A Practical Compiler for Attribute-Based Encryption: New Decentralized Constructions and More

Marloes Venema[1,2]

[1] University of Wuppertal, Wuppertal, Germany
[2] Radboud University, Nijmegen, the Netherlands
mvenemacrypto@gmail.com

Abstract. The pair encodings framework is an important result in the simplified design of complex attribute-based encryption schemes. In particular, it reduces the effort of proving security of a scheme to proving security of the associated pair encoding, which can then be transformed into a provably secure pairing-based encryption scheme with a compiler. Especially the symbolic property, as introduced by Agrawal and Chase (EUROCRYPT '17), has proven to be a valuable security notion that is both simple to verify and applies to many schemes. Nevertheless, several practical extensions using full-domain hashes or employing multiple authorities cannot be instantiated with this compiler, and therefore still require complicated proof techniques.

In this work, we present the first compiler for attribute-based encryption schemes that supports such extensions. To this end, we generalize the definitions of pair encodings and the symbolic property. With our compiler, we flexibly instantiate any pair encodings that satisfy this new notion of the symbolic property in any pairing-friendly groups, and generically prove the resulting scheme to be selectively secure. To illustrate the effectiveness of our new compiler, we give several new multi-authority and hash-based constructions.

Keywords: attribute-based encryption · multi-authority attribute-based encryption

1 Introduction

Attribute-based encryption (ABE) [50] is a powerful cryptographic primitive that associates the keys and ciphertexts with attributes. ABE is attractive for practice, as it allows for the fine-grained access control on data on a cryptographic level [37,33,55,40]. In 2014, Attrapadung [11] and Wee [58] introduced frameworks for *pair* and *predicate encodings*, respectively, to simplify the design and analysis of complex ABE schemes. Informally speaking, pair encoding schemes abstract a pairing-based ABE scheme to "what happens in the exponent of the keys and ciphertexts". The idea behind these frameworks is that the designer only needs to prove information-theoretic or algebraic notions of security for these encodings. Then, via a *generic compiler*, Attrapadung and Wee construct ABE schemes by instantiating the encodings in some carefully-constructed pairing-friendly groups. Subsequently, they generically prove full security, using dual system encryption techniques [56], of the resulting ABE from the security of the encoding and the security of the groups.

Since its invention, many works have contributed to the pair encodings framework [16,27,12,2,4,9,13,7]. Nowadays, many pairing-based schemes can be captured in this framework, ensuring that these efficiently satisfy a strong notion of security. Not only has the pair encodings framework become a powerful tool in the design of new schemes, it is also possible to generically transform or compose existing schemes [16,4,9,13,7]. As a result, increasingly complex schemes can be constructed without further complicating the security proofs. For example, revocation mechanisms [9,59] and range attributes [14] can be generically and efficiently supported [13].

Arguably the most powerful security notion for pair encodings is the symbolic property, which was first introduced as such by Agrawal and Chase [4], but builds on several prior works, e.g., [42,11,12]. In part, this security notion is more powerful, because more schemes can be captured with it [4]. Moreover, interestingly, the symbolic property is meant to make security proofs easy to verify. In particular, this effort boils down to performing simple linear algebra. This is a much simpler task than verifying complex security reductions that require a significant expertise. From a historical perspective, the symbolic property builds on the ideas behind the more classical proofs, called "program-and-cancel" proofs, which were used to prove selective security in the early days [20,50]. In the selective-security model, the attacker commits to the predicate that they are going to attack before seeing the public keys, which is unreasonable to assume in practice [26].

