphertext predicate x is extended to (x, x'), where x' is an attribute. The predicate is satisfied if $P(x, y)$ and either $y' = *$ or $y' = x'$ holds. We provide a new transformation in the pair and predicate encodings framework that extends the predicate in this way. The computational overhead incurred by our transformation is a low constant, and unlike YAHK, we do not require the PE scheme to be a delegatable KP-ABE for the transformation to work. Because we generically transform *any* PE into a scheme with this specific extended predicate, we can also efficiently support CCA-security in e.g., CP-ABE. Roughly, we take an AND-composition over the original PE and an “all-or-one-identity” IBE, by using the ideas from Ambrona et al. [7] and Attrapadung [10]. For the “all-or-one-identity” IBE, we use the first scheme of Kiltz and Vahlis [38] as inspiration, which is essentially implied by a composition of the Boneh-Boyen (BB) IBE [16] with a wildcard variant of the same scheme.

Step two: achieving CCA-security. We first consider on a high level what the CCA-transformation looks like. Let $\Gamma = (\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Encaps}, \mathrm{Decaps})$ be a predicate key-encapsulation scheme (possibly derived from a PE) for the extended predicate, such that that ciphertext is of the form

$$\mathrm{Encaps}(\mathrm{MPK}, (x, x')) = (\mathrm{K}, \mathrm{CT}_1, \mathrm{CT}_{2,(x,x')}),$$

where MPK is the master public key generated in the Setup, K is the encapsulated key to be used to symmetrically encrypt, CT_1 is some randomized part of the ciphertext that is independent of extended predicate (x, x'), and $\mathrm{CT}_{2,(x,x')}$ denotes the rest of the ciphertext. Following the approach by Kiltz and Vahlis [38] and Abe et al. (ACIK) [1], we first split the key-encapsulation algorithm in two parts, and then introduce an authenticated encryption scheme $\mathrm{SE} = (\mathrm{Enc}_{\mathrm{K}}, \mathrm{Dec}_{\mathrm{K}})$ and a random-prefix collision-resistant hash function RPC (which takes as input a random prefix k and another input to be hashed), i.e.,

$$\mathrm{Encaps}(\mathrm{MPK}, (x, x')) = (\underbrace{\mathrm{K}, \mathrm{CT}_1}_{\mathrm{Encaps}_1}, \underbrace{\mathrm{CT}_{2,(x,x')}}_{\mathrm{Encaps}_2}).$$

Then, we obtain the CCA-transformed encryption as follows

$$\mathrm{Encrypt}'(\mathrm{MPK}, \boxed{x}, M) = (\boxed{\mathrm{CT}_{\mathrm{sym}} = \mathrm{Enc}_{\mathrm{K}}(M \| \mathrm{CT}_{2,(x,x')})}, \mathrm{CT}_1, \mathrm{CT}_{2,(x,x')}, k),$$

where $(\mathrm{K}, \mathrm{CT}_1) \leftarrow \mathrm{Encaps}_1(\mathrm{MPK})$, $k \in_R \{0,1\}^\lambda$, $x' \leftarrow \mathrm{RPC}(k, \mathrm{CT}_1)$, and then $\mathrm{CT}_{2,(x,x')} \leftarrow \mathrm{Encaps}_2(\mathrm{MPK}, (x, x'))$.

Proving CCA-security. We prove CCA-security of the proposed generic construction similarly as other transformations [25,19,38,1,49]. Specifically, the decryption queries are answered as follows. Suppose that $\mathrm{CT}_x = (\mathrm{CT}_{\mathrm{sym}}, \mathrm{CT}_1, \mathrm{CT}_{2,(x,x')})$ is some ciphertext and y is some predicate such that $P(x, y) = 1$, queried by the attacker. Then, the challenger can generate a secret key for $(y, y' = x')$, and decrypt the ciphertext. The challenger rejects a decryption query if it is similar to the challenge ciphertext $\mathrm{CT}^*_{x^*} = (\mathrm{CT}^*_{\mathrm{sym}}, \mathrm{CT}^*_1, \mathrm{CT}_{2,(x^*,x'^*)})$, i.e., if $\mathrm{CT}_1 \neq \mathrm{CT}^*_1$ and $x' = x'^*$, or if $\mathrm{K} = \mathrm{K}^*$ and $\mathrm{CT}_{\mathrm{sym}} \neq \mathrm{CT}^*_{\mathrm{sym}}$ or $\mathrm{CT}_{2,(x,x')} \neq \mathrm{CT}^*_{2,(x^*,x'^*)}$. Intuitively, the probability that a valid ciphertext is rejected—i.e., the probability that a valid ciphertext satisfies any of these conditions—is negligible due to the random-prefix collision resistance of the hash RPC and the authenticity of the symmetric encryption scheme $\mathrm{SE} = (\mathrm{Enc}_{\mathrm{K}}, \mathrm{Dec}_{\mathrm{K}})$.

Alternative pathways. As mentioned, we focus on achieving CCA-security as efficiently and as generically as possible. Although the two proposed transformations for the two steps are applicable to large classes of existing PE schemes, they do not apply to *all* PE schemes. For example, post-quantum schemes [5,2,22,32] are not covered by our predicate-extension transformation, and not all schemes may be decomposable and therefore qualify for our second-step transformation. To make our second step more generic, one could also use the BK-approach [19], which does not require the extended-predicate scheme to have ciphertexts with a certain structure[5]. However, it does provide more storage overhead and relies on more primitives (i.e., two independent hash functions, a MAC and a PRG). The latter may be undesirable in practice, e.g., because no suitable implementations are available of all primitives. In this regard, our second-step transformation could provide an effective solution, as it requires only one hash function. Importantly, because the second step can be done entirely generically, the effort of achieving CCA-security is reduced to finding an efficient predicate extension. Note that several such predicate-extension techniques have been described implicitly, e.g., the CHK- and YAHK-approaches extend the (H)IBE with another

[5] The security proof of the generalized variant of the BK-transformation is analogous to that of the BK-transformation itself.

level in the hierarchy and extend the ABE ciphertext predicate with a conjunction or disjunction, respectively. Furthermore, for schemes for which there is no such predicate-extension technique available (that is sufficiently efficient), we only need to devise an efficient predicate-extension transformation, instead of performing a full-fledged CCA-security conversion.

1.2 Performance analysis and comparison

We compare the efficiency of our CCA-transformation with the others. From a theoretical standpoint, ours is the most efficient. It incurs only a small constant overhead in all algorithms and the key and ciphertext sizes in the first step, regardless of the size of the predicate. For all other transformations, this is not the case. Especially for schemes with linear-sized predicates, such as ABE, this provides a significant efficiency improvement. In contrast, the other approaches applicable to ABE incur the following efficiency trade-offs:

- FO [29,35]: in general, this approach incurs little to no overhead to most algorithms, except for the decryption algorithm, which requires an invocation of the encryption algorithm, whose costs are often linear;
- YAKK-del [49]: depending on the type of ABE, this transformation for delegatable schemes might either be very efficient or very costly. For KP-ABE, the transformation incurs only a small constant overhead in all algorithms and the key and ciphertext sizes. For CP-ABE, the transformation incurs an additional overhead that is linear in the security parameter in the encryption and decryption algorithms;
- YAHK-ver [49], BL [14]: these transformations for verifiable schemes incur little to no overhead in most of the algorithms and the key and ciphertext sizes, except for the decryption algorithm, which also verifies whether the ciphertexts are well-formed. The costs incurred by the verification step are similar to the decryption costs of the CPA-secure PE scheme, and therefore roughly double the decryption costs of the CCA-secure PE scheme (which are often linear in the predicate size);
- KW [39]: this fully-generic transformation is very costly and incurs an overhead in all algorithms and sizes that is linear in the security parameter.

In Section 5, we analyze the performance of two schemes to show the advantage of our transformation compared to existing transformations. In particular, we analyze the costs of CGW-IBE [26], an anonymous IBE scheme that is fully secure in the predicate encodings framework. For this scheme, only the FO-transformation readily yields CCA-security[6]. Our implementations show that our CCA-secure variant outperforms the FO-variant in the decryption. We also analyze the costs of the CCA-secure variants of the fully secure version of RW13 [43] in the pair encodings framework, i.e., RWAC [4,10], which is a large-universe

[6] Possibly, by modifying the scheme, one could obtain an attribute-hiding hierarchical IBE with two levels, which can then be used to obtain a CCA-secure IBE for one level with CHK, BK and ACIK, although we did not do that in this work.

CP-ABE scheme. We compare our CCA-variant with those that follow from applying FO and the transformations for delegatable and verifiable CP-ABE. Our analysis shows that our transformation has a much faster decryption than all existing transformations, while incurring a marginal overhead in the other algorithms compared to the fastest variants.

1.3 Organization

This paper is structured as follows. We first provide some notations and definitions in Section 2. Then, in Section 3, we give the generic transformations from any CPA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CPA},P'}$ for extended predicate P' into a CCA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CCA},P}$ for original predicate P, i.e., step 2. After this, in Section 4, we propose novel generic constructions for transforming any CPA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CPA},P}$ for predicate P into a CPA-secure PE $\mathit{\Gamma}_{\text{PE,IND-CPA},P'}$ for extended predicate P', i.e., step 1. We first give the more general steps of the transformation and then the less generic step, both due to the "level of genericness" and the more complicated nature of the security proofs in the pair and predicate encodings frameworks. Finally, we compare the performance of our transformation in Section 5, and conclude the paper in Sections 6 and 7 by discussing future directions.

2 Preliminaries

2.1 Notation

We use λ to denote the security parameter. We denote a negligible function parametrized by λ by $\text{negl}(\lambda)$. If an element is chosen uniformly at random from a finite set S, then we denote this as $x \in_R S$. For integers $a < b$, we denote $[a,b] = \{a, a+1, ..., b-1, b\}$, $[b] = [1,b]$ and $\overline{[b]} = [0,b]$. We use boldfaced variables $\mathbf{A}$ and $\mathbf{v}$ for matrices and vectors, respectively. We use $a \| b$ to indicate that two strings a and b are concatenated.

2.2 Pairings (or bilinear maps)

We define a pairing to be an efficiently computable map e on three groups $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ of prime order p, so that $e \colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, with generators $g \in \mathbb{G}, h \in \mathbb{H}$ is such that for all $a, b \in \mathbb{Z}_p$, it holds that $e(g^a, h^b) = e(g,h)^{ab}$ (bilinearity), and for $g^a \neq 1_{\mathbb{G}}, h^b \neq 1_{\mathbb{H}}$, it holds that $e(g^a, h^b) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}'}$ denotes the unique identity element of the associated group $\mathbb{G}'$ (non-degeneracy). We refer to $\mathbb{G}$ and $\mathbb{H}$ as the two source groups, and $\mathbb{G}_T$ as the target group.

2.3 Predicate encryption

Predicate family. A predicate family [8] is a set $P = \{P_\kappa\}_{\kappa \in \mathbb{N}^c}$ for some constant c, where $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$. For κ, it holds that $\kappa = (p, \text{par})$, where p is a natural number and par denote the rest of the entries.

Definition 1 (Predicate encryption (PE) [4]). *A predicate encryption scheme for a predicate family $P = \{P_\kappa\}_{\kappa \in \mathbb{N}^c}$ over a message space $\mathcal{M} = \{M_\lambda\}_{\lambda \in \mathbb{N}}$ consists of four algorithms:*

- $\mathrm{Setup}(\lambda, \mathrm{par})$*: On input the security parameter λ and parameters* par*, this probabilistic algorithm generates the domain parameters, the master public key* MPK *and the master secret key* MSK*. In addition, κ is set to $\kappa = (p, \mathrm{par})$, where p denotes a natural number.*
- $\mathrm{KeyGen}(\mathrm{MSK}, y)$*: On input the master secret key* MSK *and some $y \in \mathcal{Y}_\kappa$, this probabilistic algorithm generates a secret key SK_y.*
- $\mathrm{Encrypt}(\mathrm{MPK}, x, M)$*: On input the master public key* MPK*, some $x \in \mathcal{X}_\kappa$ and message M, this probabilistic algorithm generates a ciphertext CT_x.*
- $\mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x)$*: On input the master public key* MPK*, the secret key SK_y, and the ciphertext CT_x, if $P_\kappa(x, y) = 1$, then it returns M. Otherwise, it returns an error message $\perp$.*

Correctness. For all par, $M \in \mathcal{M}_\lambda$, $x \in \mathcal{X}_\kappa$, and $y \in \mathcal{Y}_\kappa$ such that $P_\kappa(x, y) = 1$,

$$\Pr[(\mathrm{MPK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(1^\lambda);$$
$$\mathrm{Decrypt}(\mathrm{MPK}, \mathrm{KeyGen}(\mathrm{MSK}, y), \mathrm{Encrypt}(\mathrm{MPK}, x, M)) \neq M] \leq \mathrm{negl}(\lambda).$$

Key-encapsulation mechanism (KEM). In the key-encapsulation variant (Appendix A), which we call predicate KEM (P-KEM), we replace Encrypt by Encaps and Decrypt by Decaps, where Encaps also outputs a symmetric key, and Decaps outputs a symmetric key instead of a plaintext message.

2.4 Full security against chosen-plaintext attacks

Definition 2 (Full security against chosen-plaintext attacks (CPA) [4]). *We define the security game* $\mathrm{IND\text{-}CPA}(\lambda, \mathrm{par})$ *between challenger and attacker as follows:*

- ***Setup phase:*** *The challenger runs* $\mathrm{Setup}(\lambda)$ *to obtain* MPK *and* MSK*, and sends the master public key* MPK *to the attacker.*
- ***First query phase:*** *The attacker queries secret keys for $y \in \mathcal{Y}_\kappa$, and obtains* $\mathrm{SK}_y \leftarrow \mathrm{KeyGen}(\mathrm{MSK}, y)$ *in response.*
- ***Challenge phase:*** *The attacker specifies some $x^* \in \mathcal{X}_\kappa$ such that for all y in the first key query phase, we have $P_\kappa(x^*, y) = 0$, and generates two messages M_0 and M_1 of equal length in $\mathcal{M}_\lambda$, and sends these to the challenger. The challenger flips a coin, i.e., $\beta \in_R \{0, 1\}$, encrypts M_β under x^*, i.e.,* $\mathrm{CT}_{x^*} \leftarrow \mathrm{Encrypt}(\mathrm{MPK}, x^*, M_\beta)$*, and sends the resulting ciphertext* CT_{x^*} *to the attacker.*
- ***Second query phase:*** *This phase is identical to the first query phase, with the additional restriction that the attacker can only query $y \in \mathcal{Y}_\kappa$ such that $P_\kappa(x^*, y) = 0$.*

- ***Decision phase:*** *The attacker outputs a guess β' for β.*

The advantage of the attacker is defined as $\mathsf{Adv}_{\mathrm{PE,IND\text{-}CPA}} = |\Pr[\beta' = \beta] - \frac{1}{2}|$. *A scheme is fully secure if all polynomial-time attackers have at most a negligible advantage in this security game, i.e.,* $\mathsf{Adv}_{\mathrm{PE,IND\text{-}CPA}} \leq \mathrm{negl}(\lambda)$.

In the selective security model, the attacker commits to the predicate $x^* \in \mathcal{X}_\kappa$ before the Setup phase.

2.5 Full security against chosen-ciphertext attacks

Definition 3 (Full security against chosen-ciphertext attacks (CCA)). *We define the security game* $\mathrm{IND\text{-}CCA}(\lambda, \mathrm{par})$ *between challenger and attacker as follows:*

- ***Setup phase:*** *The challenger runs* $\mathrm{Setup}(\lambda)$ *to obtain* MPK *and* MSK*, and sends the master public key* MPK *to the attacker.*
- ***First query phase:*** *The attacker can make two types of queries:*
  - ***Key query:*** *the attacker queries secret keys for $y \in \mathcal{Y}_\kappa$, and obtains* $\mathrm{SK}_y \leftarrow \mathrm{KeyGen}(\mathrm{MSK}, y)$ *in response.*
  - ***Decryption query:*** *the attacker sends a ciphertext* CT_x *for $x \in \mathcal{X}_\kappa$ and $y \in \mathcal{Y}_\kappa$, with $P_\kappa(x, y) = 1$, to the challenger, who returns the message* $M \leftarrow \mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x)$*, where* $\mathrm{SK}_y \leftarrow \mathrm{KeyGen}(\mathrm{MSK}, y)$.
- ***Challenge phase:*** *The attacker specifies some $x^* \in \mathcal{X}_\kappa$ such that for all y in the first key query phase, we have $P_\kappa(x^*, y) = 0$, and generates two messages M_0 and M_1 of equal length in $\mathcal{M}_\lambda$, and sends these to the challenger. The challenger flips a coin, i.e., $\beta \in_R \{0, 1\}$, encrypts M_β under x^*, i.e.,* $\mathrm{CT}^*_{x^*} \leftarrow \mathrm{Encrypt}(\mathrm{MPK}, x^*, M_\beta)$*, and sends the resulting ciphertext* $\mathrm{CT}^*_{x^*}$ *to the attacker.*
- ***Second query phase:*** *This phase is identical to the first query phase, with the additional restriction that the attacker can only query keys for $y \in \mathcal{Y}_\kappa$ such that $P_\kappa(x^*, y) = 0$, and it cannot make a decryption query for* CT^*_{x*}.
- ***Decision phase:*** *The attacker outputs a guess β' for β.*

The advantage of the attacker is defined as $\mathsf{Adv}_{\mathrm{PE,IND\text{-}CCA}} = |\Pr[\beta' = \beta] - \frac{1}{2}|$. *A scheme is fully secure if all polynomial-time attackers have at most a negligible advantage in this security game, i.e.,* $\mathsf{Adv}_{\mathrm{PE,IND\text{-}CCA}} \leq \mathrm{negl}(\lambda)$.

2.6 Authenticated symmetric encryption

Definition 4 (Symmetric encryption). *Let λ be the security parameter. A symmetric encryption scheme* $\mathrm{SE} = (\mathrm{Enc}_\mathrm{K}, \mathrm{Dec}_\mathrm{K})$*, with symmetric key* $\mathrm{K} \in \mathcal{K}(\lambda)$*, where $\mathcal{K}(\lambda)$ is some key space of size λ, is defined by*

- $\mathrm{Enc}_\mathrm{K}(M)$*: On input message $M \in \{0, 1\}^*$, encryption returns a ciphertext* CT_sym.
- $\mathrm{Dec}_\mathrm{K}(\mathrm{CT}_\mathrm{sym})$*: On input ciphertext* CT_sym*, decryption returns a message M or an error message $\perp$.*

The scheme is correct if for all keys $\mathrm{K} \in \mathcal{K}(\lambda)$ *and all messages* $M \in \{0,1\}^*$*, we have* $\mathrm{Dec}_{\mathrm{K}}(\mathrm{Enc}_{\mathrm{K}}(M)) = M$*.*

For symmetric encryption, we use the same security notions as in [38], i.e., ciphertext indistinguishability and ciphertext authenticity.

Definition 5 (Ciphertext indistinguishability of symmetric encryption). *Let* λ *be a security parameter and let* $\mathrm{SE} = (\mathrm{Enc}_{\mathrm{K}}, \mathrm{Dec}_{\mathrm{K}})$ *be an (authenticated) symmetric encryption scheme. Consider the following game between challenger* $\mathcal{C}$ *and attacker* $\mathcal{A}$*. The challenger first picks a key* $\mathrm{K} \in \mathcal{K}(\lambda)$*. Then, the attacker specifies two messages* M_0, M_1 *and gives these to the challenger, who flips a coin* $\beta \in_R \{0,1\}$ *and returns* $\mathrm{CT}_{\mathrm{sym}} \leftarrow \mathrm{Enc}_{\mathrm{K}}(M_\beta)$ *to the attacker. The attacker* $\mathcal{A}$ *outputs a guess* β' *for* β*. Then,* $\mathrm{SE} = (\mathrm{Enc}_{\mathrm{K}}, \mathrm{Dec}_{\mathrm{K}})$ *has indistinguishable ciphertexts if for all polynomial-time attackers* $\mathcal{A}$ *in the game above holds:*

$$\mathsf{Adv}_{\mathrm{SE,CIND}} = \left| \Pr[\beta' = \beta] - \frac{1}{2} \right| \leq \mathrm{negl}(\lambda).$$

In this work, we will often assume that $\mathcal{K}(\lambda)$ is the target group $\mathbb{G}_T$. Because most encryption schemes take a key that is a bit string of λ or 2λ bits as input, we use a secure key derivation function $\mathrm{KDF}\colon \mathcal{K}(\lambda) \rightarrow \{0,1\}^\lambda$ (or $\{0,1\}^{2\lambda}$) to map the target group elements to strings [28].

Definition 6 (Ciphertext authenticity of authenticated encryption). *Let* λ *be a security parameter and let* $\mathrm{SE} = (\mathrm{Enc}_{\mathrm{K}}, \mathrm{Dec}_{\mathrm{K}})$ *be an (authenticated) symmetric encryption scheme. Consider the following game between challenger* $\mathcal{C}$ *and attacker* $\mathcal{A}$*. The challenger first picks a key* $\mathrm{K} \in \mathcal{K}(\lambda)$*. Then, the attacker specifies one message* M *and gives it to the challenger, who returns* $\mathrm{CT}_{\mathrm{sym}} \leftarrow \mathrm{Enc}_{\mathrm{K}}(M)$ *to the attacker. The attacker outputs a ciphertext* $\mathrm{CT}'_{\mathrm{sym}}$*. Then, the encryption scheme has ciphertext authenticity if for all such attackers holds that* $\mathsf{Adv}_{\mathrm{SE,CAUT}} = \Pr[\mathrm{Dec}_{\mathrm{K}}(\mathrm{CT}'_{\mathrm{sym}}) \neq \perp \wedge \mathrm{CT}'_{\mathrm{sym}} \neq \mathrm{CT}_{\mathrm{sym}}] \leq \mathrm{negl}(\lambda)$*.*

We define a random-prefix collision-resistant hash function (RPC) as follows.

Definition 7 (Random-prefix collision-resistant hash function (RPC) [1]). *Let* λ *be a security parameter, and let* $\mathrm{RPC}\colon \{0,1\}^\lambda \times \mathcal{G} \rightarrow \mathcal{Z}$ *be a hash function that takes two inputs, one in* $\{0,1\}^\lambda$ *and one in* $\mathcal{G}$*, and maps it to an element in* $\mathcal{Z}$*. Consider the following game between challenger* $\mathcal{C}$ *and attacker* $\mathcal{A}$*. The attacker gives the challenger some* $g \in \mathcal{G}$*. The challenger then picks* $k \in \{0,1\}^\lambda$*, and gives* k *and* $\mathrm{RPC}(k, g)$ *to the attacker. Then, the* RPC *is random-prefix collision resistant if for all such attackers, it holds that the advantage* $\mathsf{Adv}_{\mathrm{RPC}} = \Pr[(k', g') \in \{0,1\}^\lambda \times \mathcal{G} \wedge (k', g') \neq (k, g) \wedge \mathrm{RPC}(k', g') = \mathrm{RPC}(k, g)] \leq \mathrm{negl}(\lambda)$*.*

In this work, we use the concrete instantiation given by Abe et al. [1]. In particular, their instantiation of the RPC hash is a second pre-image resistant hash that takes as input a 128-bit string k and the element in $\mathcal{G}$.

3 Our generic CCA-transformation

We introduce our generic transformation for CCA-secure PE.

3.1 Step one: extending the predicate

Let $\Gamma_{\mathrm{PE,IND\text{-}CPA},P} = (\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Encrypt}, \mathrm{Decrypt})$ be a predicate encryption scheme for the predicate family $P = \{P_\kappa\}_\kappa$ with $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$. In the first step of our approach, we transform it into a scheme $\Gamma_{\mathrm{PE,IND\text{-}CPA},P'}$ for predicate $P' = \mathrm{CCA}[P]$, where $\mathrm{CCA}[P]$ denotes the predicate extension required by the CCA-transformation on predicate P, i.e., $P'_\kappa \colon \mathcal{X}'_\kappa \times \mathcal{Y}'_\kappa \to \{0,1\}$, where

- $\mathcal{X}'_\kappa = (\mathcal{X}_\kappa, \mathcal{Z})$ and $\mathcal{Y}'_\kappa = (\mathcal{Y}_\kappa, \mathcal{Z} \cup \{*\})$, where $|\mathcal{Z}| \geq 2^{2\lambda}$;
- $P'_\kappa((x, x'), (y, y')) = 1$ if and only if
  - $P_\kappa(x, y) = 1$ and $y' = *$;
  - or $P_\kappa(x, y) = 1$ and $x' = y'$.

In Section 4, we give several predicate-extension transformations that generically transform a CPA-secure PE scheme for the predicate P in a CPA-secure PE scheme for predicate $\mathrm{CCA}[P]$. Conceptually, we do this by making an AND-composition of the original PE scheme for predicate P with an "all-or-one-identity" IBE. In an "all-or-one-identity" IBE, a user is given either a key for one particular identity $y' \in \mathcal{Z}$ or all identities $y' = *$. These transformations are not fully generic, because they only apply to pairing-based ABE. In particular, they are given in the pair encodings [8,4] and predicate encodings [48,26] frameworks, since it is relatively simple to generically prove security of such transformations [7], and many PE schemes can be instantiated in these frameworks [4,10,6].

We note that a scheme with an extended predicate can also be obtained in other ways. For instance, the approaches used for (H)IBE [25,19,17] also apply. Additionally, the generic transformations using delegation by Yamada et al. [49] yield suitable candidates as well, but only for KP-ABE and CP-ABE. Furthermore, the transformation by Tomida et al. [47] using delegation is similar to our proposed constructions in Section 4, but these only work for their specific KP-ABE and CP-ABE schemes, and are not generic in the sense that they can be applied to any PE. Additionally, while our transformations in Section 4 are specific to pairing-based PE, they may also work for PE based on other cryptographic assumptions, for instance, by creating an "all-or-one-identity" IBE from a suitable IBE from post-quantum assumptions [31], and taking an AND-composition with any post-quantum PE [5,2,22,32].

3.2 Step two: generic CCA-secure construction

Much like in [38] and [1], the predicate extension is generated from part of the ciphertext. To this end, we introduce the notion of "decomposable extended-predicate encryption (EPE)", which we use as input to the CCA-security transformation. In decomposable EPE, we decompose the ciphertext in three parts, such that one of the parts is used to generate the predicate extension with the hash.

Definition 8 (Decomposable EPE). *An EPE scheme with encryption algorithm* Encrypt *is called decomposable if the ciphertexts are decomposable. The ciphertexts* $\mathrm{CT}_{(x,x')} \leftarrow \mathrm{Encrypt}(\mathrm{MPK}, (x, x'), M)$ *are decomposable if they can be decomposed:*

$$\mathrm{CT}_{(x,x')} = (\mathrm{CT}_{\mathrm{M}}, \mathrm{CT}_1, \mathrm{CT}_{2,(x,x')}), \text{ such that}$$

- *only* $\mathrm{CT}_{2,(x,x')}$ *depends on* (x, x')*;*
- *only* CT_{M} *contains the message;*
- CT_{M} *is uniquely determined by* M*,* MPK *and* CT_1*, and conversely,* CT_1 *is uniquely determined by* M*,* MPK *and* CT_{M}*;*
- $\mathrm{CT}_1 \in \mathcal{G}$ *is generated independently of* $\mathrm{CT}_{2,(x,x')}$*;*
- *for any* $(\hat{x}, \hat{x}') \in \mathcal{X}'_\kappa$ *with* $\hat{x}' \neq x'$*, we have that any* $\mathrm{CT}_{2,(\hat{x},\hat{x}')}$ *that is valid for* CT_1 *is such that* $\mathrm{CT}_{2,(\hat{x},\hat{x}')} \neq \mathrm{CT}_{2,(x,x')}$*;*
- CT_1 *is generated uniformly at random over* $\mathcal{G}$*, such that* $\Pr[\mathrm{CT}_1 = \mathrm{CT}'_1 \mid \mathrm{CT}'_1 \in_R \mathcal{G}] \leq \mathrm{negl}(\lambda)$.

In this case, we also define two algorithms for encryption, i.e.,

- $\mathrm{Encrypt}_1(\mathrm{MPK}, M) \rightarrow (\mathrm{CT}_{\mathrm{M}}, \mathrm{CT}_1)$*;*
- $\mathrm{Encrypt}_2(\mathrm{MPK}, (x, x')) \rightarrow \mathrm{CT}_{2,(x,x')}$*,*

such that

$$\mathrm{Encrypt}(\mathrm{MPK}, (x, x'), M) = (\mathrm{Encrypt}_1(\mathrm{MPK}, M), \mathrm{Encrypt}_2(\mathrm{MPK}, (x, x'))).$$

Decomposable EP-KEM. This definition naturally extends to the key-encapsulation variants of EPE, i.e., by replacing CT_{M} by the encapsulated symmetric key K. In this case, K is required to be uniquely determined by MPK and CT_1. We can generically obtain a EP-KEM from an EPE by encrypting a randomly-generated symmetric key K. For PE schemes with a certain algebraic structure, we can also generically obtain a more efficient KEM (Appendix B).

Generic construction. We use a CPA-secure decomposable EP-KEM with an extended predicate to generically construct a CCA-secure hybrid PE for the original predicate.

Definition 9 (Generic CCA-secure construction). *Let* $\mathit{\Gamma}_{\mathrm{PE}} = (\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Encaps}, \mathrm{Decaps})$ *be a predicate key-encapsulation mechanism for the predicate family* $P = \{P_\kappa\}_\kappa$ *with* $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \rightarrow \{0, 1\}$*, and suppose* $\mathit{\Gamma}_{\text{EP-KEM}} = (\mathrm{Setup}_{\text{EP-KEM}}, \mathrm{KeyGen}_{\text{EP-KEM}}, \mathrm{Encaps}_{\text{EP-KEM}}, \mathrm{Decaps}_{\text{EP-KEM}})$ *is a decomposable extended-predicate KEM for predicate* $P' = \mathrm{CCA}[P]$ *(e.g., obtained with a predicate-extension transformation (Section 4)). Let* $\mathrm{SE} = (\mathrm{Enc}_{\mathrm{K}}, \mathrm{Dec}_{\mathrm{K}})$ *be an authenticated symmetric encryption scheme with key space* $\mathcal{K}_\lambda$ *equal to the space in which* CT_{M} *lives, and* $\mathrm{RPC} \colon \{0, 1\}^\lambda \times \mathcal{G} \rightarrow \mathcal{Z}$ *be a random-prefix collision-resistant hash function. Then, we define* $\mathit{\Gamma}'_{\mathrm{PE}} = (\mathrm{Setup}', \mathrm{KeyGen}', \mathrm{Encrypt}', \mathrm{Decrypt}')$ *to be the CCA-secure hybrid encryption version of scheme* $\mathit{\Gamma}_{\mathrm{PE}}$ *for predicate* P *as*

- $\text{Setup}'_{\text{PE}}(\lambda, \text{par})$*: On input* λ *and* par*, the setup generates* $(\text{MPK}, \text{MSK}) \leftarrow \text{Setup}_{\text{EP-KEM}}(\lambda, \text{par})$*, and sets* $\text{MPK}' = \text{MPK}$ *and* $\text{MSK}' = \text{MSK}$*.*
- $\text{KeyGen}'_{\text{PE}}(\text{MSK}', y)$*: On input the master secret key* MSK' *and some* $y \in \mathcal{Y}_\kappa$*, it returns* $\text{SK}'_y \leftarrow \text{KeyGen}_{\text{EP-KEM}}(\text{MSK}, (y, *))$*.*
- $\text{Encrypt}'_{\text{PE}}(\text{MPK}', x, M)$*: On input the master public key* MPK'*,* $x \in \mathcal{X}_\kappa$ *and message* $M \in \{0,1\}^*$*, the encrypting user computes* $(\text{K}, \text{CT}_1) \leftarrow \text{Encaps}_{1,\text{EP-KEM}}(\text{MPK})$*, picks* $k \in_R \{0,1\}^\lambda$ *and sets* $x' = \text{RPC}(k, \text{CT}_1)$*, then generates* $\text{CT}_{2,(x,x')} \leftarrow \text{Encaps}_{2,\text{EP-KEM}}(\text{MPK}, (x, x'))$*, and computes*[7] $\text{CT}_{\text{sym}} \leftarrow \text{Enc}_{\text{K}}(M \| \text{CT}_{2,(x,x')})$*, and returns*

$$\text{CT}'_x = (\text{CT}_{\text{sym}}, \text{CT}_1, \text{CT}_{2,(x,x')}, k).$$

- $\text{Decrypt}'_{\text{PE}}(\text{MPK}', \text{SK}'_y, \text{CT}'_x)$*: On input the master public key* MPK'*, the secret key* SK'_y*, and the ciphertext* $\text{CT}'_x = (\text{CT}_{\text{sym}}, \text{CT}_1, \text{CT}_{2,(x,x')}, k)$*, if* $P_\kappa(x, y) = 1$*, then the decrypting user computes* $x' = \text{RPC}(k, \text{CT}_1)$ *and*

$$\text{K}' \leftarrow \text{Decaps}_{\text{EP-KEM}}(\text{MPK}, \text{SK}_{(y,*)}, (\text{CT}_1, \text{CT}_{2,(x,x')})).$$

  The user computes $(M' \| \text{CT}'_{2,(x,x')}) \leftarrow \text{Dec}_{\text{K}'}(\text{CT}_{\text{sym}})$*, and if* $\text{CT}'_{2,(x,x')} = \text{CT}_{2,(x,x')}$*, returns* M'*.*

Correctness. The scheme is correct, i.e., if $P_\kappa(x, y) = 1$, then $M' = M$, because the correctness of the P-KEM ensures that $\text{K} = \text{K}'$, and thus, $(M' \| \text{CT}'_{2,(x,x')}) = \text{Dec}_{\text{K}'}(\text{CT}_{\text{sym}}) = \text{Dec}_{\text{K}}(\text{CT}_{\text{sym}}) = \text{Dec}_{\text{K}}(\text{Enc}_{\text{K}}(M \| \text{CT}_{2,(x,x')})) = (M \| \text{CT}_{2,(x,x')})$.

Security. We prove the following theorem.

Theorem 1. *In Definition 9, if* $\Gamma_{\text{EP-KEM}}$ *is a decomposable CPA-secure P-KEM for the extended predicate* $\text{CCA}[P]$*,* RPC *is a random-prefix collision-resistant hash function and* $\text{SE} = (\text{Enc}_{\text{K}}, \text{Dec}_{\text{K}})$ *is an authenticated encryption scheme, such that the* RPC *is independent of* $\Gamma_{\text{EP-KEM}}$ *and* SE*, then* Γ'_{PE} *is CCA-secure.*

Proof. We prove this theorem in a series of games in which we start with the real CCA-security game: Game 0. Let $\text{CT}^*_{x^*} = (\text{CT}^*_{\text{sym}}, \text{CT}^*_1, \text{CT}^*_{2,(x^*,x'^*)}, k^*)$ denote the challenge ciphertext (with K^* being the decryption key) for the challenge predicate x^* and message M_β. Let q be the number of decryption queries, and let X_i denote the event that attacker $\mathcal{A}_{\text{CCA}}$ is successful in Game i.

Game 1: In this game, everything is the same as in Game 0, except that, in the first query phase, all decryption queries with $\text{CT}_1 = \text{CT}^*_1$ are rejected. Additionally, in both query phases, the decryption queries with $(\text{CT}_1, k) \neq (\text{CT}^*_1, k^*)$ and $x' = x'^*$ are rejected. The probability that $\text{CT}_1 = \text{CT}^*_1$ holds for any honestly generated ciphertext is $\frac{1}{\mathcal{G}}$. Furthermore, the probability that any x' for $(\text{CT}_1, k) \neq (\text{CT}^*_1, k^*)$ is such that $\text{RPC}(k, \text{CT}_1) = x' = x'^* = \text{RPC}(k^*, \text{CT}^*_1)$ is

[7] If one uses an authenticated encryption scheme with associated data [42], one can also treat $\text{CT}_{2,(x,x')}$ as associated data, as it does not need to be secret.

equal to $\Pr[(\mathrm{CT}_1, k) \neq (\mathrm{CT}_1^*, k^*) \wedge \mathrm{RPC}(k, \mathrm{CT}_1) = \mathrm{RPC}(k^*, \mathrm{CT}_1^*)] = \mathsf{Adv}_{\mathrm{RPC}}$. Hence, we have

$$|\Pr[X_0] - \Pr[X_1]| \leq \frac{q}{\mathcal{G}} + \mathsf{Adv}_{\mathrm{RPC}}.$$

Game 2: In this game, everything is the same as in Game 1, except that, in the second query phase, all decryption queries are rejected where $\mathrm{CT}_{\mathrm{sym}} \neq \mathrm{CT}_{\mathrm{sym}}^*$ holds, and the key $\mathrm{K} \leftarrow \mathrm{Decaps}_{\textbf{EP-KEM}}(\mathrm{MPK}, \mathrm{SK}_{(y,*)}, \mathrm{CT}_x)$ is such that $\mathrm{K} = \mathrm{K}^*$. Because this property can only hold if the ciphertext authenticity of the SE is broken, we have

$$|\Pr[X_1] - \Pr[X_2]| \leq \mathsf{Adv}_{\mathrm{SE,CAUT}}.$$

Game 3: In this game, everything is the same as in Game 2, except that, in the second query phase, all valid decryption queries are rejected where $\mathrm{CT}_{2,(x,x')} \neq \mathrm{CT}_{2,(x^*,x'^*)}^*$ holds, and $\mathrm{K} = \mathrm{K}^*$ (and thus, $\mathrm{CT}_1 = \mathrm{CT}_1^*$). Note that this can happen only if the ciphertext authenticity of SE is broken, because the attacker has to generate a valid ciphertext for the same key K^* and another message. Hence, we have

$$|\Pr[X_2] - \Pr[X_3]| \leq \mathsf{Adv}_{\mathrm{SE,CAUT}}.$$

Game 4: At this point, all ciphertexts that are queried in the second phase and that are not rejected are such that, for the keys, it holds that $\mathrm{K} \neq \mathrm{K}^*$. This follows from the fact that K is uniquely determined by MPK and CT_1 (and vice versa), and thus, if $\mathrm{K} = \mathrm{K}^*$, then $\mathrm{CT}_1 = \mathrm{CT}_1^*$. By extension, we have $(\mathrm{CT}_{\mathrm{sym}}, \mathrm{CT}_{2,(x,x')}, k) \neq (\mathrm{CT}_{\mathrm{sym}}^*, \mathrm{CT}_{2,(x^*,x'^*)}^*, k^*)$. (Note that, if $k \neq k^*$, we also have $\mathrm{CT}_{2,(x,x')} \neq \mathrm{CT}_{2,(x^*,x'^*)}^*$, which follows from rejecting all ciphertexts with $x' = x'^*$ in Game 1. From the fact that the EP-KEM is decomposable, it follows that $x' \neq x'^*$ implies $\mathrm{CT}_{2,(x,x')} \neq \mathrm{CT}_{2,(x^*,x'^*)}^*$.) For these cases, we had rejected the decryption queries (in Games 2 and 3). Because this game is the same as Game 3, we have

$$|\Pr[X_3] - \Pr[X_4]| = 0.$$

Game 5: In this game, everything is the same as in Game 4, except that we generate the challenge ciphertext as follows. Let $\mathcal{O}_{\mathrm{RPC}}$ denote the oracle that finds $k \in \{0,1\}^\lambda$ such that $\mathrm{RPC}(k, g) = z$ for any given $(g, z) \in \mathcal{G} \times \mathcal{Z}$. Because RPC is independent of the P-KEM and symmetric encryption scheme, this does not give the attacker any advantage. Then, the challenger generates $(\mathrm{K}^*, \mathrm{CT}_1, \mathrm{CT}_{2,(x^*,x'^*)}) \leftarrow \mathrm{Encaps}_{\textbf{EP-KEM}}(\mathrm{MPK}, (x^*, x'^*))$ for the challenge predicate x^* and randomly chosen x'^*, and queries the oracle $\mathcal{O}_{\mathrm{RPC}}$ with (CT_1, x'^*), which returns k^* if it exists. (Otherwise, it repeats the process of generating new ciphertexts until the oracle provides some output k^*. This likely succeeds because of the random-prefix collision resistance of the RPC. Intuitively, if many such inputs exist for which the oracle does not return a output, we can also find many g such that there exist at least two k, k' with $\mathrm{RPC}(k, g) = \mathrm{RPC}(k', g)$, which breaks the random-prefix collision resistance of the RPC.) The challenger then outputs the challenge ciphertext as $(\hat{\mathrm{K}}^*, \mathrm{CT}_1, \mathrm{CT}_{2,(x^*,x'^*)}, k^*)$, where $\hat{\mathrm{K}}^*$

is a randomly chosen key that replaces K^*. Because the attacker cannot make decryption queries for K^*, it can only distinguish this game from Game 4 by breaking the CPA-security of the EP-KEM. Therefore, we have

$$|\Pr[X_4] - \Pr[X_5]| \leq \mathsf{Adv}_{\text{EP-KEM,IND-CPA}}.$$

Game 6: In this game, everything is the same as in Game 5, except we replace the challenge message by a randomly generated message of the same length as M_β. By the ciphertext indistinguishability of the symmetric encryption scheme, no attacker can distinguish Game 5 from Game 6, i.e.,

$$|\Pr[X_5] - \Pr[X_6]| \leq \mathsf{Adv}_{\text{SE,CIND}}.$$

Summary: In this final game, because the ciphertext is for a random message, the success probability of the attacker is $\frac{1}{2}$, i.e., $\Pr[X_6] = \frac{1}{2}$. This gives us the following upper bound on the advantage of the attacker in the real security game:

$$\begin{aligned}\mathsf{Adv}_{\text{PE,IND-CCA}} &= \left|\Pr[X_0] - \frac{1}{2}\right| \\ &\leq \frac{q}{\mathcal{G}} + \mathsf{Adv}_{\text{RPC}} + 2\mathsf{Adv}_{\text{SE,CAUT}} \\ &\quad + \mathsf{Adv}_{\text{EP-KEM,IND-CPA}} + \mathsf{Adv}_{\text{SE,CIND}}.\end{aligned}$$

Since all advantages on the right-hand side are negligible in λ, it also holds that $\mathsf{Adv}_{\text{PE,IND-CCA}}$ is negligible in λ. □

Remark 1. The predicate extension x'^* associated with the ciphertext is determined during encryption by the challenger, and not by the attacker. Possibly, to obtain a fully CCA-secure hybrid PE scheme, one can make an AND-composition of a selectively secure "all-or-one-identity" IB-KEM and a fully secure P-KEM. In this case, a selectively secure IB-KEM is sufficient, because the challenger can generate the predicate extension x'^* before generating the challenge ciphertext. Formalizing such a composition is however not trivial, for instance, because the master public keys of fully secure and selectively secure schemes have a different structure and are thus difficult to split (to build the AND-composition). We therefore believe that such a generic transformation is not as simple to prove generically secure as the proposed transformations in this work. Additionally, it may require a combination of various (complex) proof techniques.

Remark 2. Our proof techniques are similar to but also slightly different from the ACIK techniques. In fact, by feeding $\mathrm{CT}_{2,(x,x')}$ through the authenticated symmetric encryption scheme, a part of the proof is more similar to the BK-approach. However, unlike BK, we use the same key K to encrypt and authenticate the message M, and to authenticate $\mathrm{CT}_{2,(x,x')}$. To ensure that this can be done securely, we require MPK, M, CT_M and CT_1 to be highly dependent on one another. This property is arguably easier to verify than ACIK's rejection property. Furthermore, we explicitly require CT_1 to be sufficiently random (which is a requirement that is inspired by the KV scheme [38]). Lastly, note that our property that, for any $(\hat{x}, \hat{x}') \in \mathcal{X}'_\kappa$ with $\hat{x}' \neq x'$, we have that any $\mathrm{CT}_{2,(\hat{x},\hat{x}')}$ that is valid for CT_1 is such that $\mathrm{CT}_{2,(\hat{x},\hat{x}')} \neq \mathrm{CT}_{2,(x,x')}$, is similar to ACIK's unique-split property.

3.3 Variation on the construction: special decomposable EP-KEM

One of the differences between our transformation and the transformation by Abe et al. [1] is that we do not require the $\mathrm{CT}_{2,(x,x')}$ part to be uniquely defined from CT_1. Instead, we require the encapsulated key K to be uniquely determined by CT_1. We do this, because many PE schemes do not uniquely determine $\mathrm{CT}_{2,(x,x')}$ and do uniquely determine K from CT_1, e.g., the unbounded ABE schemes in [43,4]. Furthermore, such a deterministic property should also assume that the second ciphertext part $\mathrm{CT}_{2,(x,x')}$ is not delegatable in some way. For example, in many ABE schemes, one can simply drop certain ciphertext components such that this yields a valid ciphertext for a smaller set (in KP-ABE) or a more restricted policy (in CP-ABE).