Nevertheless, even though the symbolic property is strongly linked [42,11,4] to these classical proofs, it is not clear if the symbolic property can be used to prove selective security generically. Of course, this also raises the question of whether we should care about this particularly low-hanging fruit at all. If we can use the symbolic property to build fully secure schemes, then why would we want to use it to build weaker schemes? Our answer to this question is multifold: because the resulting schemes are simpler, more efficient, and we may be able to generically build practical schemes that we cannot build with the current full-security compilers yet [11,12,4]. Notably, those compilers do not readily support various practical properties, e.g.,

- the employment of multiple authorities [24,41,49];
- full-domain hashes, e.g., to achieve large-universeness[3] efficiently [57];
- or flexible instantiations in the pairing-friendly groups [6,1] (which heavily influences the scheme's efficiency [47]).

Fully secure schemes that do satisfy such properties [41,3,52] need to resort to more complicated proof techniques (and on a case-by-case basis), and move

[3] Large-universe ABE can support any string as attribute.

us further away from the simplicity of the symbolic property again. Moreover, because of this complexity, many schemes that do have such desirable properties have turned out to be broken [53]. This is, by any means, much worse than using a scheme that is "only" selectively secure.

In addition, the broader audience seems to have confidence in selectively secure schemes, and considers these to be practical. In particular, selectively secure schemes are typically at least a factor 2 more efficient than similar schemes in the full-security setting [4,55] (assuming they are instantiated in the same pairing-friendly groups). Because their descriptions do not require the use of complex structures such as dual system groups [28,29], they are also simpler and more intuitive. By extension, they are easier to prototype and analyze for any given practical setting [47]. Presumably, these are reasons why many public cryptographic libraries contain many implementations of selectively secure schemes [5,60,46,35], or why half of the schemes considered by the European Telecommunications Standards Institute [34] are selectively secure. All in all, even if the eventual goal is to implement a fully secure scheme, simplifying the design of selectively secure schemes is valuable.

1.1 Our contribution

We propose a new generic compiler. This compiler uses the symbolic property to generically prove selective security of the resulting ABE scheme. With this new compiler, we are able to achieve properties that cannot be generically supported with existing full-security compilers (yet), i.e.,

- multi-authority extensions;
- full-domain hashes;
- flexible instantiations in the pairing-friendly groups.

To achieve these properties, we generalize the definitions of pair encodings and the symbolic property, and introduce mappings that explicitly address the use of hashes and the instantiations of the encodings in the pairing-friendly groups.

New schemes. As a result of our compiler, we also obtain new schemes. In particular, we give new constructions for

- decentralized large-universe multi-authority ciphertext-policy ABE (CP-ABE) for monotone span programs [41,49];
- decentralized non-monotone large-universe multi-authority CP-ABE;
- single-authority CP-ABE and KP-ABE with attribute-wise key generation—i.e., one single user can request keys for different attributes at different points in time [55]—which is the first single-authority scheme that explicitly enjoys this property;
- decentralized identity-based broadcast encryption [13].

Relation to fully secure schemes in the generic group model. Our compiler also strenghtens the connection between selectively and fully secure schemes. Previously, Ambrona et al. [8] showed that any scheme that is not trivially broken is provably fully secure in the generic group model (GGM) [51,21,23]. The class of encoding schemes that they consider overlaps with that of the Agrawal-Chase compiler [4], which is also covered by our compiler. For this class of schemes, we obtain the following result: the compiled scheme is provably fully secure in the GGM (with some non-trivial security loss), and it is provably selectively secure in the standard model under a q-type assumption (which is a type of assumption that becomes stronger as q grows). Possibly, this insight can help the design of fully secure multi-authority schemes in future work.

Supporting practical extensions with full-security compilers. We briefly discuss the difficulty of supporting the aforementioned practical extensions in existing full-security compilers. In principle, it seems that most full-security compilers can readily support any instantiation in the pairing-friendly groups, see e.g., the discussion in [3, §1.1]. This incurs a significant performance penalty: compared to selectively secure schemes, the resulting fully secure schemes are a factor 3 less efficient. However, for full-domain hashes and multi-authority extensions, multiple difficulties need to be overcome. For a discussion on supporting full-domain hashes, we also refer to the discussion in [3, §1.1]. Roughly, the problem is that the structure of the underlying groups of the compiler is considerably more complex than in the selective-security setting. Public-key variables can therefore not simply be instantiated with a full-domain hash like in selectively secure schemes (see e.g., [37]). Lastly, we argue that, with the current tools, we cannot effectively support multi-authority extensions in the full-security setting. First, the structure of most existing multi-authority schemes [41,49,30] is not captured by the pair encodings framework. Second, the proof techniques used for such schemes [41] are more advanced, because the attacker has more power. Hence, the pair encodings framework needs to be extended with respect to these two aspects, which both may require a significantly more intricate approach. In this work, we address the first aspect.