In many cases, however, we can decompose the ciphertext in such a way that one part is dedicated to the predicate extension x' only. In this case, the ciphertext is of the form

$$\mathrm{CT}_{(x,x')} = (\mathrm{K}, \mathrm{CT}_1, \mathrm{CT}_{2,x}, \mathrm{CT}_{3,x'}),$$

such that only $\mathrm{CT}_{3,x'}$ depends on x' and it is uniquely determined by x', CT_1 and MPK. In this case, we can make two different variants of the generic CCA-secure construction. Instead of including $\mathrm{CT}_{2,(x,x')}$ in the payload of the symmetric encryption scheme, include only $\mathrm{CT}_{2,x}$. Alternatively, we can include $\mathrm{CT}_{2,x}$ in the input to the RPC hash, such that the indistinguishability between Game 2 and 3 follows from the target-collision resistance of the RPC hash. Furthermore, because $\mathrm{CT}_{3,x'}$ is uniquely determined by x', CT_1, and MPK, $\mathrm{CT}_{3,x'}$ cannot differ from the challenge ciphertext if x' and CT_1 are equal to the challenge ciphertext, which is the case if $\mathrm{K} = \mathrm{K}^*$. With this latter approach, the key-encapsulation and data-encapsulation mechanisms are also strictly separated, which can be advantageous in the implementation of the schemes. In the case that $\mathrm{CT}_{2,x}$ is also uniquely determined by x and CT_1, we can also include x in the hash and leave out $\mathrm{CT}_{2,x}$. (In the case that x is some fixed-length predicate such as an identity, we can leave out x altogether. For variable-length predicates such as policies, we have to include the predicate x to ensure that $\mathrm{CT}_{2,x}$ is not altered, e.g., by delegating the ciphertext to a more restricted policy.)

3.4 Variation on the construction: non-decomposable EP-KEM

To convert extended-predicate schemes that do not have decomposable ciphertexts, we can also base our second step of the transformation on a more generic conversion technique than ACIK [1], such as CHK [25] or BK [19]. To apply those techniques, we can treat the extended predicate (x', y') similarly as the identity in those transformations. For example, as in the CHK-transformation, we can embed the verification key in the ciphertext's extended predicate x', and sign the resulting ciphertext with the associated signing key of the one-time signature scheme. Recall, however, that both these methods provide trade-offs in various practical aspects. That is, OTSs provide a significant efficiency trade-off, and the BK-approach induces a higher storage overhead and relies on more primitives.

4 New predicate-extension transformations

We give a high-level description of a concrete predicate-extension transformation (for which we provide a formal description in the appendix) for pairing-based ABE. Roughly, this transformation and its security proof follow a similar approach as Attrapadung [10]. In particular, we take as input a secure PE scheme (satisfying some properties) and perform a predicate transformation on it, i.e., an AND-composition (on the key) of the original scheme and an "all-or-one-identity" IBE scheme. To this end, we adapt the key-policy augmentation transformation of Attrapadung [10]. Our adaptation differs from the original in two ways. First, we ensure that, for the extended key predicate $(y, *)$, we can generate a key for all identities (y, y'). Second, we re-use the randomness used in the keys of the original scheme to randomize the partial "all-or-one-identity" key. In this way, we minimize the amount of randomness, and ultimately, the computational costs. For schemes with an admissible pair encoding[8], we use the key randomness r that is used in the polynomial that masks the master-key α, i.e., $\alpha + rb$.

4.1 "All-or-one-identity" IBE

For the predicate extension, we use an "all-or-one-identity" IBE scheme. On a high level, we define the "all-or-one-identity" IBE scheme with identities $x', y' \in \mathbb{Z}_p = \mathcal{Z}$ as follows:

$$\begin{aligned} \mathrm{MPK}' &= (g, h, e(g,h)^{\alpha s}, g^{b'_0}, g^{b'_1}), \\ \mathrm{SK}'_{y'} &= (h^{\alpha + r(b'_0 + y' b'_1)}, h^r), \mathrm{SK}'_* = (h^{\alpha + r b'_0}, h^{r b'_1}, h^r), \\ \mathrm{CT}'_{x'} &= (M \cdot e(g,h)^{\alpha s}, g^{s(b'_0 + x' b'_1)}, g^s). \end{aligned}$$

With SK'_*, we can generate $\mathrm{SK}'_{y'}$ for any $y' \in \mathbb{Z}_p$, by computing:

$$h^{\alpha + r b'_0} \cdot \left(h^{r b'_1}\right)^{y'} = h^{\alpha + r(b'_0 + y' b'_1)}.$$

Note that this scheme is similar to the Boneh-Boyen IBE1 scheme [16], which is selectively secure, with the modification that it allows for the generation of a secret key that can be used for all identities.

[8] This is a pair encoding with some additional properties, used in [10]. Note that any secure pair encoding can be converted into an admissible pair encoding by applying the Layer-Trans transformation in [10].

4.2 AND-composition with a PE

The transformation of a PE for predicate P to the PE with extended predicate CCA[P] consists of an AND-composition with the "all-or-one-identity" IBE. For example, consider the following scheme:

$$\begin{aligned}\mathrm{MPK} &= (g, h, e(g,h)^{\alpha s}, g^{\mathbf{b}}),\\ \mathrm{SK}_y &= (h^{\mathbf{r}}, h^{\mathbf{k}(\alpha,\mathbf{r},\mathbf{b},y)}),\\ \mathrm{CT}_x &= (M \cdot e(g,h)^{\alpha s}, g^{\mathbf{s}}, g^{\mathbf{c}(\mathbf{s},\mathbf{b},x)}),\end{aligned}$$

where $\mathbf{r}$, $\mathbf{k}$, $\mathbf{s}$, and $\mathbf{c}$ denote the vectors that describe the secret key and ciphertext, respectively. Then, the transformed scheme is of the form:

$$\begin{aligned}\mathrm{MPK} &= (g, h, e(g,h)^{\alpha s}, g^{\mathbf{b}}, g^{b'_0}, g^{b'_1}),\\ \mathrm{SK}_{(y,y')} &= \begin{cases} (h^{\mathbf{r}}, h^{\mathbf{k}(\alpha_1,\mathbf{r},\mathbf{b},y)}, h^{\alpha-\alpha_1+rb'_0}, h^{rb'_1}) & \text{if } y' = *,\\ (h^{\mathbf{r}}, h^{\mathbf{k}(\alpha_1,\mathbf{r},\mathbf{b},y)}, h^{\alpha-\alpha_1+r(b'_0+y'b'_1)}) & \text{if } y' \in \mathbb{Z}_p \end{cases}\\ \mathrm{CT}_{(x,x')} &= (M \cdot e(g,h)^{\alpha s}, g^{\mathbf{s}}, g^{\mathbf{c}(\mathbf{s},\mathbf{b},x)}, g^{s(b'_0+x'b'_1)}),\end{aligned}$$

where $\alpha_1 \in_R \mathbb{Z}_p$. We formulate this transformation in the pair encodings and the predicate encodings frameworks in Appendices C and D. We prove security of the transformation in several ways. We show that the transformation for pair encodings preserves the symbolic security[9] and perfectly master-key hiding properties. Because we re-use the randomness of the key and ciphertext encodings of the original scheme, the transformation can also be formulated in the predicate encodings framework [48,26], and its security follows from the similarity between the perfectly master-key hiding and α-privacy—the security notion for predicate encodings [7].

4.3 Decomposability of the ciphertexts

The resulting extended-predicate encryption scheme is decomposable (and even special decomposable):

$$\mathrm{CT}_{(x,x')} = (\underbrace{M \cdot e(g,h)^{\alpha s}}_{\mathrm{CT_M}}, \underbrace{g^{\mathbf{s}}}_{\mathrm{CT}_1}, \underbrace{g^{\mathbf{c}(\mathbf{s},\mathbf{b},x)}}_{\mathrm{CT}_{2,x}}, \underbrace{g^{s(b'_0+x'b'_1)}}_{\mathrm{CT}_{3,x'}}),$$

and can be easily transformed in a KEM by removing $\mathrm{CT_M}$ and setting $\mathrm{K} = e(g,h)^{\alpha s}$. For a fixed master public key MPK, the key K is then uniquely defined by CT_1 and vice versa, and CT_1 is generated uniformly at random. For $\hat{x}' \neq x'$, we have that $g^{s(b'_0+\hat{x}'b'_1)} \neq g^{s(b'_0+x'b'_1)}$. We can also define a different split, e.g., $\mathrm{CT}_1 = g^s$ and push the rest of $g^{\mathbf{s}}$ in $\mathrm{CT}_{2,x}$. Note that, if one chooses to encapsulate some randomly generated symmetric key $\mathrm{K} = M$, then $M \cdot e(g,h)^{\alpha s}$ should be included in CT_1 to ensure that K is uniquely defined by CT_1 and MPK.

[9] Due to the strong relationship between the selective symbolic property and selective security [8,10], it may follow from the selective symbolic property that the AND-composition of a selectively secure PE and the selectively secure "all-or-one-identity" IBE is selectively secure.

Table 2: Comparison of the storage and computational costs of the P-KEM part of several CCA-secure variants of CGW-IBE and the fully secure variant of RW13. The storage costs are expressed in bytes and the timings are expressed in milliseconds. The lowest costs are typeset in **bold**, and for 100 attributes, we also include the increase in costs compared to the CPA-secure P-KEM version. For RWAC, we consider inputs of 1, 10 and 100 attributes. (Note that we use compressed point representation to minimize the storage costs.)

Variant	$\lvert\text{MPK}\rvert$	$\lvert\text{SK}_{\mathcal{S}}\rvert$	$\lvert\text{CT}_{\mathbb{A}}\rvert$	KeyGen	Encrypt	Decrypt
CPA	576	448	192	4.10	4.50	1.56
FO	**576**	1024	480	**4.10**	**4.50**	6.06
Ours	672	**576**	**208**	6.13	**4.50**	**4.46**

(a) CGW-IBE [26], the fully secure and anonymous variant of BB-IBE1 [16]

Variant	$\lvert\text{MPK}\rvert$	$\lvert\text{SK}_{\mathcal{S}}\rvert$				$\lvert\text{CT}_{\mathbb{A}}\rvert$			
		1	10	100	Increase	1	10	100	Increase
CPA	768	768	4,224	38,784	-	384	2,976	28,896	-
FO	**768**	1,536	4,992	39,552	2%	672	3,264	29,184	1%
Del.	**768**	99,072	102,528	137,088	253%	37,248	39,840	65,760	128%
Ver.	**768**	**768**	**4,224**	**38,784**	0%	672	3,264	29,184	1%
Ours	960	1,152	4,608	39,168	1%	**480**	**3,072**	**29,008**	0.4%

(b) RWAC, the fully secure variant of RW13 [43] in AC17 [4,10] (storage costs)

Variant	KeyGen				Encrypt				Decrypt			
	1	10	100	Increase	1	10	100	Increase	1	10	100	Increase
CPA	8.40	46.2	424	-	6.73	46.6	445	-	3.84	16.6	145	-
FO	8.40	46.2	424	0.4%	**6.73**	**46.6**	**445**	0%	10.6	63.1	590	307%
Del.	1082	1121	1499	255%	573	614	1013	127%	186	200	329	127%
Ver.	**8.37**	**46.0**	**422**	0%	11.1	51.0	449	0.9%	10.5	35.8	292	101%
Ours	12.5	50.1	427	1%	8.21	48.2	448	0.7%	**6.0**	**18.7**	**148**	2%

(c) RWAC, the fully secure variant of RW13 [43] in AC17 [4,10] (computational costs)

5 Performance analysis of concrete constructions

To illustrate the benefits of our transformation with respect to the efficiency compared to other generic transformation techniques, we analyze the storage and computational costs of two concrete constructions. On a high level, the efficiency of the transformation depends on the P-KEM part that encapsulates the symmetric key, and the other primitives used. For simplicity, we assume that all transformations considered in the introduction can support KEM variants, which encapsulate a symmetric key in the P-KEM part and use this key to symmetrically encrypt the plaintext.

We analyze the efficiency of the P-KEM part by implementing and benchmarking the available CCA-transformed versions of two schemes. (We leave out the KW [39] transformation due to its evident blowup in costs as shown in Section 1.2.) The first scheme is CGW-IBE, the anonymous IBE scheme by Chen, Gay and Wee [26]. The second scheme is RWAC, the fully secure variant [4,10] of the CP-ABE scheme by Rouselakis and Waters [43]. We provide full descriptions of the schemes and their CCA-secure variants in Appendices F and G. For CGW-IBE, we provide a more optimized variant implied by our CCA-secure variant, which, interestingly, resembles the first Kiltz-Vahlis scheme [38]. For RWAC, we approximate the efficiency of the FO, delegation and verifiability-based transformations. For FO, we add the encryption costs to the decryption costs (for same-length inputs). Note that this also includes the public-key storage cost in the secret key size as this is required for re-encryption. For verifiability-based transformations, we add one attribute in the ciphertext-policy input, and multiply the decryption costs by a factor 2. For delegation-based transformations, we assume that the length of the verification key of the used OTS is at least 128 bits (at the 128-bit security level), and thus, that the key set is extended with $2 \cdot 128 = 256$ attributes, and the ciphertext policy with 128 attributes. We have implemented the schemes in Rust[10] using the BLS12-381 crate provided by the zkCrypto group [51].

Table 2 summarizes the benchmarks obtained by running the code on an AMD Ryzen 7 3700X CPU, with a frequency of 4.1 GHz. For CGW-IBE, our keys and ciphertexts are both the smallest. For RWAC, we observe that our ciphertexts are generally the smallest, while the keys are only a little larger than the smallest. For CGW-IBE, we observe that FO key generation is significantly faster than ours, while our decryption is in turn faster than FO. For RWAC, we observe that our decryption is by far the most efficient, i.e., at least a factor 2 than all other variants. Furthermore, the key generation and encryption costs are only milliseconds slower than the most efficient variants. In conclusion, all transformations except for ours incur a significant trade-off: either attaining a large overhead in the key or ciphertext sizes, or incurring a very large overhead in at least one of the algorithms. In contrast, with respect to the decryption algorithm, our transformation outperforms all other transformations, with incredibly little sacrifice in key generation and encryption efficiency.

6 Future work

Throughout this work, we have mentioned several interesting directions for future work. First, because the second step of the transformation can be done fully generically, it may be used to convert (decomposable) post-quantum PE as well, e.g., by making an AND-composition of a post-quantum PE and "all-or-one-identity" IBE. Second, we might be able to obtain even more efficient transformations by using a selectively secure "all-or-one-identity" IBE (Remark

[10] The code is available at https://github.com/leonbotros/pe_cca.

1). Finally, while this work focuses on predicate encryption, it might be applicable to an even larger class of encryption schemes, e.g., functional encryption [20], which contains PE.

7 Conclusion

We have presented a new two-step approach to achieving CCA-security generically in PE schemes, which aims to convert PE schemes as efficiently as possible. Additionally, for each of these steps, we have proposed a new transformation. For the second-step transformation, we have generalized the ACIK-transform [1], which can now be applied to any PE scheme that is decomposable and for which the predicate can be securely extended. Compared to the more generic CHK- and BK-approaches, ACIK provides less storage overhead and relies on fewer primitives. For the first-step transformation, we have proposed a new predicate-extension transformation that can be applied to any pairing-based schemes that can be captured in the pair and predicate encodings frameworks. Compared to existing (implicitly-described) predicate-extension techniques, ours is more efficient. Notably, for CP-ABE, existing such techniques are very inefficient. To show that our predicate-extension transformation indeed yields interesting improvements on existing ones, we have implemented two schemes: CGW-IBE and RWAC. Especially for RWAC, the results are convincing. For all algorithms, our transformation incurs only a small constant overhead compared to the CPA-secure variant. In contrast, all other transformations incur a sizable overhead in at least one of the algorithms. In fact, our transformation is at least twice as fast in the decryption algorithm compared to all other transformations.

Acknowledgments. The authors would like to thank Eike Kiltz for helping out with a part of the security proof, and Greg Alpár for proofreading the paper. They would also like to thank the reviewers for their helpful comments and suggestions. This work is partly funded by the NWO under the project "Encryption for all" (project number CS.002) of the research programme "Topsectoren 18-19 Cybersecurity", which is (partly) financed by the Dutch Research Council (NWO).

References

1. Abe, M., Cui, Y., Imai, H., Kiltz, E.: Efficient hybrid encryption from id-based encryption. Des. Codes Cryptogr. **54**(3), 205–240 (2010)
2. Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or fuzzy IBE) from lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC. LNCS, vol. 7293, pp. 280–297. Springer (2012)
3. Agrawal, S., Chase, M.: A study of pair encodings: Predicate encryption in prime order groups. In: TCC. pp. 259–288. Springer (2016)

4. Agrawal, S., Chase, M.: Simplifying design and analysis of complex predicate encryption schemes. In: EUROCRYPT. pp. 627–656. Springer (2017)
5. Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional encryption for inner product predicates from learning with errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT. LNCS, vol. 7073, pp. 21–40. Springer (2011)
6. Ambrona, M.: Generic negation of pair encodings. In: Garay, J.A. (ed.) PKC. LNCS, vol. 12711, pp. 120–146. Springer (2021)
7. Ambrona, M., Barthe, G., Schmidt, B.: Generic transformations of predicate encodings: Constructions and applications. In: Katz, J., Shacham, H. (eds.) CRYPTO. LNCS, vol. 10401, pp. 36–66. Springer (2017)
8. Attrapadung, N.: Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In: EUROCRYPT. pp. 557–577. Springer (2014)
9. Attrapadung, N.: Dual system encryption framework in prime-order groups via computational pair encodings. In: ASIACRYPT. pp. 591–623. Springer (2016)
10. Attrapadung, N.: Unbounded dynamic predicate compositions in attribute-based encryption. In: EUROCRYPT. pp. 34–67. Springer (2019)
11. Attrapadung, N., Imai, H.: Dual-policy attribute based encryption. In: Abdalla, M., Pointcheval, D., Fouque, P., Vergnaud, D. (eds.) ACNS. LNCS, vol. 5536, pp. 168–185 (2009)
12. Beimel, A.: Secure schemes for secret sharing and key distribution (1996)
13. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: S&P. pp. 321–334. IEEE (2007)
14. Blömer, J., Liske, G.: Construction of fully cca-secure predicate encryptions from pair encoding schemes. In: Sako, K. (ed.) CT-RSA. LNCS, vol. 9610, pp. 431–447. Springer (2016)
15. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications. In: Simon, J. (ed.) Proceedings of the 20th Annual ACM Symposium on Theory of Computing. pp. 103–112. ACM (1988)
16. Boneh, D., Boyen, X.: Efficient selective-id secure identity-based encryption without random oracles. In: EUROCRYPT. pp. 223–238. Springer (2004)
17. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. SIAM J. Comput. **36**(5), 1301–1328 (2007)
18. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO. LNCS, vol. 2139, pp. 213–229. Springer (2001)
19. Boneh, D., Katz, J.: Improved efficiency for cca-secure cryptosystems built using identity-based encryption. In: Menezes, A. (ed.) CT-RSA. LNCS, vol. 3376, pp. 87–103. Springer (2005)
20. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC. LNCS, vol. 6597, pp. 253–273. Springer (2011)
21. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) TCC. LNCS, vol. 4392, pp. 535–554. Springer (2007)
22. Boyen, X.: Attribute-based functional encryption on lattices. In: Sahai, A. (ed.) TCC. LNCS, vol. 7785, pp. 122–142. Springer (2013)
23. Boyen, X., Waters, B.: Anonymous hierarchical identity-based encryption (without random oracles). In: Dwork, C. (ed.) CRYPTO. LNCS, vol. 4117, pp. 290–307. Springer (2006)
24. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRYPT. LNCS, vol. 2656, pp. 255–271. Springer (2003)

25. Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT. LNCS, vol. 3027, pp. 207–222. Springer (2004)
26. Chen, J., Gay, R., Wee, H.: Improved dual system ABE in prime-order groups via predicate encodings. In: EUROCRYPT. pp. 595–624. Springer (2015)
27. Chen, J., Wee, H.: Dual system groups and its applications — compact hibe and more. Cryptology ePrint Archive, Report 2014/265 (2014)
28. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. **33**(1), 167–226 (2003)
29. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M.J. (ed.) CRYPTO. LNCS, vol. 1666, pp. 537–554. Springer (1999)
30. Gentry, C.: Practical identity-based encryption without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT. LNCS, vol. 4004, pp. 445–464. Springer (2006)
31. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Dwork, C. (ed.) STOC. pp. 197–206. ACM (2008)
32. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) STOC. pp. 545–554. ACM (2013)
33. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: CCS. ACM (2006)
34. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. Cryptology ePrint Archive, Report 2006/309 (2006)
35. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the fujisaki-okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC. LNCS, vol. 10677, pp. 341–371. Springer (2017)
36. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Smart, N.P. (ed.) EUROCRYPT. LNCS, vol. 4965, pp. 146–162. Springer (2008)
37. Kiltz, E., Galindo, D.: Direct chosen-ciphertext secure identity-based key encapsulation without random oracles. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP. LNCS, vol. 4058, pp. 336–347. Springer (2006)
38. Kiltz, E., Vahlis, Y.: CCA2 secure IBE: standard model efficiency through authenticated symmetric encryption. In: Malkin, T. (ed.) CT-RSA. LNCS, vol. 4964, pp. 221–238. Springer (2008)
39. Koppula, V., Waters, B.: Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO. LNCS, vol. 11693, pp. 671–700. Springer (2019)
40. Lewko, A., Waters, B.: Decentralizing attribute-based encryption. Cryptology ePrint Archive, Report 2010/351 (2010)
41. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Ortiz, H. (ed.) STOC. pp. 427–437. ACM (1990)
42. Rogaway, P.: Authenticated-encryption with associated-data. In: Atluri, V. (ed.) CCS. pp. 98–107. ACM (2002)
43. Rouselakis, Y., Waters, B.: Practical constructions and new proof methods for large universe attribute-based encryption. In: CCS. pp. 463–474. ACM (2013)
44. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: EUROCRYPT. pp. 457–473. Springer (2005)

45. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO. LNCS, vol. 196, pp. 47–53. Springer (1984)
46. Shoup, V.: Why chosen ciphertext security matters. IBM Research Report RZ 3076 (November 1998)
47. Tomida, J., Kawahara, Y., Nishimaki, R.: Fast, compact, and expressive attribute-based encryption. Des. Codes Cryptogr. **89**(11), 2577–2626 (2021)
48. Wee, H.: Dual system encryption via predicate encodings. In: TCC. pp. 616–637. Springer (2014)
49. Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.: Generic constructions for chosen-ciphertext secure attribute based encryption. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC. LNCS, vol. 6571, pp. 71–89. Springer (2011)
50. Yamada, S., Attrapadung, N., Santoso, B., Schuldt, J.C.N., Hanaoka, G., Kunihiro, N.: Verifiable predicate encryption and applications to CCA security and anonymous predicate authentication. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC. LNCS, vol. 7293, pp. 243–261. Springer (2012)
51. ZkCrypto: Zero-knowledge cryptography in Rust (2020), https://github.com/zkcrypto

A Predicate key encapsulation

In the key-encapsulation variant of predicate encryption (Definition 27), which we call predicate KEM (P-KEM), we replace Encrypt by Encaps and Decrypt by Decaps, where Encaps also outputs a symmetric key, and Decaps outputs a symmetric key instead of a plaintext message. This symmetric key is used to symmetrically encrypt the data.

Definition 10 (Predicate key-encapsulation mechanism (P-KEM)). *A predicate key-encapsulation mechanism for a predicate family $P = \{P_\kappa\}_{\kappa \in \mathbb{N}^c}$ over a message space $\mathcal{M} = \{M_\lambda\}_{\lambda \in \mathbb{N}}$ consists of four algorithms:*

- $\mathrm{Setup}(\lambda, \mathrm{par}) \rightarrow (\mathrm{MPK}, \mathrm{MSK})$*: On input the security parameter λ and parameters* par*, this probabilistic algorithm generates the domain parameters, the master public key* MPK *and the master secret key* MSK*. In addition, κ is set to $\kappa = (p, \mathrm{par})$, where p denotes a natural number.*
- $\mathrm{KeyGen}(\mathrm{MSK}, y) \rightarrow \mathrm{SK}_y$*: On input the master secret key* MSK *and some $y \in \mathcal{Y}_\kappa$, this probabilistic algorithm generates a secret key* SK_y*.*
- $\mathrm{Encaps}(\mathrm{MPK}, x) \rightarrow (\mathrm{K}, \mathrm{CT}_x)$*: On input the master public key* MPK *and some $x \in \mathcal{X}_\kappa$, this probabilistic algorithm generates an encapsulated symmetric key* K *and a ciphertext* CT_x*.*
- $\mathrm{Decaps}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x) \rightarrow \mathrm{K}$*: On input the master public key* MPK*, the secret key* SK_y*, and the ciphertext* CT_x*, if $P_\kappa(x, y) = 1$, then it returns the encapsulated symmetric key* K*. Otherwise, it returns an error message $\perp$.*

Correctness. For all par, $M \in \mathcal{M}_\lambda$, $x \in \mathcal{X}_\kappa$, and $y \in \mathcal{Y}_\kappa$ such that $P_\kappa(x, y) = 1$,

$$\Pr[(\mathrm{MPK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(1^\lambda); (\mathrm{K}, \mathrm{CT}_x) \leftarrow \mathrm{Encaps}(\mathrm{MPK}, x);$$
$$\mathrm{Decaps}(\mathrm{MPK}, \mathrm{KeyGen}(\mathrm{MSK}, y)), \mathrm{CT}_x) \neq \mathrm{K}] \leq \mathrm{negl}(\lambda).$$

Full security against chosen-plaintext attacks. The full security model for P-KEM is defined similarly as that for PE (Definition 2). The crucial difference between the two is that the goal of the attacker is to distinguish a symmetric key produced by the encapsulation algorithm from a randomly generated key.

Definition 11 (CPA-security for P-KEM). *We define the security game* IND-CPA(λ) *between challenger and attacker as follows:*

- ***Setup phase:*** *The challenger runs* $\mathrm{Setup}(\lambda)$ *to obtain* MPK *and* MSK*, and sends the master public key* MPK *to the attacker.*
- ***First query phase:*** *The attacker queries secret keys for $y \in \mathcal{Y}$, and obtains* $\mathrm{SK}_y \leftarrow \mathrm{KeyGen}(\mathrm{MSK}, y)$ *in response.*
- ***Challenge phase:*** *The attacker specifies some $x^* \in \mathcal{X}$ such that for all y in the first key query phase, we have $P(x^*, y) = 0$, and sends these to the challenger. The challenger first encapsulates a key under x^*, i.e.,* $(\mathrm{K}^*, \mathrm{CT}_{x^*}) \leftarrow \mathrm{Encaps}(\mathrm{MPK}, x^*)$*, and then flips a coin $\beta \in_R \{0, 1\}$. If $\beta = 0$,*

the key K^* *is replaced by a value that is selected uniformly at random from the key space. The challenger then sends the resulting encapsulation key* K^* *and ciphertext* CT_{x^*} *to the attacker.*

- ***Second query phase:*** *This phase is identical to the first query phase, with the additional restriction that the attacker can only query* $y \in \mathcal{Y}$ *such that* $P(x^*, y) = 0$*.*
- ***Decision phase:*** *The attacker outputs a guess* β' *for* β*.*

The advantage of the attacker is defined as $\mathsf{Adv}_{\text{P-KEM,IND-CPA}} = |\Pr[\beta' = \beta] - \frac{1}{2}|$*. A scheme is fully secure if all polynomial-time attackers have at most a negligible advantage in this security game, i.e.,* $\mathsf{Adv}_{\text{P-KEM,IND-CPA}} \leq \mathrm{negl}(\lambda)$*.*

In the selective security model, the attacker commits to the predicate $x^* \in \mathcal{X}$ *before the Setup phase.*

B More efficient transformation from PE to P-KEM

Let $\mathit{\Gamma}_{\text{PE}} = (\text{Setup}, \text{KeyGen}, \text{Encaps}, \text{Decaps})$ be a decomposable predicate encryption scheme for the predicate family $P = \{P_\kappa\}_\kappa$ with $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$. Suppose that the operation on the group in which CT_{M} lives is multiplicative[11] and its operator is $\cdot$, and in particular, that $\mathrm{CT}_{\mathrm{M}} = M \cdot \mathrm{rand}$, where rand is some random element in the group in which CT_{M} lives. Let id denote the identity in this group. Then, we can generically define Encaps and Decaps from Encrypt and Decrypt as follows.

- $\mathrm{Encaps}(\mathrm{MPK}, x)$: Let $(\mathrm{CT}_{\mathrm{M}}, \mathrm{CT}_1, \mathrm{CT}_{2,x}) \leftarrow \mathrm{Encrypt}(\mathrm{MPK}, x, \mathrm{id})$. Then, this algorithm outputs $\mathrm{K} = \mathrm{CT}_{\mathrm{M}}$ as the symmetric key and $(\mathrm{CT}_1, \mathrm{CT}_{2,x})$ as the rest of the ciphertext.
- $\mathrm{Decaps}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x)$: This algorithm outputs the decapsulated symmetric key as $\mathrm{K}' \leftarrow \mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, (\mathrm{id}, \mathrm{CT}_1, \mathrm{CT}_{2,x}))^{-1}$.

The correctness of the P-KEM follows from the correctness of the PE:

$$\begin{aligned}
\mathrm{Decaps}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x) &= \mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, (\mathrm{id}, \mathrm{CT}_1, \mathrm{CT}_{2,x}))^{-1} \\
&= \mathrm{K} \cdot \mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, (\mathrm{id} \cdot \mathrm{K}, \mathrm{CT}_1, \mathrm{CT}_{2,x}))^{-1} \\
&= \mathrm{K} \cdot \mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{Encrypt}(\mathrm{MPK}, x, \mathrm{id}))^{-1} \\
&= \mathrm{K} \cdot \mathrm{id}^{-1} = \mathrm{K}.
\end{aligned}$$

The CPA-security of the P-KEM also follows readily from the PE. Let $\mathcal{A}_{\text{P-KEM}}$ be an attacker on the P-KEM, i.e., which can distinguish for a given $(\mathrm{K}, \mathrm{CT}_1, \mathrm{CT}_{2,x})$ whether K is a symmetric key or K is random. Then, it can be used to construct an attacker $\mathcal{A}_{\text{PE}}$ for the PE scheme. Suppose $(\mathrm{CT}_{\mathrm{M}}, \mathrm{CT}_1, \mathrm{CT}_{2,x})$ is the challenge ciphertext for M_0 or M_1. Then, pick $\beta \in_R \{0,1\}$ and send $(\mathrm{K} = \mathrm{CT}_{\mathrm{M}}/M_\beta, \mathrm{CT}_1, \mathrm{CT}_{2,x})$ to attacker $\mathcal{A}_{\text{P-KEM}}$. If it outputs that K is a symmetric key, then attacker $\mathcal{A}_{\text{PE}}$ outputs β as the guess, and otherwise, it outputs $1 - \beta$ as the guess. The advantage of $\mathcal{A}_{\text{P-KEM}}$ is equal to the advantage of $\mathcal{A}_{\text{PE}}$.

[11] Something similar works for other algebraic groups such as additive groups as well.

C Pair encodings

Notation. We denote $a : \mathbf{A}$ to substitute variable a by a matrix or vector $\mathbf{A}$. We define $\mathbf{1}_{i,j} \in \mathbb{Z}_p^{d_1 \times d_2}$ as the matrix with 1 in the i-th row and j-th column, and 0 everywhere else, and similarly $\mathbf{1}_i$ and $\overline{\mathbf{1}}_i^\intercal$ as the row and column vectors with 1 in the i-th entry and 0 everywhere else.

C.1 Pair encoding schemes

We give the definitions of pair encoding schemes, and their associated security notions: selective and co-selective symbolic properties.

Definition 12 (Pair encoding schemes (PES) [4]). *A pair encoding scheme for a predicate family $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$, indexed by $\kappa = (p, \mathrm{par})$, where* par *specifies some parameters, is given by four deterministic polynomial-time algorithms as described below.*

– $\mathrm{Param}(\mathrm{par}) \to n$*: On input* par*, the algorithm outputs $n \in \mathbb{N}$ that specifies the number of common variables, which are denoted as $\mathbf{b} = (b_1, ..., b_n)$.*

– $\mathrm{EncKey}(y, p) \to (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \mathbf{b}))$*: On input $p \in \mathbb{N}$ and $y \in \mathcal{Y}_\kappa$, this algorithm outputs a vector of polynomials $\mathbf{k} = (k_1, ..., k_{m_3})$ defined over non-lone variables $\mathbf{r} = (r_1, ..., r_{m_1})$ and lone variables $\hat{\mathbf{r}} = (\hat{r}_1, ..., \hat{r}_{m_2})$. Specifically, the polynomial k_i is expressed as*

$$k_i = \delta_i \alpha + \sum_{j \in [m_2]} \delta_{i,j} \hat{r}_j + \sum_{j \in [m_1], k \in [n]} \delta_{i,j,k} r_j b_k,$$

for all $i \in [m_3]$, where $\delta_i, \delta_{i,j}, \delta_{i,j,k} \in \mathbb{Z}_p$.

– $\mathrm{EncCt}(x, p) \to (w_1, w_2, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}))$*: On input $p \in \mathbb{N}$ and $x \in \mathcal{X}_\kappa$, this algorithm outputs a vector of polynomials $\mathbf{c} = (c_1, ..., c_{w_3})$ defined over non-lone variables $\mathbf{s} = (s, s_2, ..., s_{w_1})$ and lone variables $\hat{\mathbf{s}} = (\hat{s}_1, ..., \hat{s}_{w_2})$. Specifically, the polynomial c_i is expressed as*

$$c_i = \sum_{j \in [w_2]} \eta_{i,j} \hat{s}_j + \sum_{j \in \overline{[w_1]}, k \in [n]} \eta_{i,j,k} s_j b_k,$$

for all $i \in [w_3]$, where $\eta_{i,j}, \eta_{i,j,k} \in \mathbb{Z}_p$.

– $\mathrm{Pair}(x, y, p) \to (\mathbf{E}, \overline{\mathbf{E}})$*: On input p, x, and y, this algorithm outputs two matrices $\mathbf{E}$ and $\overline{\mathbf{E}}$ of sizes $(w_1 + 1) \times m_3$ and $w_3 \times m_1$, respectively.*

A PES is correct for every $\kappa = (p, \mathrm{par})$, $x \in \mathcal{X}_\kappa$ and $y \in \mathcal{Y}_\kappa$ such that $P_\kappa(x, y) = 1$, it holds that

$$\mathbf{s}\mathbf{E}\mathbf{k}^\intercal + \mathbf{c}\overline{\mathbf{E}}\mathbf{r}^\intercal = \alpha s.$$

The symbolic property is a powerful security notion for PESs that applies to a large class of predicate encryption schemes.

Definition 13 (Symbolic property ($\mathsf{Sym\text{-}Prop}^+$) [4,10]). *A pair encoding scheme* Γ = (Param, EncKey, EncCt, Pair) *for a predicate family* $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ *satisfies the* (d_1, d_2)*-selective symbolic property for positive integers* d_1 *and* d_2 *if there exist deterministic polynomial-time algorithms* EncB, EncS, *and* EncR *such that for all* $\kappa = (p, \mathrm{par})$, $x \in \mathcal{X}_\kappa$ *and* $y \in \mathcal{Y}_\kappa$ *with* $P_\kappa(x,y) = 0$, *we have*

- $\mathrm{EncB}(x) \to \mathbf{B}_1, ..., \mathbf{B}_n \in \mathbb{Z}_p^{d_1 \times d_2}$;
- $\mathrm{EncR}(x,y) \to \mathbf{r}_1, ..., \mathbf{r}_{m_1} \in \mathbb{Z}_p^{d_1}, \mathbf{a}, \hat{\mathbf{r}}_1, ..., \hat{\mathbf{r}}_{m_2} \in \mathbb{Z}_p^{d_2}$;
- $\mathrm{EncS}(x) \to \mathbf{s}_0, ..., \mathbf{s}_{w_1} \in \mathbb{Z}_p^{d_2}, \hat{\mathbf{s}}_1, ..., \hat{\mathbf{s}}_{w_2} \in \mathbb{Z}_p^{d_1}$;

such that $\langle \mathbf{s}_0, \mathbf{a} \rangle \neq 0$ *and* $\mathbf{a} = (1, \mathbf{0}^{d_2 - 1})$, *and if we substitute*

$$\hat{s}_{i'} : \hat{\mathbf{s}}_{i'}^{\intercal} \qquad s_i b_j : \mathbf{B}_j \mathbf{s}_i^{\intercal} \qquad \alpha : \mathbf{a} \qquad \hat{r}_{k'} : \hat{\mathbf{r}}_{k'} \qquad r_k b_j : \mathbf{r}_k \mathbf{B}_j,$$

for $i \in [w_1], i' \in [w_2], j \in [n], k \in [m_1], k' \in [m_2]$ *in all the polynomials of* $\mathbf{k}$ *and* $\mathbf{c}$ *(output by* EncKey *and* EncCt, *respectively), they evaluate to* $\mathbf{0}$.

Similarly, a pair encoding scheme satisfies the (d_1, d_2)*-co-selective symbolic security property if there exist* EncB, EncR, EncS *that satisfy the above properties but where* EncB *and* EncR *only take* y *as input, and* EncS *takes* x *and* y *as input.*

A scheme satisfies the (d_1, d_2)*-symbolic property if it satisfies the* (d_1', d_2')*-selective and* (d_1'', d_2'')*-co-selective properties for* $d_1', d_1'' \leq d_1$ *and* $d_2', d_2'' \leq d_2$.

Agrawal and Chase [4] prove that any PES satisfying the (d_1, d_2)-symbolic property can be transformed in a fully secure predicate encryption scheme. The resulting schemes are proven secure under a q-type assumption, which is a security assumption that becomes stronger as some parameter q grows.

In some works [8,9], the information-theoretic security notion of *perfectly master-key hiding* is used to achieve security under non-parametrized assumptions such as the symmetric external Diffie-Hellman (SXDH).

Definition 14 (Perfectly master-key hiding (PMH) [9]). *A pair encoding scheme* Γ = (Param, EncKey, EncCt, Pair) *for a predicate family* $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ *is perfectly master-key hiding if, for all* $\kappa = (p, \mathrm{par})$, $x \in \mathcal{X}_\kappa$ *and* $y \in \mathcal{Y}_\kappa$ *with* $P_\kappa(x,y) = 0$, *we have that the following distributions are identical:*

$$\{\mathbf{k}(\alpha, \mathbf{r}, \hat{\mathbf{r}}, \mathbf{b}, y), \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, x)\} \text{ and } \{\mathbf{k}(0, \mathbf{r}, \hat{\mathbf{r}}, \mathbf{b}, y), \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, x)\},$$

where all variables $\alpha, \mathbf{r}, \hat{\mathbf{r}}, \mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}$ *are taken uniformly at random from* $\mathbb{Z}_p$.

Attrapadung [9] proves that any pair encoding scheme that is perfectly master-key hiding can be converted to a fully secure predicate encryption scheme. The resulting scheme is then secure under a static assumption such as SXDH.

C.2 Transformation for pair encodings

We define the transformation in Section 4 for pair encodings as follows.

Definition 15 (PredEx-Trans **for PES).** *Let $\mathit{\Gamma}$ be a PES for predicate P. Then, we construct a PES for* $\mathrm{CCA}[P]$ *as follows:*

- $\mathrm{Param}'(\mathrm{par}) = \mathrm{Param}(\mathrm{par}) + 2$. *The common variables are* $\mathbf{b}' = (\mathbf{b}, b'_0, b'_1)$, *where* $\mathbf{b}$ *are the common variables of* $\mathit{\Gamma}$.
- $\mathrm{EncKey}'((y, y'), p)$. *Let $y \in \mathcal{Y}_\kappa$ and $y' \in \mathbb{Z}_p \cup \{*\}$, and generate $\alpha_1 \in_R \mathbb{Z}_p$, and set $\alpha_2 = \alpha - \alpha_1$. Then, compute* $\mathbf{k}^{(1)}(\alpha, \mathbf{r}^{(1)}, \hat{\mathbf{r}}^{(1)}, \mathbf{b}, y) \leftarrow \mathrm{EncKey}(y, p)$, *and replace each occurrence of α by α_1, yielding* $\mathbf{k}^{(2)}(\alpha_1, \mathbf{r}^{(1)}, \hat{\mathbf{r}}^{(1)}, \mathbf{b}, y)$. *Additionally, compute*

$$\mathbf{k}^{(3)}(\alpha_2, \mathbf{r}^{(1)}, \hat{\mathbf{r}}^{(1)}, \mathbf{b}', y') = \begin{cases} (\alpha_2 + r_1(b'_0 + y'b'_1)), \textit{for } y' \in \mathbb{Z}_p \\ (\alpha_2 + r_1 b'_0, r_1 b'_1)), \quad \textit{for } y' = *. \end{cases}$$

  Output $\mathbf{k}(\alpha, \mathbf{r}, \hat{\mathbf{r}}, \mathbf{b}', (y, y')) = (\mathbf{k}^{(2)}, \mathbf{k}^{(3)})$, *where* $\mathbf{r} = \mathbf{r}^{(1)}$, *and* $\hat{\mathbf{r}} = (\alpha_1, \hat{\mathbf{r}}^{(1)})$.
- $\mathrm{EncCt}'((x, x'), p)$. *Let $x \in \mathcal{X}_\kappa$ and $x' \in \mathbb{Z}_p$. Compute* $\mathbf{c}' = \mathbf{c}'(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, x) \leftarrow \mathrm{EncCt}(x, p)$. *Output* $\mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}', (x, x')) \leftarrow (\mathbf{c}', s(b'_0 + x'b'_1))$.

Pair/Correctness. Let $(x, x') \in \mathcal{X}'_\kappa$ and $(y, y') \in \mathcal{Y}'_\kappa$ be such that $P'((x, x'), (y, y')) = 1$. In particular, we have $P(x, y) = 1$ and either $y' = *$ or $x' = y'$. Let $(\mathbf{E}', \bar{\mathbf{E}}') \leftarrow \mathrm{Pair}(x, y, p)$, such that $\mathbf{s}\mathbf{E}'(\mathbf{k}'_1)^\intercal + \mathbf{c}'\bar{\mathbf{E}}'\mathbf{r}^\intercal = \alpha_1 s$. If $y' = *$, we recover

$$\alpha_2 s = \mathbf{s} \begin{pmatrix} 1 & x' \\ \mathbf{0}^\intercal & \mathbf{0}^\intercal \end{pmatrix} \mathbf{k}_2^\intercal + (s(b'_0 + x'b'_1)) \begin{pmatrix} 1 \; \mathbf{0} \end{pmatrix} \mathbf{r}^\intercal.$$

If $y' \in \mathbb{Z}_p$, we recover

$$\alpha_2 s = \mathbf{s} \begin{pmatrix} 1 \\ \mathbf{0}^\intercal \end{pmatrix} \mathbf{k}_2^\intercal + (s(b'_0 + x'b'_1)) \begin{pmatrix} 1 \; \mathbf{0} \end{pmatrix} \mathbf{r}^\intercal.$$

Finally, we recover $\alpha s = \alpha_1 s + \alpha_2 s$. Note that the output of $\mathrm{Pair}'((x, x'), (y, y'))$ is $(\mathbf{E}, \bar{\mathbf{E}})$, where

$$\mathbf{E} = \begin{cases} \left(\mathbf{E}', \begin{pmatrix} 1 & x' \\ \mathbf{0}^\intercal & \mathbf{0}^\intercal \end{pmatrix}\right) & \text{for } y' = *, \\ \left(\mathbf{E}', \begin{pmatrix} 1 \\ \mathbf{0}^\intercal \end{pmatrix}\right) & \text{for } y' \in \mathbb{Z}_p, \end{cases} \qquad \bar{\mathbf{E}} = \begin{pmatrix} \bar{\mathbf{E}}' \\ (1 \; \mathbf{0}) \end{pmatrix}.$$

C.3 The PES-transformation preserves symbolic security

Theorem 2. *Suppose that $\mathit{\Gamma}$ satisfies (d_1, d_2)-$\mathsf{Sym\text{-}Prop}^+$. Then, $\mathit{\Gamma}' = $ PredEx-Trans($\mathit{\Gamma}$) for $\mathrm{CCA}[P]$ satisfies $(d_1 + 1, 2d_2)$-$\mathsf{Sym\text{-}Prop}^+$.*

Proof. We show that the PES satisfies both the selective and co-selective symbolic properties. We define the partial predicate $\bar{P}_\kappa$ such that $\bar{P}_\kappa(x', y') = 1$ if and only if $x' = y'$ or $y' = *$. Suppose that $(x, x') \in \mathcal{X}'_\kappa$ and $(y, y') \in \mathcal{Y}'_\kappa$ are such that $P'_\kappa((x, x'), (y, y')) = 0$. This means that $P_\kappa(x, y) = 0$ or $x' \neq y'$ (with $y' \in \mathbb{Z}_p$) holds (or both). (Note that, if $y' = *$, then we necessarily have $P_\kappa(x, y) = 0$.)