Full security through complexity leveraging or random oracles. Once we have a selectively secure scheme, we can use complexity leveraging [20,26] or random oracles [18,22] to achieve full security. This may yield a more efficient instantiation of the scheme than a scheme built using dual system encryption techniques. For example, the identity-based encryption scheme by Boneh and Boyen [20] is a factor 2-3 more efficient in the random oracle model than its most efficient fully secure counterpart using dual system encryption techniques [27]. Alternatively, if we use complexity leveraging [26], we need to implement the scheme with pairing-friendly groups that provide a higher level of security. Although this also influences the efficiency, it may be more efficient than using dual system encryption techniques.

1.2 Background

Ciphertext-policy ABE. Although our generic compiler is general in the sense that it applies to any ABE, our new constructions are ciphertext-policy ABE schemes [19]. In CP-ABE, the messages are encrypted under access policies (often represented as Boolean formulas over attributes). Subsequently, any user with an authorized secret key can decrypt the message. A key is authorized, if the associated set of attributes satisfies the policy. Owing to this functionality, CP-ABE has proven to be an attractive primitive for practice [19,33,39,55]. However, CP-ABE often employs a single trusted third party called "the key generation authority" that issues the secret keys, which needs to be fully trusted.

Multi-authority ABE. Multi-authority ABE, as first proposed by Chase [24], employs various authorities to mitigate the trust issues in regular ABE. An especially interesting subtype of multi-authority ABE is called "decentralized" ABE [41]. In decentralized ABE[4], the authorities can act fully autonomously, without requiring interaction between one another to act securely or correctly. Although this is a very desirable feature, the number of existing schemes that securely provide this property is limited [41,44,49,30,10]. Of these schemes, few satisfy practical properties such as large-universeness and unboundedness[5]. In fact, only the scheme by Rouselakis and Waters (RW15) [49] satisfies both.

Non-monotone ABE. Another desirable feature in ABE is non-monotonicity, i.e., the support for negations in the policies. Although this property was quite difficult to achieve efficiently, the pair encodings framework can support these generically by applying various transformations [13,15,7]. In this work, we provide both single-authority and decentralized schemes that support the type of negations as first introduced by Okamoto and Takashima (OT) [43], which we call "OT-type negations". In such negations, the label of the attribute also plays a role. In particular, an attribute set satisfies a negation, e.g., "name: NOT Alice", only if it has an attribute with the same label, and the attribute value is not equal to the value of the negated attribute, e.g., "name: Bob". Currently, the only decentralized scheme that is also non-monotone is the scheme by Okamoto and Takashima [44,45].

Generalizing pair encoding schemes. We generalize the definitions of pair encoding schemes and the symbolic property. One of the reasons why multi-authority ABE cannot be captured in the pair encodings framework is that existing multi-authority schemes do not (fully) match the structure of pair encodings. Roughly, pair encoding schemes consider schemes of the form:

$$\mathrm{SK} = h^{\mathbf{k}(\alpha,\mathbf{r},\mathbf{b})}, \qquad \mathrm{CT} = (M \cdot e(g,h)^{\alpha s}, g^{\mathbf{c}(\mathbf{s},\mathbf{b})}),$$

[4] This subtype only exists for CP-ABE, and not for key-policy ABE (Appendix A.2).