In particular, EncB, EncR, and EncS output matrix/vector substitutions for the variables α, $\mathbf{b}$, $\mathbf{r}, \hat{\mathbf{r}}$, $\mathbf{s} = (s, s_1, ...)$ and $\hat{\mathbf{s}}$, i.e., $\mathbf{a}, \mathbf{B}^{(1)}, \mathbf{r}^{(1)}, \hat{\mathbf{r}}^{(1)}, \mathbf{s}^{(1)}$, and $\hat{\mathbf{s}}^{(1)}$ (which are vectors of matrices/vectors). For these substitutions, it holds that, if $P_\kappa(x, y) = 0$, then the polynomials in the encodings evaluate to $\mathbf{0}$.

- **The selective symbolic property:** First, we show that the selective symbolic property holds. We use the substitutions of Γ for the selective symbolic property to substitute the variables and polynomials of Γ' as follows:

  $$b_i \ : \ \begin{pmatrix} \mathbf{B}_i^{(1)} \\ \mathbf{0} \end{pmatrix}, \qquad b'_0 \ : \ -x'\mathbf{1}_{d_1+1,1}, \qquad b'_1 \ : \ \mathbf{1}_{d_1+1,1},$$

  $$r_1 \ : \ \left(\beta \mathbf{r}_1^{(1)}, \frac{\beta'(1-\beta)}{x'-y'}\right) \text{ if } y' \in \mathbb{Z}_p, \qquad r_1 \ : \ \left(\beta \mathbf{r}_1^{(1)}, 0\right) \text{ if } y = *,$$

  $$r_{i'} \ : \ (\beta \mathbf{r}_{i'}^{(1)}, 0), \qquad \hat{r}_{i^{(2)}} \ : \ \beta \hat{\mathbf{r}}_{i^{(2)}}^{(1)},$$

  $$\alpha \ : \ \mathbf{a}, \qquad \alpha_1 \ : \ (\beta, \mathbf{0})$$

  $$s \ : \ \mathbf{s}_0^{(1)}, \qquad s_j \ : \ \mathbf{s}_j^{(1)}, \qquad \hat{s}_{j'} \ : \ (\mathbf{s}_{j'}^{(1)}, 0),$$

  for all $i \in [n], i' \in [2, m_1], i^{(2)} \in [m_2], j \in [0, w_1], j' \in [w_2]$, where $\beta = 1 - P_\kappa(x, y)$ and $\beta' = 1 - \bar{P}_\kappa(x', y')$. Note that $\frac{\beta'(1-\beta)}{x'-y'}$ is well-defined, because if $y' = x'$, then $\beta' = 0$. Note that we indeed have $\mathbf{a} \cdot \mathbf{s_0}^{(1)} \neq 0$, because Γ satisfies Sym-Prop$^+$.

  We show that, for these substitutions, the polynomials evaluate to $\mathbf{0}$. We have $\mathbf{k} = (\mathbf{k}^{(2)}, \mathbf{k}^{(3)})$ and $\mathbf{c} = (\mathbf{c}', s(b'_0 + x'b'_1))$, where the polynomials in $\mathbf{c}'$ and $\mathbf{k}^{(2)}$ in which α_1 does not occur evaluate to $\mathbf{0}$ due to the selective symbolic property of Γ. The polynomials k' in which α_1 does occur can be written as $k'(\alpha_1) = \delta'\alpha_1 + k''$, where $\delta' \in \mathbb{Z}_p$ and k'' is a polynomial in which α_1 does not occur. If $P_\kappa(x, y) = 1$, then $\beta = 0$ and $\bar{P}_\kappa(x', y') = 0$, and thus, $\mathbf{r}$ and $\hat{\mathbf{r}}$ are all-zero, except possibly the last entry of r_1, which may be $\frac{1}{x'-y'}$. Since the only common variables that occur in k'' are b_i, which are substituted by matrices in which the last rows are all-zero, all combinations $r_i b_j$ evaluate to $\mathbf{0}$. Furthermore, $\alpha_1 = \mathbf{0}$, and therefore $k'(\alpha_1) = \mathbf{0}$. On the other hand, if $P_\kappa(x, y) = 0$, then $\beta = 1$, and all combinations of $r_i b_j$ are substituted as in Γ itself: $r_i b_j = (\mathbf{r}_i^{(1)}, r_{d_1+1}) \begin{pmatrix} \mathbf{B}_j^{(1)} \\ \mathbf{0} \end{pmatrix} = \mathbf{r}_i^{(1)} \mathbf{B}_j^{(1)}$. And, because in this case, the substitutions for α and α_1 are equal, we have $k'(\alpha_1) = k'(\alpha) = \mathbf{0}$.

  For the "new" polynomials in $\mathbf{k}^{(3)}$ and $s(b'_0 + x'b'_1)$, we may need to consider whether $y' \in \mathbb{Z}_p$ or $y' = *$. In general, we have $s(b'_0 + x'b'_1) = \mathbf{s}_1^{(1)}(-x'\mathbf{1}_{d_1+1,1} + x'\mathbf{E}\mathbf{1}_{d_1+1,1}) = \mathbf{0}$. For $\mathbf{k}^{(3)}$, and $y' \in \mathbb{Z}_p$, we have:

  $$\begin{aligned} &\alpha_2 + r_1(b'_0 + y'b'_1) \\ &= (\mathbf{a}, 0) - (\beta, \mathbf{0}) + \left(\beta \mathbf{r}_1^{(1)}, \frac{\beta'(1-\beta)}{x'-y'}\right)((-x'\mathbf{1}_{d_1+1,1} + y'\mathbf{1}_{d_1+1,1})) \\ &= (1-\beta, \mathbf{0}) + \frac{\beta'(1-\beta)}{x'-y'}(-x' + y', \mathbf{0}). \end{aligned}$$

If $\bar{P}_\kappa(x', y') = 1$, then $P_\kappa(x, y) = 0$, and thus $\beta = 1$. Also, $\beta' = 0$, and therefore we have $\alpha_2 + r_1(b'_0 + y'b'_1) = \mathbf{0}$. Otherwise, $\bar{P}_\kappa(x', y') = 0$, and thus $\beta' = 1$. Then, we have $\alpha_2 + r_1(b'_0 + y'b'_1) = (1-\beta, \mathbf{0}) - (1-\beta, \mathbf{0}) = \mathbf{0}$. For $\mathbf{k}^{(3)}$ and $y = *$, we necessarily have $P_\kappa(x, y) = 0$ and thus, $\beta = 1$. Then,

$$\alpha_2 + r_1 b'_0 = (\mathbf{a}, 0) - (\beta, \mathbf{0}) + \left(\beta \mathbf{r}_1^{(1)}, 0\right) - x'\mathbf{1}_{d_1+1,1} = \mathbf{0},$$

and $r_1 b'_1 = \left(\beta \mathbf{r}_1^{(1)}, 0\right) \mathbf{1}_{d_1+1,1} = \mathbf{0}$.

- **The co-selective property:** We also show that the co-selective property holds. We use the substitutions of Γ for the co-selective symbolic property to substitute the variables and polynomials of Γ' as follows:

$$b_i \;:\; \begin{pmatrix} \mathbf{0}^{d_1 \times d_2} & \mathbf{B}_i^{(1)} \\ \mathbf{0}^{1 \times d_2} & \mathbf{0}^{1 \times d_2} \end{pmatrix},$$

$$b'_0 \;:\; \begin{pmatrix} 0 & 0 & \mathbf{0}^{d_2-1} & 0 & 0 & \mathbf{0}^{d_2-1} \\ -1 & -y' & \mathbf{0}^{d_2-2} & 1 & y' & \mathbf{0}^{d_2-2} \end{pmatrix}, b'_1 \;:\; \begin{pmatrix} 0 & 0 & \mathbf{0}^{d_2-1} & 0 & 0 & \mathbf{0}^{d_2-1} \\ 0 & 1 & \mathbf{0}^{d_2-2} & 0 & -1 & \mathbf{0}^{d_2-2} \end{pmatrix}, \text{ if } y' \in \mathbb{Z}_p$$

$$b'_0 \;:\; \mathbf{1}_{d_1+1,d_2+1}^{(d_1+1)\times 2d_2} - \mathbf{1}_{d_1+1,1}^{(d_1+1)\times 2d_2}, \text{ and } b'_1 \;:\; \mathbf{0}^{(d_1+1)\times 2d_2}, \text{ if } y' = *$$

$$r_1 \;:\; (\mathbf{r}_1^{(1)}, 1), \qquad r_{i'} \;:\; (\mathbf{r}_{i'}^{(1)}, 0), \qquad \hat{r}_{i^{(2)}} \;:\; \begin{pmatrix} \mathbf{0}^{d_2 \times 1} \\ \hat{\mathbf{r}}_{i^{(2)}}^{(1)} \end{pmatrix},$$

$$\alpha \;:\; \mathbf{a}^\intercal, \qquad \alpha_1 \;:\; \mathbf{1}_{d_2+1}^{2d_2}$$

$$s \;:\; \beta \begin{pmatrix} \mathbf{s}_0^{(1)} \\ \mathbf{s}_0^{(1)} \end{pmatrix} + \beta' \begin{pmatrix} 1 \\ \frac{1}{x'-y'} \\ \mathbf{0}^{2d_2-2} \end{pmatrix} \qquad s_j \;:\; \begin{pmatrix} \mathbf{s}_j^{(1)} \\ \mathbf{0}^{d_2} \end{pmatrix}, \qquad \hat{s}_{j'} \;:\; (\mathbf{s}_{j'}^{(1)}, 0),$$

for all $i \in [n], i' \in [2, m_1], i^{(2)} \in [m_2], j \in [0, w_1], j' \in [w_2]$, where $\beta = 1 - P_\kappa(x, y)$, and $\beta' = 1 - \bar{P}_\kappa(x', y')$. Here, we assume that $d_2 \geq 2$. Note that $\alpha s \neq 0$, because the first entry of s is non-zero, which holds because not both β and β' can be 0 (which would hold only if $P'_\kappa((x, x'), (y', y')) = 1$). We show that, for these substitutions, the polynomials evaluate to $\mathbf{0}$. Like in the selective case, for the polynomials in $\mathbf{c}'$ and $\mathbf{k}^{(2)}$, in which α_1 does not occur, it follows readily that the polynomials evaluate to $\mathbf{0}$. Similarly, for the polynomials k' in $\mathbf{k}^{(2)}$ in which α_1 does occur, we can write these polynomials as $k'(\alpha_1) = \delta'\alpha_1 + k''$, where $k'' = \sum_{j \in [m_2]} \delta_{i,j} \hat{r}_j + \sum_{j \in [m_1], k \in [n]} \delta_{i,j,k} r_j b_k$. For the original substitutions in Γ, we have

$$k'' = \delta'\mathbf{a} + \sum_{j \in [m_2]} \delta''_j \hat{\mathbf{r}}_j^{(1)} + \sum_{j \in [m_1], k \in [n]} \delta''_{j,k} \mathbf{r}_j^{(1)} \mathbf{B}_k^{(1)} = \mathbf{0}$$

For the "new" substitutions, we have

$$\begin{aligned} k'' = \delta' \mathbf{1}_{d_2+1}^{2d_2} &+ \sum_{j \in [m_2]} \delta_{i,j} \begin{pmatrix} \mathbf{0}^{d_2 \times 1} \\ \hat{\mathbf{r}}_{i^{(2)}}^{(1)} \end{pmatrix} \\ &+ \sum_{j \in [m_1], k \in [n]} \left(\delta_{i,j,k} (\mathbf{r}_j^{(1)}, (\mathbf{r}_1)_1) \begin{pmatrix} \mathbf{0}^{d_1 \times d_2} & \mathbf{B}_i^{(1)} \\ \mathbf{0}^{1 \times d_2} & \mathbf{0}^{1 \times d_2} \end{pmatrix} \right)^\intercal \end{aligned}$$

$$= (\mathbf{0}^{d_2}, \delta', \mathbf{0}^{d_2-1})^\intercal + \sum_{j\in[m_2]} \delta_{i,j} \begin{pmatrix} \mathbf{0}^{d_2\times 1} \\ \hat{\mathbf{r}}^{(1)}_{i^{(2)}} \end{pmatrix} + \sum_{j\in[m_1],k\in[n]} \delta_{i,j,k}(\mathbf{0}^{d_2}, \mathbf{r}_j^{(1)}\mathbf{B}_k^{(1)})^\intercal$$

$$= \left(\mathbf{0}^{d_2}, \left(\delta'\mathbf{a} + \sum_{j\in[m_2]} \delta_{i,j}\hat{\mathbf{r}}_j^{(1)} + \sum_{j\in[m_1],k\in[n]} \delta_{i,j,k}\mathbf{r}_j^{(1)}\mathbf{B}_k^{(1)}\right)\right) = \mathbf{0}.$$

Now, we show for the "new" polynomials $\mathbf{k}^{(3)}$ and $s(b_0' + x'b_1')$ evaluate to $\mathbf{0}$. If $y' \in \mathbb{Z}_p$, then we have

$$\begin{aligned} s(b_0' + x'b_1') &= \left(\beta \begin{pmatrix} \mathbf{s}_0^{(1)} \\ \mathbf{s}_0^{(1)} \end{pmatrix} + \beta' \begin{pmatrix} 1 \\ \frac{1}{x'-y'} \\ \mathbf{0}^{2d_2-2} \end{pmatrix}\right) (b_0' + x'b_1) \\ &= \left(\mathbf{0}^{d_1}, \beta(\mathbf{s}_0^{(1)})_1(-1+1) + (-y' + y' + x' - x')(\mathbf{s}_0^{(1)})_2\right) \\ &+ (\mathbf{0}^{d_1}, \beta'(-1 - \frac{y'}{x'-y'} + \frac{x'}{x'-y'})) \\ &= \mathbf{0}^{d_1+1} + (\mathbf{0}^{d_1}, \beta'(-1 + \frac{x'-y'}{x'-y'})) = \mathbf{0}^{d_1+1}, \end{aligned}$$

because either we have $x' = y'$ and then, $\beta' = 0$, or we have $(-1 + \frac{x'-y'}{x'-y'}) = 0$. If $y' = *$, then we have $\bar{P}_\kappa(x', y') = 1$ and thus, $\beta' = 0$, and

$$\begin{aligned} s(b_0' + x'b_1') &= \beta \begin{pmatrix} \mathbf{s}_0^{(1)} \\ \mathbf{s}_0^{(1)} \end{pmatrix} (\mathbf{1}^{(d_1+1)\times 2d_2}_{d_1+1,d_2+1} - \mathbf{1}^{(d_1+1)\times 2d_2}_{d_1+1,1} + x'\mathbf{0}^{(d_1+1)\times 2d_2}) \\ &= (\mathbf{0}^{d_1}, \beta(\mathbf{s}_0^{(1)})_1 - \beta(\mathbf{s}_0^{(1)})_1) = \mathbf{0}^{d_1+1}. \end{aligned}$$

For $\mathbf{k}^{(3)}$ and $y' \in \mathbb{Z}_p$, we have

$$\begin{aligned} \alpha_2 + r_1(b_0' + y'b_1') &= \mathbf{1}_1^{2d_2} - \mathbf{1}_{d_2+1}^{2d_2} + (\mathbf{r}_1^{(1)}, 1)(b_0' + y'b_1') \\ &= \mathbf{1}_1^{2d_2} - \mathbf{1}_{d_2+1}^{2d_2} + (-1, -y' + y', \mathbf{0}^{d_2-2}, 1, y' - y', \mathbf{0}^{d_2-2}) \\ &= \mathbf{1}_1^{2d_2} - \mathbf{1}_{d_2+1}^{2d_2} - \mathbf{1}_1^{2d_2} + \mathbf{1}_{d_2+1}^{2d_2} = \mathbf{0}^{2d_2}. \end{aligned}$$

For $\mathbf{k}^{(3)}$ and $y' = *$, we have

$$\begin{aligned} \alpha_2 + r_1b_0' &= \mathbf{1}_1^{2d_2} - \mathbf{1}_{d_2+1}^{2d_2} + (\mathbf{r}_1^{(1)}, 1)\left(\mathbf{1}^{(d_1+1)\times 2d_2}_{d_1+1,d_2+1} - \mathbf{1}^{(d_1+1)\times 2d_2}_{d_1+1,1}\right) \\ &= \mathbf{1}_1^{2d_2} - \mathbf{1}_{d_2+1}^{2d_2} + \mathbf{1}_{d_2+1}^{2d_2} - \mathbf{1}_1^{2d_2} = \mathbf{0}^{2d_2}, \end{aligned}$$

and $r_1b_1' = \mathbf{0}^{2d_2}$.

Thus, Sym-Prop$^+$ holds for Γ'. □

C.4 The transformation preserves perfectly master-key hiding

Theorem 3. *Suppose Γ is perfectly master-key hiding. Then, Γ' =* PredExTrans(Γ) *for* CCA[P] *is also perfectly master-key hiding.*

Proof. Let $(x,x') \in \mathcal{X}'_\kappa$ and $(y,y') \in \mathcal{Y}'_\kappa$ be such that $P'_\kappa((x,x'),(y,y')) = 0$. First, we show that, if $x' \neq y'$ and $y' \neq *$, we have that α_2 is perfectly hidden, i.e., the distributions

$$\{\alpha_2 + r_1(b'_0 + y'b'_1), \hat{\mathbf{r}}^{(1)}, s(b'_0 + x'b'_1)\} \text{ and } \{r_1(b'_0 + y'b'_1), s(b'_0 + x'b'_1)\}$$

are equal. This follows from the fact that, if $x' \neq y'$, then $b'_0 + y'b'_1$ and $b'_0 + x'b'_1$ are pairwise independent [26]. Furthermore, if $P(x,y) = 0$, then α_1 is perfectly hidden by the assumption that Γ is perfectly master-key hidden.

Suppose that $x' = y'$ or $y' = *$. Then, $\alpha_2 = \alpha - \alpha_1$ is not hidden. To ensure that α is hidden, we sample some random $\bar{\alpha} \in_R \mathbb{Z}_p$, which we subtract from α and α_1, i.e., replace α with $\alpha' \leftarrow \alpha - \bar{\alpha}$ and α_1 with $\alpha'_1 \leftarrow \alpha_1 - \bar{\alpha}$. Note that we still have $\alpha_2 = \alpha' - \alpha'_1$, and thus, this does not change the encodings for x' and y'. Because $P(x,y) = 0$, we can switch out α_1 for α'_1 in $\mathbf{k}^{(2)}$, because α_1 is hidden. Therefore, α is hidden.

Suppose now that $P(x,y) = 1$. In this case, α_1 is not hidden. Then, we similarly hide α by subtracting randomly generated $\bar{\alpha} \in_R \mathbb{Z}_p$, i.e., replace α by $\alpha' \leftarrow \alpha - \bar{\alpha}$. Because $x' \neq y'$ and $y' \neq *$, we have that $\alpha'_2 = \alpha' - \alpha_1 = \alpha - \bar{\alpha} - \alpha_1 \neq \alpha_2$ is hidden. Thus, α is hidden. □

C.5 Transformation for predicate encodings

Because our transformation for pair encodings re-uses the randomness $\mathbf{r}$ and $\mathbf{s}$ in the extension, it can also be applied to predicate encodings [48,26]. In particular, if $\mathbf{r}$ and $\mathbf{s}$ are of length 1, then our transformation does not increase the number of key and ciphertext variables, and thus, the transformation yields a (new) predicate encoding. By Theorem 3, the predicate encoding satisfies the α-privacy property, which is similar to the perfectly master-key hiding property [7]. In fact, the encoding for equality given in [26] is the same as our "all-or-one-identity" IBE for $y' \in \mathbb{Z}_p$. It can be simply adjusted to also include the encodings for $y' = *$. We give a proof in Appendix D.2.

D Predicate encodings

D.1 Definition of predicate encodings

Definition 16 (Predicate encodings [48,26]). *A $\mathbb{Z}_p$-bilinear predicate encoding scheme for a predicate family $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$, indexed by $\kappa = (p, \mathrm{par})$, where* par *specifies some parameters, is given by five deterministic polynomial-time algorithms* $(\mathrm{sE}, \mathrm{rE}, \mathrm{kE}, \mathrm{sD}, \mathrm{rD})$*, such that, for all κ, the following properties are satisfied:*

- ***Linearity:*** *For all $s(x,y) \in \mathcal{X}_\kappa \times \mathcal{Y}_\kappa$, the functions $\mathrm{sE}(x,\cdot)$, $\mathrm{rE}(y,\cdot)$, $\mathrm{kE}(y,\cdot)$, $\mathrm{sD}(x,y,\cdot)$ and $\mathrm{rD}(x,y,\cdot)$ are $\mathbb{Z}_p$-linear.*
- ***Restricted α-reconstruction:*** *For all $(x,y) \in \mathcal{X}_\kappa \times \mathcal{Y}_\kappa$ such that $P_\kappa(x,y) = 1$, and for all $\mathbf{w} \in \mathbb{Z}_p^n$:*

$$\mathrm{sD}(x,y,\mathrm{sE}(x,\mathbf{w})) = \mathrm{rD}(x,y,\mathrm{rE}(y,\mathbf{w})) \textit{ and } \mathrm{rD}(x,y,\mathrm{kE}(y,\alpha)) = \alpha.$$

- α-***privacy:*** *For all* $(x, y) \in \mathcal{X}_\kappa \times \mathcal{Y}_\kappa$ *such that* $P_\kappa(x, y) = 0$*, and for all* $\alpha \in \mathbb{Z}_p$*, the following distributions are identically distributed:*

  $$\{x, y, \alpha, \mathrm{sE}(x, \mathbf{w}), \mathrm{kE}(y, \alpha) + \mathrm{rE}(y, \mathbf{w})\} \text{ and } \{x, y, \alpha, \mathrm{sE}(x, \mathbf{w}), \mathrm{rE}(y, \mathbf{w})\},$$

  where $\mathbf{w} \in_R \mathbb{Z}_p^n$*.*

D.2 Transformation for predicate encodings

We define the transformation in Section 4 for predicate encodings as follows.

Definition 17 (PredEx-Trans **for predicate encodings).** *Let* $\Gamma = (\mathrm{sE}, \mathrm{rE}, \mathrm{kE}, \mathrm{sD}, \mathrm{rD})$ *be a predicate encoding for predicate* P*. Then, we construct a predicate encoding for the extended predicate* $\mathrm{CCA}[P]$ *as follows:*

- *The length of* $\mathbf{w}'$ *is increased by three compared to* $\mathbf{w}$ *of* Γ*:* $\mathbf{w}' = (\mathbf{w}, w_0', w_1', u)$*.*
- $\mathrm{sE}'((x, x'), \mathbf{w}') = (\mathbf{c} = \mathrm{sE}(x, \mathbf{w}), c' = w_0' + x'w_1')$*.*
- $\mathrm{sD}'((x, x'), (y, y'), (\mathbf{c}, c')) = \mathrm{sD}(x, y, \mathbf{c}) + c'$*.*
- $\mathrm{rE}'((y, y'), \mathbf{w}') = (\mathbf{k} = \mathrm{rE}(y, \mathbf{w}) + \mathrm{kE}(y, u), \mathbf{k}' = \mathrm{rE}''(y', \mathbf{w}'))$*, where*

  $$\mathrm{rE}''(y', \mathbf{w}') = \begin{cases} (-u + w_0' + y'w_1'), & \textit{for } y' \in \mathbb{Z}_p \\ (-u + w_0', w_1'), & \textit{for } y' = *. \end{cases}$$

- $\mathrm{kE}'((y, y'), \alpha) = (\mathbf{0}^{|\mathrm{kE}(y,\alpha)|}, \mathrm{kE}''(y', \alpha))$*, where*

  $$\mathrm{kE}''(y', \alpha)) = \begin{cases} (\alpha), & \textit{for } y' \in \mathbb{Z}_p \\ (\alpha, 0), & \textit{for } y' = *. \end{cases}$$

- $\mathrm{rD}'((x, x'), (y, y'), (\mathbf{k}, \mathbf{k}')) = \mathrm{rD}(x, y, \mathbf{k}) + \mathrm{rD}''(x', y', \mathbf{k}')$*, where*

  $$\mathrm{rD}''(x', y', \mathbf{k}') = \begin{cases} 1, & \textit{for } y' \in \mathbb{Z}_p \\ \mathbf{k}_1' + x'\mathbf{k}_2', & \textit{for } y' = *. \end{cases}$$

- ***Restricted*** α***-reconstruction:*** *We have*

  $$\begin{aligned} \mathrm{sD}'((x, x'), (y, y), \mathrm{sE}'((x, x'), \mathbf{w}')) &= \mathrm{sD}'((x, x'), (y, y), (\mathrm{sE}(x, \mathbf{w})), w_0' + x'w_1')) \\ &= \mathrm{sD}(x, y, \mathrm{sE}(x, \mathbf{w}))) + w_0' + x'w_1', \end{aligned}$$

  which is equal to

  $$\begin{aligned} \mathrm{rD}'((x, x'), (y, y), \mathrm{rE}'((y, y'), \mathbf{w}')) &= \mathrm{rD}'((x, x'), (y, y'), (\mathrm{rE}(y, \mathbf{w}) + \mathrm{kE}(y, u), \mathbf{k}')) \\ &= \mathrm{rD}(x, y, \mathrm{rE}(y, \mathbf{w}) + \mathrm{kE}(y, u)) + \mathrm{rD}''(x', y', \mathbf{k}') \\ &= \mathrm{rD}(x, y, \mathrm{rE}(y, \mathbf{w})) + \mathrm{rD}(x, y, \mathrm{kE}(y, u)) \\ &\quad + \begin{cases} -u + w_0' + y'w_1', & \textit{for } y' \in \mathbb{Z}_p \\ -u + w_0' + x'w_1', & \textit{for } y' = *. \end{cases} \\ &= \mathrm{sD}(x, y, \mathrm{sE}(y, \mathbf{w})) + u - u + w_0' + x'w_1'. \end{aligned}$$

- α-***privacy:*** *The argument is similar as in the case of information-theoretic pair encoding (Section 3).*

E The transformations preserve attribute-hiding

We show that our transformations preserve the attribute-hiding property [21,36] (which includes anonymous IBE [23] as a special case). In anonymous/attribute-hiding PE, the attribute x of the ciphertext is hidden, and cannot be inferred from the ciphertext either. Intuitively, the reasoning behind why our transformations preserve this property is simple. Because the extended-predicate functionality is independent of the original predicate functionality and does not reveal any additional information about the original predicate, the transformed scheme is also anonymous or attribute-hiding. More formally, we prove this by reducing the anonymity/attribute-hiding security of the resulting scheme to the original scheme. To this end, we first give a definition of weakly attribute-hiding PE. Then, we show how a scheme is created from a predicate encoding, and what the original and resulting scheme looks like.

E.1 Attribute-hiding PE

Definition 18 (Attribute-hiding and fully CPA-secure PE [26]). *Let* $\Gamma = (\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Encrypt}, \mathrm{Decrypt})$ *be a PE scheme for predicate* P*. We define the security game* $\text{IND-CPA-AH}(\lambda, \mathrm{par})$ *between challenger and attacker as follows:*

- ***Setup phase:*** *The challenger runs* $\mathrm{Setup}(\lambda)$ *to obtain* MPK *and* MSK*, and sends the master public key* MPK *to the attacker.*
- ***First query phase:*** *The attacker queries secret keys for* $y \in \mathcal{Y}_\kappa$*, and obtains* $\mathrm{SK}_y \leftarrow \mathrm{KeyGen}(\mathrm{MSK}, y)$ *in response.*
- ***Challenge phase:*** *The attacker specifies some* $x_0^*, x_1^* \in \mathcal{X}_\kappa$ *such that for all* y *in the first key query phase, we have* $P(x_0^*, y) = P(x_1^*, y) = 0$*, and generates two messages* M_0 *and* M_1 *of equal length in* $\mathcal{M}_\lambda$*, and sends these to the challenger. The challenger flips a coin, i.e.,* $\beta \in_R \{0,1\}$*, encrypts* M_β *under* x_β^**, i.e.,* $\mathrm{CT}_{x_\beta^*} \leftarrow \mathrm{Encrypt}(\mathrm{MPK}, x_\beta^*, M_\beta)$*, and sends the resulting ciphertext* $\mathrm{CT}_{x_\beta^*}$ *to the attacker.*
- ***Second query phase:*** *This phase is identical to the first query phase, with the additional restriction that the attacker can only query* $y \in \mathcal{Y}_\kappa$ *such that* $P(x_0^*, y) = P(x_1^*, y) = 0$*.*
- ***Decision phase:*** *The attacker outputs a guess* β' *for* β*.*

The advantage of the attacker is defined as $\mathsf{Adv}_{\mathrm{PE,IND\text{-}CPA\text{-}AH}} = |\Pr[\beta' = \beta] - \frac{1}{2}|$*. A scheme is fully secure and attribute-hiding if all polynomial-time attackers have at most a negligible advantage in this security game, i.e., we have* $\mathsf{Adv}_{\mathrm{PE,IND\text{-}CPA\text{-}AH}} \leq \mathrm{negl}(\lambda)$*.*

In the selective security model, the attacker commits to the predicate $x^* \in \mathcal{X}_\kappa$ *before the Setup phase.*

E.2 Generic compiler from dual system groups

Chen, Gay and Wee [26] devised a generic compiler that transforms any predicate encoding into a fully secure PE using dual system groups (DSG) [27] from k-Lin.

We will consider their specific instantiation for $k = 1$, i.e., SXDH, which is the most efficient.

Notation. Given $a \in \mathbb{Z}_p$, we use $[a]_1$ to denote g^a, $[a]_2$ to denote h^a and $[a]_T$ to denote $e(g,h)^a$. This extends to vectors and matrices in an obvious way, e.g., $[(a_1, a_2, ...)]_1$ denotes $(g^{a_1}, g^{a_2}, ...)$. We define $e([\mathbf{A}]_1, [\mathbf{B}]_2) = [\mathbf{A}^\intercal \mathbf{B}]_T$. Let $\mathcal{D}_1$ denote the distribution over matrices $\mathbf{A} = \begin{pmatrix} a_1 \\ 1 \end{pmatrix}$, where $a_1 \in_R \mathbb{Z}_p$.

Definition 19 (Generic compiler for DSGs from SXDH [26]). *Let $\Gamma =$ $(\mathrm{sE}, \mathrm{rE}, \mathrm{kE}, \mathrm{sD}, \mathrm{rD})$ be a predicate encoding as in Definition 16.*

- $\mathrm{Setup}(\lambda)$*: On input the security parameter λ, the PKG first generates domain parameters $(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, g, h, e)$. Then, it generates $k_1, k_2 \in \mathbb{Z}_p$, $\mathbf{A}, \mathbf{B} \in \mathcal{D}_1$, and for each entry w_i in vector $\mathbf{w} \in \mathbb{Z}_p^n$, it generates $\mathbf{W} \in_R \mathbb{Z}_p^{2\times 2}$. It sets* $\mathrm{MSK} = (\mathbf{k} = \begin{pmatrix} k_1 \\ k_2 \end{pmatrix}, \mathbf{A}, \mathbf{B}, \mathbf{W})$*, and outputs the master public key*
$$\mathrm{MPK} = (A = e(g,h)^{k_1 a_1 + k_2}, [\mathbf{A}]_1, [\mathbf{B}]_2, \{[\mathbf{W}_i^\intercal \mathbf{A}]_1, [\mathbf{W}_i \mathbf{B}]_2\}_{i \in [n]}).$$
- $\mathrm{KeyGen}(\mathrm{MSK}, y)$*: On input the master secret key* MSK *and some $y \in \mathcal{Y}_\kappa$, the PKG generates $r \in_R \mathbb{Z}_p$, and output*
$$\mathrm{SK}_y = (\mathbf{K} = [\mathbf{B}r]_2, \mathbf{K}' = \mathrm{kE}(y, [\mathbf{k}]_2) \cdot \mathrm{rE}(y, [\mathbf{W}_1 \mathbf{B} r, ..., \mathbf{W}_n \mathbf{B} r]_2))$$
- $\mathrm{Encrypt}(\mathrm{MPK}, x, M)$*: On input the master public key* MPK*, some $x \in \mathcal{X}_\kappa$ and message M, it generates $s \in_R \mathbb{Z}_p$, and outputs*
$$\mathrm{CT}_x = (C = M \cdot A^s, \mathbf{C}' = [\mathbf{A}s]_1, \mathbf{C}'' = \mathrm{sE}(x, [\mathbf{W}_1^\intercal \mathbf{A} s, ..., \mathbf{W}_n^\intercal \mathbf{A} s]_1))$$
- $\mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x)$*: On input the master public key* MPK*, the secret key* SK_y*, and the ciphertext* CT_x*, if $P(x,y) = 1$, then the message can be obtained as*
$$M' = C / \left(e(\mathbf{C}', \mathrm{rD}(x, y, \mathbf{K}')) / e(\mathrm{sD}(x, y, \mathbf{C}''), \mathbf{K}) \right).$$

E.3 The security proof

Proposition 1. *Let $\Gamma = (\mathrm{sE}, \mathrm{rE}, \mathrm{kE}, \mathrm{sD}, \mathrm{rD})$ be a predicate encoding such that the associated PE scheme $\Psi = (\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Encrypt}, \mathrm{Decrypt})$ is attribute-hiding. Then, the PE scheme Ψ' associated with the predicate encoding $\Gamma' =$ $(\mathrm{sE}', \mathrm{rE}', \mathrm{kE}', \mathrm{sD}', \mathrm{rD}')$ that follows with the transformations in Definitions 17 and 9 is also attribute-hiding.*

Proof. Let $\mathcal{A}_{\Psi'}$ denote the attacker that can break the attribute-hiding property of scheme Ψ' with advantage $\mathsf{Adv}'_{\mathrm{PE,IND\text{-}CPA\text{-}AH}}$. We use it to construct an attacker $\mathcal{A}_\Psi$ that can break the attribute-hiding property of scheme Ψ. Let $\mathcal{C}_{\Psi'}$ and $\mathcal{C}_\Psi$ denote the respective challengers of attackers $\mathcal{A}_{\Psi'}$ and $\mathcal{A}_\Psi$.

- **Setup phase:** Challenger $\mathcal{C}_{\Psi}$ runs the setup algorithm, returning

$$\text{MPK} = (A = e(g,h)^{k_1 a_1 + k_2}, [\mathbf{A}]_1, [\mathbf{B}]_2, \{[\mathbf{W}_i^{\intercal}\mathbf{A}]_1, [\mathbf{W}_i\mathbf{B}]_2\}_{i\in[n]})$$

to attacker $\mathcal{A}_{\Psi}$. Challenger then selects target-collision resistant hash TCR and authenticated encryption scheme $\text{SE} = (\text{Enc}_\text{K}, \text{Dec}_\text{K})$, generates $\mathbf{W}_{n+1}, \mathbf{W}_{n+2}, \mathbf{W}_{n+3} \in_R \mathbb{Z}_p^{2\times 2}$, computes $[\mathbf{W}_i^{\intercal}\mathbf{A}]_1$ and $[\mathbf{W}_i\mathbf{B}]_2$ from $[\mathbf{A}]_1$ and $[\mathbf{B}]_2$, i.e.

$$[\mathbf{W}_i^{\intercal}\mathbf{A}]_1 = \begin{pmatrix} [a_1]_1^{w_{11}} \cdot [1]_1^{w_{21}} \\ [a_1]_1^{w_{12}} \cdot [1]_1^{w_{22}} \end{pmatrix} \text{ and } [\mathbf{W}_i\mathbf{B}]_2 = \begin{pmatrix} [b_1]_2^{w_{11}} \cdot [1]_2^{w_{12}} \\ [b_1]_2^{w_{21}} \cdot [1]_2^{w_{22}} \end{pmatrix},$$

and returns to attacker $\mathcal{A}_{\Psi'}$:

$$\text{MPK}' = (A, [\mathbf{A}]_1, [\mathbf{B}]_2, \{[\mathbf{W}_i^{\intercal}\mathbf{A}]_1, [\mathbf{W}_i\mathbf{B}]_2\}_{i\in[n+3]}).$$

- **First query phase:** For each $y \in \mathcal{Y}_\kappa$ for which attacker $\mathcal{A}_{\Psi'}$ queries a secret key, attacker $\mathcal{A}_{\Psi}$ queries challenger $\mathcal{C}_{\Psi}$ for a secret key:

$$\text{SK}_y = (\mathbf{K} = [\mathbf{B}r]_2, \mathbf{K}' = \text{kE}(y, [\mathbf{k}]_2) \cdot \text{rE}(y, [\mathbf{W}_1\mathbf{B}r, ..., \mathbf{W}_n\mathbf{B}r]_2)),$$

which is used to construct

$$\text{SK}'_y = (\mathbf{K} = [\mathbf{B}r]_2, \mathbf{K}'' = \text{kE}'((y,*), [\mathbf{k}]_2) \cdot \text{rE}'((y,*), [\mathbf{W}_1\mathbf{B}r, ..., \mathbf{W}_{n+3}\mathbf{B}r]_2)).$$

In particular, note that $\text{kE}''(*, [\mathbf{k}]_2)$ can be computed trivially from SK_y, and $\text{rE}''(*, [\mathbf{W}_1\mathbf{B}r, ..., \mathbf{W}_{n+3}\mathbf{B}r]_2)$ can be computed by using that

$$[\mathbf{W}_i\mathbf{B}r] = \begin{pmatrix} [b_1 r]_2^{w_{11}} \cdot [r]_2^{w_{12}} \\ [b_1 r]_2^{w_{21}} \cdot [r]_2^{w_{22}} \end{pmatrix}.$$

- **Challenge phase:** At some point, attacker $\mathcal{A}_{\Psi'}$ sends a message $M \in \{0,1\}^*$ and two predicates $x_0^*, x_1^* \in \mathcal{X}_\kappa$. Attacker $\mathcal{A}_{\Psi}$ sends x_0^*, x_1^* and $M' \in_R \mathbb{G}_T$ to $\mathcal{C}_{\Psi}$, who flips a coin $\beta \in_R \{0,1\}$ and returns

$$\text{CT}^*_{x^*_\beta} = (C = M' \cdot A^s, \mathbf{C}' = [\mathbf{A}s]_1, \mathbf{C}'' = \text{sE}(x, [\mathbf{W}_1^{\intercal}\mathbf{A}s, ..., \mathbf{W}_n^{\intercal}\mathbf{A}s]_1)).$$

This is used to construct

$$\bar{\text{C}}\text{T}^*_{x^*_\beta} = (\text{CT}^*_{\text{sym}}, \mathbf{C}', \bar{\mathbf{C}}'' = \text{sE}'((x^*_\beta, x'), [\mathbf{W}_1^{\intercal}\mathbf{A}s, ..., \mathbf{W}_n^{\intercal}\mathbf{A}s]_1),$$

where $\text{CT}^*_{\text{sym}} \leftarrow \text{Enc}_{C/M'}(M)$, $x' \leftarrow \text{TCR}([a_1 s]_1)$, and $\text{sE}''(x', [\mathbf{W}_1^{\intercal}\mathbf{A}s, ..., \mathbf{W}_n^{\intercal}\mathbf{A}s]_1)$ is generated from $\text{CT}^*_{x^*_\beta}$ in a similar way as in the key generation:

$$[\mathbf{W}_i^{\intercal}\mathbf{A}s]_1 = \begin{pmatrix} [a_1 s]_1^{w_{11}} \cdot [s]_1^{w_{21}} \\ [a_1 s]_1^{w_{12}} \cdot [s]_1^{w_{22}}. \end{pmatrix}$$

- **Second query phase:** This phase is identical to the first query phase.
- **Decision phase:** Attacker $\mathcal{A}_{\Psi'}$ outputs a guess β' for β, which attacker $\mathcal{A}_{\Psi}$ also outputs as its guess.

The advantage $\mathsf{Adv}_{\text{PE,IND-CPA-AH}}$ of attacker $\mathcal{A}_{\Psi}$ is equal to the advantage of attacker $\mathcal{A}_{\Psi'}$: $\mathsf{Adv}_{\text{PE,IND-CPA-AH}} = \mathsf{Adv}'_{\text{PE,IND-CPA-AH}}$. □

F An anonymous IBE scheme

F.1 Identity-based encryption

A special case of predicate encryption is identity-based encryption.

Definition 20 (Identity-based encryption (IBE) [45,18]). *An identity-based encryption scheme consists of four algorithms:*

- $\mathrm{Setup}(\lambda)$*: On input the security parameter* λ*, this probabilistic algorithm, performed by the Private Key Generator (PKG), generates the domain parameters, the master public key* MPK *and the master secret key* MSK*. The master public key and domain parameters are published, while the master secret key is kept secret by the PKG.*
- $\mathrm{KeyGen}(\mathrm{MSK}, \mathsf{ID})$*: On input the master secret key and some identifier* $\mathsf{ID} \in \{0,1\}^*$*, this probabilistic algorithm, performed by the PKG, generates a secret key* $\mathrm{SK}_{\mathsf{ID}}$ *for identifier* ID*.*
- $\mathrm{Encrypt}(\mathrm{MPK}, \mathsf{ID}, M)$*: On input the master public key, identifier* $\mathsf{ID} \in \{0,1\}^*$ *and message* M*, this probabilistic algorithm generates a ciphertext* $\mathrm{CT}_{\mathsf{ID}}$*.*
- $\mathrm{Decrypt}(\mathrm{MPK}, \mathrm{CT}_{\mathsf{ID}}, \mathrm{SK}_{\mathsf{ID}'})$*: On input the ciphertext* $\mathrm{CT}_{\mathsf{ID}}$ *for identifier* ID *and secret key* $\mathrm{SK}_{\mathsf{ID}'}$ *for identifier* ID'*, if* $\mathsf{ID} = \mathsf{ID}'$*, then it returns* M*. Otherwise, it returns an error message* $\perp$*.*

F.2 The CGW anonymous IBE scheme

Definition 21 (CGW-IBE [26]). *The anonymous identity-based encryption scheme proposed by Chen, Gay and Wee is defined as follows.*

- $\mathrm{Setup}(\lambda)$*: On input the security parameter* λ*, the algorithm generates three groups* $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ *of prime order* p *with generators* $g \in \mathbb{G}$ *and* $h \in \mathbb{H}$*, and chooses a pairing* $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$*. It also specifies a collision-resistant hash function* $\mathcal{H}\colon \{0,1\}^* \to \mathbb{Z}_p$*. It then generates random* $k_i, a_i, b_i, w_{0ij}, w_{1ij} \in_R \mathbb{Z}_p$ *for all* $i, j \in \{1,2\}$*. It outputs* $\mathrm{MSK} = (\{k_i, a_i, b_i, w_{0ij}, w_{1ij}\}_{i,j\in\{1,2\}})$ *as the master secret key and publishes the domain parameters* $(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, \mathcal{H})$ *and the master public key as*

$$\mathrm{MPK} = (g, h, A = e(g,h)^{a_1k_1+a_2k_2},$$
$$A_1 = g^{a_1}, A_2 = g^{a_2}, \{W_{l,j} = g^{a_1w_{l1j}+a_2w_{l2j}}\}_{j\in\{1,2\},l\in\{0,1\}}).$$

- $\mathrm{KeyGen}(\mathrm{MSK}, \mathsf{ID})$*: On input identifier* $\mathsf{ID} \in \{0,1\}^*$*, the PKG first hashes* $x = \mathcal{H}(\mathsf{ID})$*, and then generates random integer* $r \in_R \mathbb{Z}_p$ *and computes the secret key as*

$$\mathrm{SK}_{\mathsf{ID}} = (\{K_i = h^{rb_i}, K_i' = h^{-k_i - r(b_1w_{0i1}+b_2w_{0i2}+x(b_1w_{1i1}+b_2w_{1i2}))}\}_{i\in\{1,2\}}).$$

- $\mathrm{Encrypt}(\mathrm{MPK}, \mathsf{ID}, M)$*: Message* $M \in \mathbb{G}_T$ *is encrypted under identifier* ID *by first hashing* $x = \mathcal{H}(\mathsf{ID})$*, then picking random integer* $s \in_R \mathbb{Z}_p$*, and computing the ciphertext as*

$$\mathrm{CT}_{\mathsf{ID}} = \big(C = M \cdot A^s, \{C_i = A_i^s, C_i' = (W_{0i}W_{1i}^x)^s\}_{i\in\{1,2\}}\big).$$

- Decrypt(MPK, $\mathrm{SK}_{\mathsf{ID}'}$, $\mathrm{CT}_{\mathsf{ID}}$)*: Suppose that* $\mathsf{ID} = \mathsf{ID}'$, *then the ciphertext can decrypted by computing*

$$M' = C \cdot e(C_1, K_1') \cdot e(C_2, K_2') \cdot e(C_1', K_1) \cdot e(C_2', K_2).$$

The scheme is correct, i.e., we have

$$\begin{aligned} e(C_1, K_1') \cdot e(C_2, K_2') &= \prod_{i \in \{1,2\}} e(A_i^s, h^{-k_i - r(b_1 w_{0i1} + b_2 w_{0i2} + x(b_1 w_{1i1} + b_2 w_{1i2}))}) \\ &= e(g,h)^{-s(a_1k_1+a_2k_2)} \cdot e(g,h)^{-sr(a_1b_1w_{011}+a_1b_2w_{012}+x(a_1b_1w_{111}+a_1b_2w_{112})} \\ &\quad \cdot e(g,h)^{-sr(a_2b_1w_{021}+a_2b_2w_{022}+x(a_2b_1w_{121}+a_2b_2w_{122}))} \\ &= A^{-s} \cdot e(g,h)^{-srb_1(a_1w_{011}+a_2w_{021}+x(a_1w_{111}+a_2w_{121})} \\ &\quad \cdot e(g,h)^{-srb_2(a_1w_{012}+xa_1w_{112}+a_2w_{022}+xa_2w_{122})} \\ &= A^{-s} \cdot e(C_1', K_1)^{-1} \cdot e(C_2', K_2)^{-1}. \end{aligned}$$

Hence, computing

$$\begin{aligned} &C \cdot e(C_1, K_1') \cdot e(C_2, K_2') \cdot e(C_1', K_1) \cdot e(C_2', K_2) \\ &= M \cdot A^{-s} \cdot A^{-s} \cdot e(C_1', K_1)^{-1} \cdot e(C_2', K_2)^{-1} \cdot e(C_1', K_1) \cdot e(C_2', K_2) = M \end{aligned}$$

yields the original plaintext message.

They prove the following:

Proposition 2. *The identity-based encryption scheme in Definition 21 is fully CPA-secure and anonymous.*

F.3 Our CCA-transformation for predicate encodings

Definition 22 (CCA-secure CGW15-IBE). *The CCA-secure version of the anonymous identity-based encryption scheme proposed by Chen, Gay and Wee, obtained by applying our CCA-transformations in Sections 3 and 4 is defined as follows. (Note that, because the KEM in Definition 21 is special decomposable and both* $\mathrm{CT}_{2,x}$ *and* $\mathrm{CT}_{3,x'}$ *are uniquely determined by* CT_1*, we use the variation in Section 3.3. In this way, we obtain a strict separation between the KEM and DEM.)*

- Setup(λ)*: On input the security parameter* λ*, the algorithm generates three groups* $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ *of prime order* p *with generators* $g \in \mathbb{G}$ *and* $h \in \mathbb{H}$*, and chooses a pairing* $e \colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$*. It also specifies a collision-resistant hash function* $\mathcal{H} \colon \{0,1\}^* \to \mathbb{Z}_p$*, an authenticated symmetric encryption scheme* $\mathrm{SE} = (\mathrm{Enc_K}, \mathrm{Dec_K})$ *with* $\mathcal{K}(\lambda) = \mathbb{G}_T$*, and a random-prefix collision-resistant hash function* $\mathrm{RPC} \colon \{0,1\}^\lambda \times \mathbb{G} \to \mathbb{Z}_p$*. It then generates random* $k_i, a_i, b_i, w_{0ij}, w_{1ij}, w_{0ij}', w_{1ij}', u_{ij} \in_R \mathbb{Z}_p$ *for all* $i, j \in \{1,2\}$*. It outputs* $\mathrm{MSK} = (\{k_i, a_i, b_i,$

$w_{0ij}, w_{1ij}, w'_{0ij}, w'_{1ij}, u_{ij}\}_{i,j\in\{1,2\}})$ *as the master secret key and publishes the domain parameters* $(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, \mathcal{H})$ *and the master public key as*

$$\mathrm{MPK} = (g, h, A = e(g,h)^{a_1k_1+a_2k_2}, A_1 = g^{a_1}, A_2 = g^{a_2},$$
$$\{W_{l,j} = g^{a_1w_{l1j}+a_2w_{l2j}}\}_{j\in\{1,2\},l\in\{0,1\}}, \{W'_{l,j} = g^{a_1w'_{l1j}+a_2w'_{l2j}}\}_{j\in\{1,2\},l\in\{0,1\}}).$$

- KeyGen(MSK, ID)*: On input identifier* $\mathsf{ID} \in \{0,1\}^*$*, the PKG first hashes* $x = \mathcal{H}(\mathsf{ID})$*, and then generates random integers* $r \in_R \mathbb{Z}_p$*, sets* $k_{i,1} \leftarrow b_1u_{i1} + b_2u_{i2}$ *and* $k_{i,2} \leftarrow k_i - b_1u_{i1} - b_2u_{i2}$*, and computes the secret key as*

$$\mathrm{SK}_{\mathsf{ID}} = (\{K_i = h^{rb_i}, K'_{i,1} = h^{k_{i,1}-r(b_1w_{0i1}+b_2w_{0i2}+x(b_1w_{1i1}+b_2w_{1i2}))},$$
$$K'_{i,2} = h^{k_{i,2}-r(b_1w'_{0i1}+b_2w'_{0i2})}, K''_{i,2} = h^{-r(b_1w'_{1i1}+b_2w'_{1i2})}\}_{i\in\{1,2\}}).$$

- Encrypt(MPK, ID, M)*: Message* $M \in \{0,1\}^*$ *is encrypted under identifier* ID *by first hashing* $x = \mathcal{H}(\mathsf{ID})$*, then picking random integer* $s \in_R \mathbb{Z}_p$*, and computing the ciphertext as*

$$\mathrm{CT}_{\mathsf{ID}} = \big(\mathrm{CT}_{\mathrm{sym}} = \mathrm{Enc}_{\mathrm{K}}(M), \{C_i = A_i^s,$$
$$C'_{i,1} = (W_{0i}W_{1i}^x)^s, C'_{i,2} = (W'_{0i}W_{1i}'^{x'})^s\}_{i\in\{1,2\}}, k\big),$$

  where $k \in_R \{0,1\}^\lambda$*,* $x' \leftarrow \mathrm{RPC}(k, C_1\|C_2)$*, and* $\mathrm{K} \leftarrow A^s$*.*
- Decrypt(MPK, $\mathrm{SK}_{\mathsf{ID}'}$, $\mathrm{CT}_{\mathsf{ID}}$)*: Suppose that* $\mathsf{ID} = \mathsf{ID}'$*, then the ciphertext can decrypted by computing* $y' \leftarrow \mathrm{RPC}(k, C_1\|C_2)$*,*

$$\mathrm{K}' = e(C_1, K'_{1,1}K'_{1,2}K_{1,2}''^{y'}) \cdot e(C_2, K'_{2,1}K'_{2,2}K_{2,2}''^{y'}) \cdot e(C'_{1,1}C'_{1,2}, K_1) \cdot e(C'_{2,1}C'_{2,2}, K_2),$$

  and retrieving $M' \leftarrow \mathrm{Dec}_{\mathrm{K}'}(\mathrm{CT}_{\mathrm{sym}})$*.*

Corollary 1. *The scheme in Definition 22 is CCA-secure and anonymous.*

Proof. This follows directly from Theorem 1 and Proposition 1. Note that we use that the ciphertext part $\{C'_{i,1} = (W_{0i}W_{1i}^x)^s, C'_{i,2} = (W'_{0i}W_{1i}'^{x'})^s\}_{i\in\{1,2\}}$ is uniquely defined by C_1 and C_2. □

This scheme can be further optimized. In particular, for its correctness, we do not require that $K_{i,1}$ and $K_{i,2}$, and $C'_{i,1}$ and $C'_{i,2}$ (for $i \in \{1,2\}$) are given separately during key generation and encryption, respectively, for decryption to work. We also show that we do not need these to be separate for its security either. First, we define the optimized version of this scheme:

Definition 23 (Optimized CCA-secure CGW-IBE). *The optimized CCA-secure version of the anonymous identity-based encryption scheme proposed by Chen, Gay and Wee, obtained by applying our CCA-transformations in Sections 3 and 4 is defined as follows.*

- Setup(λ)*: On input the security parameter λ, the algorithm generates three groups $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ of prime order p with generators $g \in \mathbb{G}$ and $h \in \mathbb{H}$, and chooses a pairing $e \colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$. It also specifies a collision-resistant hash function $\mathcal{H} \colon \{0,1\}^* \to \mathbb{Z}_p$, an authenticated symmetric encryption scheme* $\mathrm{SE} = (\mathrm{Enc}_{\mathrm{K}}, \mathrm{Dec}_{\mathrm{K}})$ *with $\mathcal{K}(\lambda) = \mathbb{G}_T$, and a random-prefix collision-resistant hash function* $\mathrm{RPC} \colon \{0,1\}^\lambda \times \mathbb{G} \to \mathbb{Z}_p$*. It then generates random $k_i, a_i, b_i, w_{0ij}, w_{1ij}, w'_{ij} \in_R \mathbb{Z}_p$ for all $i, j \in \{1,2\}$. It outputs* $\mathrm{MSK} = (\{k_i, a_i, b_i, w_{0ij}, w_{1ij}, w'_{ij}\}_{i,j \in \{1,2\}})$ *as the master secret key and publishes the domain parameters $(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, \mathcal{H})$ and the master public key as*
$$\mathrm{MPK} = (g, h, A = e(g,h)^{a_1k_1 + a_2k_2}, A_1 = g^{a_1}, A_2 = g^{a_2},$$
$$\{W_{l,j} = g^{a_1 w_{l1j} + a_2 w_{l2j}}\}_{j \in \{1,2\}, l \in \{0,1\}}, \{W'_j = g^{a_1 w'_{1j} + a_2 w'_{2j}}\}_{j \in \{1,2\}}).$$
- KeyGen$(\mathrm{MSK}, \mathsf{ID})$*: On input identifier $\mathsf{ID} \in \{0,1\}^*$, the PKG first hashes $x = \mathcal{H}(\mathsf{ID})$, and then generates random integers $r \in_R \mathbb{Z}_p$, and computes the secret key as*
$$\mathrm{SK}_{\mathsf{ID}} = (\{K_i = h^{rb_i}, K'_{i,1} = h^{k_i - r(b_1 w_{0i1} + b_2 w_{0i2} + x(b_1 w_{1i1} + b_2 w_{1i2}))},$$
$$K'_{i,2} = h^{-r(b_1 w'_{i1} + b_2 w'_{i2})}\}_{i \in \{1,2\}}).$$
- Encrypt$(\mathrm{MPK}, \mathsf{ID}, M)$*: Message $M \in \{0,1\}^*$ is encrypted under identifier* ID *by first hashing $x = \mathcal{H}(\mathsf{ID})$, then picking random integer $s \in_R \mathbb{Z}_p$, and computing the ciphertext as*
$$\mathrm{CT}_{\mathsf{ID}} = \big(\mathrm{CT}_{\mathrm{sym}} = \mathrm{Enc}_{\mathrm{K}}(M), \{C_i = A_i^s, C'_i = (W_{0i} W_{1i}^x W_i'^{x'})^s\}_{i \in \{1,2\}}, k\big),$$
where $k \in_R \{0,1\}^\lambda$, $x' \leftarrow \mathrm{RPC}(k, C_1 \| C_2)$, and $\mathrm{K} \leftarrow A^s$.
- Decrypt$(\mathrm{MPK}, \mathrm{SK}_{\mathsf{ID}'}, \mathrm{CT}_{\mathsf{ID}})$*: Suppose that $\mathsf{ID} = \mathsf{ID}'$, then the ciphertext can decrypted by computing $y' \leftarrow \mathrm{RPC}(k, C_1 \| C_2)$,*
$$\mathrm{K}' = e(C_1, K'_{1,1} K'^{y'}_{1,2}) \cdot e(C_2, K'_{2,1} K'^{y'}_{2,2}) \cdot e(C'_1, K_1) \cdot e(C'_2, K_2),$$
and retrieving $M' \leftarrow \mathrm{Dec}_{\mathrm{K}'}(\mathrm{CT}_{\mathrm{sym}})$.

Proposition 3. *The scheme in Definition 23 is fully CCA-secure.*

Proof. We show this by reducing the CPA-security of the associated EPE variant of the optimized scheme to the CPA-security of the associated EPE variant of the basic scheme (which essentially remove the symmetric encryption scheme from the CCA-secure variants of these schemes).

Let $\mathcal{A}_1$ be an attacker that can break the SEPE scheme associated with the scheme in Definition 23. We construct an attacker $\mathcal{A}_2$ that can break the EPE scheme associated with the scheme in Definition 22. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the challengers in the games with $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively.

- **Setup phase:** In the setup, challenger $\mathcal{C}_2$ generates a master public key as $\mathrm{MPK} \leftarrow (g, h, A = e(g,h)^{a_1k_1 + a_2k_2}, A_1 = g^{a_1}, A_2 = g^{a_2}, \{W_{l,j} = g^{a_1 w_{l1j} + a_2 w_{l2j}}\}_{j \in \{1,2\}, l \in \{0,1\}},$

$\{W'_{l,j} = g^{a_1 w'_{l1j} + a_2 w'_{l2j}}\}_{j \in \{1,2\}, l \in \{0,1\}})$, and sends it to attacker $\mathcal{A}_2$. Challenger $\mathcal{C}_1$ sets

$$\overline{\mathrm{MPK}} \leftarrow (g, h, A, A_1, A_2, \{\bar{W}_{0,j} = W_{0,j} \cdot W'_{0,j}, \bar{W}_{1,j} = W_{1,j}, \bar{W}'_j = W'_{1,j}\}_{j \in \{1,2\}}),$$

and sends it to attacker $\mathcal{A}_1$.

- **First query phase:** For each identity ID for which attacker $\mathcal{A}_1$ requests a secret key, we relay the request to challenger $\mathcal{C}_2$, who generates

  $$\mathrm{SK}_{\mathsf{ID}} = (\{K_i = h^{r b_i}, K'_{i,1} = h^{-k_{i,1} - r(b_1 w_{0i1} + b_2 w_{0i2} + x(b_1 w_{1i1} + b_2 w_{1i2}))},$$
  $$K'_{i,2} = h^{-k_{i,2} - r(b_1 w'_{0i1} + b_2 w'_{0i2})}, K''_{i,2} = h^{-r(b_1 w'_{1i1} + b_2 w'_{1i2})}\}_{i \in \{1,2\}}).$$

  Challenger $\mathcal{C}_1$ then sets

  $$\overline{\mathrm{SK}}_{\mathsf{ID}} = (\{K_i, \bar{K}'_{i,1} = K'_{i,1} \cdot K'_{i,2}, \bar{K}'_{i,2} = K''_{i,2}\}_{i \in \{1,2\}}).$$

- **Challenge phase:** Attacker $\mathcal{A}_1$ sends a challenge identity ID^* and two messages M_0, M_1 to challenger $\mathcal{C}_1$, who relays these to challenger $\mathcal{C}_2$. The challenger flips a coin $\beta \in_R \{0,1\}$ and sends back ciphertext

  $$\mathrm{CT}^*_{\mathsf{ID}^*} = \big(C = M_\beta \cdot A^s, \{C_i = A_i^s,$$
  $$C'_{i,1} = (W_{0i} W_{1i}^x)^s, C'_{i,2} = (W'_{0i} W_{1i}'^{x'})^s\}_{i \in \{1,2\}}\big).$$

  Challenger $\mathcal{C}_1$ then sends the ciphertext

  $$\overline{\mathrm{CT}}^*_{\mathsf{ID}^*} = \big(C, \{C_i, \bar{C}'_i = C'_{i,1} \cdot C'_{i,2}\}_{i \in \{1,2\}}\big)$$

  to attacker $\mathcal{A}_1$.
- **Second query phase:** This phase is identical to the first query phase.
- **Guessing phase:** Attacker $\mathcal{A}_1$ outputs a guess β' for β, which attacker $\mathcal{A}_2$ also outputs as its guess. □

It also follows quickly (by slightly adjusting the proof of Proposition 1) that the scheme is anonymous.

Corollary 2. *The scheme in Definition 23 is anonymous.*

F.4 CCA-security with the FO-transformation

We compare our CCA-transformed variants of CGW-IBE with a CCA-variant obtained by applying the Fujisaki-Okamoto transform [29,35]. In particular, we apply the transformation yielding an explicit rejection during the decapsulation that does not take the ciphertext of the KEM as input to the hash that is used to derive a symmetric key.

Definition 24 (CCA-secure CGW-IBE with FO (CCA-CGW-IBE-FO)). *The CCA-secure variant of the anonymous identity-based encryption scheme proposed by Chen, Gay and Wee obtained from the FO-transform [35] is defined as follows.*

- Setup(λ)*: On input the security parameter λ, the algorithm generates three groups $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ of prime order p with generators $g \in \mathbb{G}$ and $h \in \mathbb{H}$, and chooses a pairing $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$. It also specifies a collision-resistant hash function $\mathcal{H}\colon \{0,1\}^* \to \mathbb{Z}_p$, a cryptographic hash function $\mathcal{G}\colon \mathbb{G}_T \to \mathbb{Z}_p$, a key derivation function* KDF: $\mathbb{G}_T \to \{0,1\}^{2\lambda}$, *and an authenticated encryption scheme* SE $= (\mathrm{Enc_K}, \mathrm{Dec_K})$ *with $\mathcal{K}(\lambda) = \{0,1\}^{2\lambda}$. It then generates random $k_i, a_i, b_i, w_{0ij}, w_{1ij} \in_R \mathbb{Z}_p$ for all $i, j \in \{1,2\}$. It outputs* $\mathrm{MSK} = (\{k_i, a_i, b_i, w_{0ij}, w_{1ij}\}_{i,j\in\{1,2\}})$ *as the master secret key and publishes the domain parameters $(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, \mathcal{H})$ and the master public key as*

$$\mathrm{MPK} = (g, h, A = e(g,h)^{a_1k_1+a_2k_2},$$
$$A_1 = g^{a_1}, A_2 = g^{a_2}, \{W_{l,j} = g^{a_1w_{l1j}+a_2w_{l2j}}\}_{j\in\{1,2\},l\in\{0,1\}}).$$

- KeyGen(MSK, ID)*: On input identifier $\mathsf{ID} \in \{0,1\}^*$, the PKG first hashes $x = \mathcal{H}(\mathsf{ID})$, and then generates random integer $r \in_R \mathbb{Z}_p$ and computes the secret key as*

$$\mathrm{SK}_{\mathsf{ID}} = (\{K_i = h^{rb_i}, K'_i = h^{-k_i - r(b_1w_{0i1}+b_2w_{0i2}+x(b_1w_{1i1}+b_2w_{1i2}))}\}_{i\in\{1,2\}}).$$

- Encrypt(MPK, ID, M)*: Message $M \in \{0,1\}^*$ is encrypted under identifier ID by first hashing $x = \mathcal{H}(\mathsf{ID})$, then picking random $M' \in_R \mathbb{G}_T$, setting $s \leftarrow \mathcal{G}(M')$, and computing the ciphertext as*

$$\mathrm{CT}_{\mathsf{ID}} \leftarrow (\mathrm{CT_{sym}} = \mathrm{Enc_K}(M), \mathrm{Encrypt}'(\mathrm{MPK}, \mathsf{ID}, M'; s)),$$

where $\mathrm{K} \leftarrow \mathrm{KDF}(M')$, *and*

$$\mathrm{Encrypt}'(\mathrm{MPK}, \mathsf{ID}, M'; s) = \big(C = M' \cdot A^s, \{C_i = A_i^s, C'_i = (W_{0i}W_{1i}^x)^s\}_{i\in\{1,2\}}\big).$$

- Decrypt(MPK, $\mathrm{SK}_{\mathsf{ID}'}$, $\mathrm{CT}_{\mathsf{ID}}$)*: Suppose that $\mathsf{ID} = \mathsf{ID}'$, then the ciphertext can decrypted by computing*

$$M'' = C \cdot e(C_1, K'_1) \cdot e(C_2, K'_2) \cdot e(C'_1, K_1) \cdot e(C'_2, K_2),$$

then verifying whether

$$(C, C_1, C_2, C'_1, C'_2) \overset{?}{=} \mathrm{Encrypt}'(\mathrm{MPK}, \mathsf{ID}, M''; \mathcal{G}(M''))$$

holds and, if so, return $M \leftarrow \mathrm{Dec}_{\mathrm{KDF}(M'')}(\mathrm{CT_{sym}})$.

G A large-universe CP-ABE scheme

G.1 Access structures

Definition 25 (Monotone access structures [12]). *Let $\{a_1, ..., a_n\}$ be a set of attributes. An access structure is a collection $\mathbb{A}$ of non-empty subsets of $\{a_1, ..., a_n\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets that are not in $\mathbb{A}$ are called the unauthorized sets. An access structure $\mathbb{A} \subseteq 2^{\{a_1,...,a_n\}}$ is monotone if for all B, C holds: $B \in \mathbb{A}$ and $B \subseteq C$, then also $C \in \mathbb{A}$.*

We represent access policies $\mathbb{A}$ by linear secret sharing scheme (LSSS) matrices, which support monotone span programs [12,34].

Definition 26 (Access structures represented by LSSS matrices [34]). *An access structure can be represented as a pair $\mathbb{A} = (\mathbf{A}, \rho)$ such that $\mathbf{A} \in \mathbb{Z}_p^{n_1 \times n_2}$ is an LSSS matrix, where $n_1, n_2 \in \mathbb{N}$, and ρ is a function that maps its rows to attributes in the universe. Then, for some vector with randomly generated entries $\mathbf{v} = (s, v_2, ..., v_{n_2}) \in \mathbb{Z}_p^{n_2}$, the i-th secret generated by this matrix is $\lambda_i = \mathbf{A}_i \mathbf{v}^\intercal$, where $\mathbf{A}_i$ denotes the i-th row of $\mathbf{A}$. In particular, if $\mathcal{S}$ satisfies $\mathbb{A}$, then there exist a set of rows $\Upsilon = \{i \in [n_1] \mid \rho(i) \in \mathcal{S}\}$ and coefficients $\varepsilon_i \in \mathbb{Z}_p$ for all $i \in \Upsilon$ such that $\sum_{i \in \Upsilon} \varepsilon_i \mathbf{A}_i = (1, 0, ..., 0)$, and by extension $\sum_{i \in \Upsilon} \varepsilon_i \lambda_i = s$, holds.*

In [40], Lewko and Waters devise a way to convert Boolean formulas into LSSS matrices. For our implementations, we use strictly ANDs in our policies, which ensures that the matrix $\mathbf{A}$ is a square matrix in which the number of rows and columns is equal to the length n of the policy, the first row consists of 1s in the first two entries (and the rest is 0), the last row has -1 in the last column and the rest all-zero, and the rest of the rows $i \in [2, n-1]$ is of the form $\mathbf{1}_{i+1}^n - \mathbf{1}_i^n$:

$$\mathbf{A}_1 = \mathbf{1}_1^n + \mathbf{1}_2^n \qquad \mathbf{A}_i = \mathbf{1}_{i+1}^n - \mathbf{1}_i^n \qquad \mathbf{A}_n = -\mathbf{1}_n^n,$$

for all $i \in [2, n-1]$.

G.2 Ciphertext-policy ABE

Definition 27 (Ciphertext-policy ABE [13]). *A ciphertext-policy ABE (CP-ABE) scheme consists of four algorithms:*

- $\text{Setup}(\lambda) \to (\text{MPK}, \text{MSK})$*: The setup takes as input a security parameter λ, it outputs the master public-secret key pair* (MPK, MSK).
- $\text{KeyGen}(\mathcal{S}, \text{MSK}) \to \text{SK}_\mathcal{S}$*: The key generation takes as input a set of attributes $\mathcal{S}$ and the master secret key* MSK*, and outputs a secret key* $\text{SK}_\mathcal{S}$.
- $\text{Encrypt}(M, \mathbb{A}, \text{MPK}) \to \text{CT}_\mathbb{A}$*: The encryption takes as input a plaintext message M, an access policy $\mathbb{A}$ and the master public keys* MPK*. It outputs a ciphertext* $\text{CT}_\mathbb{A}$.
- $\text{Decrypt}(\text{CT}_\mathbb{A}, \text{SK}_\mathcal{S}) \to M'$*: The decryption takes as input the ciphertext* $\text{CT}_\mathbb{A}$ *that was encrypted under an access policy $\mathbb{A}$, and a secret key* $\text{SK}_\mathcal{S}$ *associated with a set of attributes $\mathcal{S}$. It succeeds and outputs the plaintext message M' if $\mathcal{S}$ satisfies $\mathbb{A}$. Otherwise, it aborts.*

A scheme is called correct if decryption of a ciphertext with secret key yields the original plaintext message.

Large-universe ABE. We consider a scheme to be large-universe if it does not impose bounds on the size of the universe.

G.3 The selectively secure variant of RW13

Definition 28 (The RW13 CP-ABE scheme [43]). *The ciphertext-policy attribute-based encryption scheme by Rouselakis and Waters (RW13) [43] is defined as follows.*

- $\mathrm{Setup}(\lambda)$*: Taking as input the security parameter* λ*, the setup generates three groups* $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ *of prime order* p *with generators* $g \in \mathbb{G}, h \in \mathbb{H}$*, and chooses a pairing* $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$*. The setup also defines the universe of attributes* $\mathcal{U} = \mathbb{Z}_p$*. It then generates random* $\alpha, b, b_0, b_1, b' \in_R \mathbb{Z}_p$*. It outputs* $\mathrm{MSK} = (\alpha, b, b_0, b_1, b')$ *as its master secret key and publishes the master public key as*
$$\mathrm{MPK} = (g, h, A = e(g,h)^{\alpha}, B = g^{b}, B_0 = g^{b_0}, B_1 = g^{b_1}, B' = g^{b'}).$$
- $\mathrm{KeyGen}(\mathrm{MSK}, \mathcal{S})$*: On input a set of attributes* $\mathcal{S}$*, the algorithm generates random integers* $r, r_{\mathrm{att}} \in_R \mathbb{Z}_p$ *for each* $\mathrm{att} \in \mathcal{S}$*, letting* $x_{\mathrm{att}} \in \mathbb{Z}_p$ *denote the representation of* att *in* $\mathbb{Z}_p$ *and computes the secret key as*
$$\mathrm{SK}_{\mathcal{S}} = (K = h^{\alpha - rb}, K' = h^{r}, \{K_{1,\mathrm{att}} = h^{-r_{\mathrm{att}}(b_1 x_{\mathrm{att}} + b_0) - rb'}, K_{2,\mathrm{att}} = h^{r_{\mathrm{att}}}\}_{\mathrm{att} \in \mathcal{S}}).$$
- $\mathrm{Encrypt}(M, \mathrm{MPK}, \mathbb{A})$*: A message* $M \in \mathbb{G}_T$ *is encrypted under* $\mathbb{A} = (\mathbf{A}, \rho)$ *with* $\mathbf{A} \in \mathbb{Z}_p^{n_1 \times n_2}$ *and* $\rho\colon [n_1] \to \mathcal{U}$ *by generating random integers* $s, s_i, v_j \in_R \mathbb{Z}_p$ *for all* $i \in [n_1]$ *and* $j \in [2, n_2]$*, and computes the ciphertext as*
$$\mathrm{CT}_{\mathbb{A}} = (C = M \cdot A^{s}, C' = g^{s}, \{C_{1,j} = B^{\lambda_j} (B')^{s_j},$$
$$C_{2,j} = \left(B_1^{\rho(j)} B_0\right)^{s_j}, C_{3,j} = g^{s_j}\}_{j \in [1, n_1]}),$$
such that λ_i *denotes the i-th entry of* $\mathbf{A} \cdot (s, v_2, ..., v_{n_2})^{\intercal}$.
- $\mathrm{Decrypt}(\mathrm{SK}_{\mathcal{S}}, \mathrm{CT}_{\mathbb{A}})$*: Suppose that* $\mathcal{S}$ *satisfies* $\mathbb{A}$*, and suppose* $\Upsilon = \{j \in [1, n_1] \mid \rho(j) \in \mathcal{S}\}$*, such that* $\{\varepsilon_j \in \mathbb{Z}_p\}_{j \in \Upsilon}$ *exist with* $\sum_{i \in \Upsilon} \varepsilon_j \mathbf{A}_j = (1, 0, ..., 0)$*. Then the plaintext* M *is retrieved by computing*
$$C / \left(e(C', K) \cdot e(\prod_{j \in \Upsilon} C_{1,j}^{\varepsilon_j}, K') \prod_{j \in \Upsilon} \left(e(C_{2,j}^{\varepsilon_j}, K_{2,\rho(j)}) \cdot e(C_{3,j}^{\varepsilon_j}, K_{1,\rho(j)}) \right) \right).$$
Note that, in the case of AND-gates, $\varepsilon_j \in \{0, 1\}$.

G.4 A fully secure variant of RW13

We present a fully secure variant of this scheme, given in the Agrawal-Chase framework (AC17) [4].

Definition 29 (The fully secure RW13 CP-ABE scheme (RWAC) [4]). *The ciphertext-policy attribute-based encryption scheme by Rouselakis and Waters (RW13) [43] is defined in the Agrawal-Chase framework, using the prime-order dual system groups for SXDH in [27], as follows.*

- Setup(λ)*: Taking as input the security parameter* λ*, the setup generates three groups* $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ *of prime order* p *with generators* $g \in \mathbb{G}, h \in \mathbb{H}$*, and chooses a pairing* $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$*. The setup also defines the universe of attributes* $\mathcal{U} = \mathbb{Z}_p$*. It then generates random* $\alpha_1, \alpha_2, d_1, d_2, d_3, d_4, d_5, b_i, b_{0,i}, b_{1,i}, b'_i \in_R \mathbb{Z}_p$ *for all* $i \in \{1, 2, 3\}$*, such that* $d_1 d_4 \neq d_2 d_3$*. It outputs* $\mathrm{MSK} = (\alpha_1, \alpha_2, d_1, d_2, d_3, d_4, d_5, b_i, b_{0,i}, b_{1,i}, b'_i)$ *as its master secret key and publishes the master public key as*
$$\mathrm{MPK} = (g, h, A = e(g,h)^{\alpha_1 d_1 + \alpha_2 d_2}, \{g_i = g^{d_i}, B_i = g^{b_1 d_i + b_3 d_{i+2}},$$
$$\{B_{l,i} = g^{b_{l,1} d_i + b_{l,3} d_{i+2}}\}_{l \in \{0,1\}}, B'_i = g^{b'_1 d_i + b'_3 d_{i+2}}\}_{i \in \{1,2\}}).$$
- KeyGen$(\mathrm{MSK}, \mathcal{S})$*: On input a set of attributes* $\mathcal{S}$*, the algorithm generates random integers* $r, r_{\mathrm{att}} \in_R \mathbb{Z}_p$ *for each* $\mathrm{att} \in \mathcal{S}$*, letting* $x_{\mathrm{att}} \in \mathbb{Z}_p$ *denote the representation of* att *in* $\mathbb{Z}_p$ *and computes the secret key as*
$$\mathrm{SK}_{\mathcal{S}} = (\{K_i = h^{\alpha_i - r\bar{b}_i}, K'_1 = h^{r d_4 d_6}, K'_2 = h^{-r d_3 d_6},$$
$$K_{1,\mathrm{att},i} = h^{-r_{\mathrm{att}}(\bar{b}_{1,i} x_{\mathrm{att}} + \bar{b}_{0,i}) - r\bar{b}'_i},$$
$$K_{2,\mathrm{att},1} = h^{r_{\mathrm{att}} d_4 d_6}, K_{2,\mathrm{att},2} = h^{-r_{\mathrm{att}} d_3 d_6}\}_{i \in \{1,2\}, \mathrm{att} \in \mathcal{S}}),$$
where for $l \in \{0, 1\}$*, we have*
$$d_6 = \frac{d_5}{d_1 d_4 - d_2 d_3}$$
$$\bar{b}_1 = d_6(b_1 d_4 - b_2 d_2), \bar{b}_2 = d_6(-b_1 d_3 + b_2 d_1),$$
$$\bar{b}_{l,1} = d_6(b_{l,1} d_4 - b_{l,2} d_2), \bar{b}_{l,2} = d_6(-b_{l,1} d_3 + b_{l,2} d_1),$$
$$\bar{b}'_1 = d_6(b'_1 d_4 - b'_2 d_2), \bar{b}'_2 = d_6(-b'_1 d_3 + b'_2 d_1).$$
- Encrypt$(M, \mathrm{MPK}, \mathbb{A})$*: A message* $M \in \mathbb{G}_T$ *is encrypted under* $\mathbb{A} = (\mathbf{A}, \rho)$ *with* $\mathbf{A} \in \mathbb{Z}_p^{n_1 \times n_2}$ *and* $\rho\colon [n_1] \to \mathcal{U}$ *by generating random integers* s,s_i,$v_j \in_R \mathbb{Z}_p$ *for all* $i \in [n_1]$ *and* $j \in [2, n_2]$*, and computes the ciphertext as*
$$\mathrm{CT}_{\mathbb{A}} = (C = M \cdot A^s, \{C'_i = g_i^s, C_{1,i,j} = B_i^{A_{j,1} s} g_i^{\lambda_j} (B'_i)^{s_j},$$
$$C_{2,i,j} = \left(B_{1,i}^{\rho(j)} B_{0,i}\right)^{s_j}, C_{3,i,j} = g_i^{s_j}\}_{i \in \{1,2\}, j \in [1,n_1]}),$$
such that λ_j *denotes the* j*-th entry of* $\mathbf{A} \cdot (0, v_2, ..., v_{n_2})^{\intercal}$*.*
- Decrypt$(\mathrm{SK}_{\mathcal{S}}, \mathrm{CT}_{\mathbb{A}})$*: Suppose that* $\mathcal{S}$ *satisfies* $\mathbb{A}$*, and suppose* $\Upsilon = \{j \in [1, n_1] \mid \rho(j) \in \mathcal{S}\}$*, such that* $\{\varepsilon_j \in \mathbb{Z}_p\}_{j \in \Upsilon}$ *exist with* $\sum_{i \in \Upsilon} \varepsilon_j \mathbf{A}_j = (1, 0, ..., 0)$*. Then the plaintext* M *is retrieved by computing*
$$C / \textstyle\prod_{i \in \{1,2\}} \left(e(C'_i, K_i) \cdot e(\prod_{j \in \Upsilon} C_{1,i,j}^{\varepsilon_j}, K'_i) \prod_{j \in \Upsilon} \left(e(C_{2,i,j}^{\varepsilon_j}, K_{2,\rho(j),i}) \cdot e(C_{3,i,j}^{\varepsilon_j}, K_{1,\rho(j),i}) \right) \right).$$

G.5 Our CCA-transformation for PES

We transform the fully CPA-secure scheme in Definition 29 to a fully CCA-secure scheme with the transformation for PES in Section 4. Like for the CGW-IBE, because RWAC is special decomposable, we use the variation in Section 3.3, which hashes the partial ciphertext $\mathrm{CT}_{2,x}$ with the RPC hash.

Definition 30 (The fully CCA-secure RWAC scheme). *The CCA-secure version obtained with our transformation in Section 4 of RWAC [43,4,10] is defined as follows.*

- Setup(λ)*: Taking as input the security parameter λ, the setup generates three groups $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ of prime order p with generators $g \in \mathbb{G}, h \in \mathbb{H}$, and chooses a pairing $e \colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$. It also specifies a collision-resistant hash function $\mathcal{H} \colon \{0,1\}^* \to \mathbb{Z}_p$, an authenticated symmetric encryption scheme* $\mathrm{SE} = (\mathrm{Enc}_\mathrm{K}, \mathrm{Dec}_\mathrm{K})$ *with $\mathcal{K}(\lambda) = \mathbb{G}_T$, and a random-prefix collision-resistant hash function* $\mathrm{RPC} \colon \{0,1\}^\lambda \times \mathbb{G} \to \mathbb{Z}_p$. *The setup also defines the universe of attributes$\mathcal{U} = \mathbb{Z}_p$. It then generates random $\alpha_1, \alpha_2, d_1, d_2, d_3, d_4, d_5, b_i, b_{0,i}, b_{1,i}, b'_{0,i}, b'_{1,i}, b'_i \in_R \mathbb{Z}_p$ for all $i \in \{1,2,3\}$, such that $d_1 d_4 \neq d_2 d_3$. It outputs* $\mathrm{MSK} = (\alpha_1, \alpha_2, d_1, d_2, d_3, d_4, d_5, b_i, b_{0,i}, b_{1,i}, b'_{1,i}, b'_i)$ *as its master secret key and publishes the master public key as*

$$\mathrm{MPK} = (g, h, A = e(g,h)^{\alpha_1 d_1 + \alpha_2 d_2}, \{g_i = g^{d_i}, B_i = g^{b_1 d_i + b_3 d_{i+2}},$$
$$\{B_{l,i} = g^{b_{l,1} d_i + b_{l,3} d_{i+2}}, B'_{l,i} = g^{b'_{l,1} d_i + b'_{l,3} d_{i+2}}\}_{l \in \{0,1\}}, B'_i = g^{b'_1 d_i + b'_3 d_{i+2}}\}_{i \in \{1,2\}}).$$

- KeyGen($\mathrm{MSK}, \mathcal{S}$)*: On input a set of attributes $\mathcal{S}$, the algorithm generates random integers $r, \alpha_{1,1}, \alpha_{2,1}, r_{\mathrm{att}} \in_R \mathbb{Z}_p$ for each* att $\in \mathcal{S}$, *sets $\alpha_{1,2} \leftarrow \alpha_1 - \alpha_{1,1}$ and $\alpha_{2,2} \leftarrow \alpha_2 - \alpha_{2,1}$, letting $x_{\mathrm{att}} = \mathcal{H}(\mathrm{att})$ denote the representation of* att *in $\mathbb{Z}_p$, and computes the secret key as*

$$\mathrm{SK}_{\mathcal{S}} = (\{K_i = h^{\alpha_{i,1} - r\bar{b}_i}, K'_1 = h^{r d_4 d_6}, K'_2 = h^{-r d_3 d_6},$$
$$K_i^{(2)} = h^{\alpha_{i,2} - r\bar{b}'_{0,i}}, K_i^{(3)} = h^{-r\bar{b}'_{1,i}}, K_{1,\mathrm{att},i} = h^{-r_{\mathrm{att}}(\bar{b}_{1,i} x_{\mathrm{att}} + \bar{b}_{0,i}) - r\bar{b}'_i},$$
$$K_{2,\mathrm{att},1} = h^{r_{\mathrm{att}} d_4 d_6}, K_{2,\mathrm{att},2} = h^{-r_{\mathrm{att}} d_3 d_6}\}_{i \in \{1,2\}, \mathrm{att} \in \mathcal{S}}),$$

where for $l \in \{0,1\}$, we have

$$d_6 = \frac{d_5}{d_1 d_4 - d_2 d_3}$$
$$\bar{b}_1 = d_6(b_1 d_4 - b_2 d_2), \bar{b}_2 = d_6(-b_1 d_3 + b_2 d_1),$$
$$\bar{b}_{l,1} = d_6(b_{l,1} d_4 - b_{l,2} d_2), \bar{b}_{l,2} = d_6(-b_{l,1} d_3 + b_{l,2} d_1),$$
$$\bar{b}'_{l,1} = d_6(b'_{l,1} d_4 - b'_{l,2} d_2), \bar{b}'_{l,2} = d_6(-b'_{l,1} d_3 + b'_{l,2} d_1),$$
$$\bar{b}'_1 = d_6(b'_1 d_4 - b'_2 d_2), \bar{b}'_2 = d_6(-b'_1 d_3 + b'_2 d_1).$$

- Encrypt($M, \mathrm{MPK}, \mathbb{A}$)*: A message $M \in \{0,1\}^*$ is encrypted under $\mathbb{A} = (\mathbf{A}, \rho)$ with $\mathbf{A} \in \mathbb{Z}_p^{n_1 \times n_2}$ and $\rho \colon [n_1] \to \mathcal{U}$ by generating random integers s,s_j,$v_{j'} \in_R \mathbb{Z}_p$ for all $j \in [n_1]$ and $j' \in [2, n_2]$, and computes the ciphertext as*

$$\mathrm{CT}_{\mathbb{A}} = (\mathrm{CT}_{\mathrm{sym}} = \mathrm{Enc}_\mathrm{K}(M), \{C'_i = g_i^s, C_{1,i,j} = B_i^{A_{j,1} s} g_i^{\lambda_j} (B'_i)^{s_j},$$
$$C_{2,i,j} = \left(B_{1,i}^{\rho(j)} B_{0,i}\right)^{s_j}, C_{3,i,j} = g_i^{s_j}, C_{4,i} = \left(B'^{x'}_{1,i} B'_{0,i}\right)^{s}\}_{i \in \{1,2\}, j \in [1,n_1]}, k),$$

such that $k \in \{0,1\}^\lambda$,

$$x' \leftarrow \mathrm{RPC}(k, \{C'_i, C_{1,i,j}, C_{2,i,j}, C_{3,i,j}\}_{i \in \{1,2\}, j \in [1,n_1]}),$$

$\mathrm{K} \leftarrow A^s$*, and* λ_j *denotes the* j*-th entry of* $\mathbf{A} \cdot (0, v_2, ..., v_{n_2})^\intercal$*. Note that, although we use set notation in the input to the hash, the ciphertext components should be concatenated deterministically (in a domain-separated fashion), such that encryption and decryption yield the same output.*

- Decrypt$(\mathrm{SK}_{\mathcal{S}}, \mathrm{CT}_{\mathbb{A}})$*: Suppose that* $\mathcal{S}$ *satisfies* $\mathbb{A}$*, and suppose* $\Upsilon = \{j \in [1, n_1] \mid \rho(j) \in \mathcal{S}\}$*, such that* $\{\varepsilon_j \in \mathbb{Z}_p\}_{j \in \Upsilon}$ *exist with* $\sum_{i \in \Upsilon} \varepsilon_j \mathbf{A}_j = (1, 0, ..., 0)$*. Then the plaintext* M *is retrieved by computing*

$$\mathrm{K}' \leftarrow \prod_{i \in \{1,2\}} \left(e(C'_i, K_i K_i^{(2)} (K_i^{(3)})^{y'}) \cdot e\left(\left(\prod_{j \in \Upsilon} C_{1,i,j}^{\varepsilon_j} \right) \cdot C_{4,i}, K'_i \right) \right.$$
$$\left. \cdot \prod_{j \in \Upsilon} \left(e(C_{2,i,j}^{\varepsilon_j}, K_{2,\rho(j),i}) \cdot e(C_{3,i,j}^{\varepsilon_j}, K_{1,\rho(j),i}) \right) \right),$$

where

$$y' \leftarrow \mathrm{RPC}(k, \{C'_i, C_{1,i,j}, C_{2,i,j}, C_{3,i,j}\}_{i \in \{1,2\}, j \in [1, n_1]}),$$

and then obtaining $M' \leftarrow \mathrm{Dec}_{\mathrm{K}'}(\mathrm{CT}_{\mathrm{sym}})$*.*

G.6 CCA-security with other transformations

We compare the efficiency of the CCA-secure scheme using our transformations with several other CCA-secure variants of RWAC. In particular, we consider the

- FO-transformation [29] using the techniques in [35];
- YAHK-transformation [49] using the delegatability property;
- YAHK-transformation [49] using the verifiability property.

More generic transformations exist, as mentioned in the introduction, but these incur similar computational trade-offs. Rather than implementing fully functional variants of these schemes, we estimate the storage and computational costs based on the operations required in the algorithms of the transformed variants. For instance, the FO-transformed variant calls the encryption algorithm during decryption, and the efficiency of the variants using the YAHK-transformations depends on the efficiency of the chosen OTS. Table 3 estimates the overhead of all variants, and shows that our transformation yields the fastest decryption algorithm. In particular, the additional costs are a small constant.

Table 3: Comparison of the additional storage and computational costs incurred by the CCA-transformation among several CCA-secure variants of RWAC. We do not list the symmetric operations, such as hashes, encryptions and MACs.

Variant	$\vert\mathrm{MPK}\vert$	$\vert\mathrm{SK}_{\mathcal{S}}\vert$	$\vert\mathrm{CT}_{\mathbb{A}}\vert$	KeyGen	Encrypt	Decrypt
FO	-	$s_{\mathbb{G}_T} + 10s_{\mathbb{G}}$	-	-	-	$10\vert\varUpsilon\vert c_{\mathrm{exp},\mathbb{G}}$
Delegatability	-	$8\vert\mathrm{vk}\vert s_{\mathbb{H}}$	$6\vert\mathrm{vk}\vert s_{\mathbb{G}}$	$8\vert\mathrm{vk}\vert c_{\mathrm{exp},\mathbb{H}}$	$6\vert\mathrm{vk}\vert c_{\mathrm{exp},\mathbb{G}}$	$2\vert\mathrm{vk}\vert p$
Verifiability	-	-	$6s_{\mathbb{G}}$	-	$10c_{\mathrm{exp},\mathbb{G}}$	$2\vert\varUpsilon\vert p$
Ours	$4s_{\mathbb{G}}$	$4s_{\mathbb{H}}$	$2s_{\mathbb{G}}$	$4c_{\mathrm{exp},\mathbb{H}}$	$4c_{\mathrm{exp},\mathbb{G}}$	$2c_{\mathrm{exp},\mathbb{H}}$

Note: $c_{\mathrm{exp},\mathbb{G}'}$ = costs of an exponentiation in $\mathbb{G}'$, $s_{\mathbb{G}'}$ = the size of an element in $\mathbb{G}'$, p = the costs of a pairing operation, vk = verification key used in YAHK

Table of Contents