[5] Unbounded ABE places no bounds on the attribute sets associated with the keys, or on the policies associated with the ciphertexts. This includes the number of times that one attribute occurs.

where $g \in \mathbb{G}, h \in \mathbb{H}$ are two generators, e is a pairing $e \colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$ and $\mathbf{k}$ and $\mathbf{c}$ denote vectors over the variables $\alpha, \mathbf{r}, \mathbf{b}$ and $\mathbf{s}, \mathbf{b}$, such that each key component is of the form h^{k_i} and each ciphertext component is of the form g^{c_i}. In contrast, most multi-authority schemes include multiple elements in $\mathbb{G}_T$ in the ciphertexts, and mask the message M with e.g., $e(g, h)^{\tilde{s}}$. To capture such schemes, we generalize the definition of pair encodings.

Generalizing the symbolic property. The symbolic property considers the existence of some vectors and matrices such that, if all variables $\alpha, \mathbf{r}, \mathbf{s}$ and $\mathbf{b}$ in the polynomials of $\mathbf{k}$ and $\mathbf{c}$ are substituted by these vectors and matrices, the polynomials evaluate to 0. The symbolic property also needs to be generalized to match our generalized definition of pair encodings, which is complicated for two reasons. First, because the masking value may be different, we need to be able to find a more general way to simulate it in the security proofs than existing compilers currently do. Second, multi-authority ABE security models allow the corruption of authorities, which requires the challenger to share e.g., the master key α, with the attacker. In proofs based on the symbolic property, the master key cannot be simulated explicitly, and is canceled by other values instead to simulate the secret keys. To overcome these difficulties, we use program-and-cancel strategies for decentralized ABE [49,30] as inspiration. However, like [49,30], we prove decentralized schemes secure in the static-security model. This model does not only require the attacker to commit to the challenge policy, but also to the attribute sets that they are going to query.

2 Preliminaries

2.1 Notation

We use λ to denote the security parameter. A negligible function parametrized by λ is denoted as $\text{negl}(\lambda)$. If an element x is chosen uniformly at random from a finite set S, then we denote this as $x \in_R S$. If an element x is produced by running algorithm Alg, then we denote this as $x \leftarrow \text{Alg}$. We use $\mathbb{Z}_p = \{x \in \mathbb{Z} \mid 0 \leq x < p\}$ for the set of integers modulo p. 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, where $(\mathbf{A})_{i,j}$ denotes the entry of $\mathbf{A}$ in the i-th row and j-th column, and $(\mathbf{v})_i$ denotes the i-th entry of $\mathbf{v}$. We denote $a \ : \ \mathbf{A}$ to substitute variable a by a matrix or vector $\mathbf{A}$. We define $\mathbf{1}_{i,j}^{d_1 \times d_2} \in \mathbb{Z}_p^{d_1 \times d_2}$ as the matrix with 1 in the i-th row and j-th column, and 0 everywhere else, and similarly $\mathbf{1}_i^{d_1}$ and $\overline{\mathbf{1}}_i^{d_2}$ as the row and column vectors with 1 in the i-th entry and 0 everywhere else. If some algorithm yields no output or outputs an error message, then we use $\perp$ to indicate this.

2.2 Access structures

We represent access policies $\mathbb{A}$ by linear secret sharing scheme (LSSS) matrices, which support monotone span programs [17,38].

Definition 1 (Access structures represented by LSSS [38]). *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 share of secret* s *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. If* $\mathcal{S}$ *does not satisfy* $\mathbb{A}$*, there exists* $\mathbf{w} = (1, w_2, ..., w_{n_2}) \in \mathbb{Z}_p^{n_2}$ *such that* $\mathbf{A}_i \mathbf{w}^\intercal = 0$ *for all* $i \in \Upsilon$ *[17].*

2.3 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 (i) for all $a, b \in \mathbb{Z}_p$, it holds that $e(g^a, h^b) = e(g, h)^{ab}$ (bilinearity), and (ii) 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*. In practical instantiations, type-III pairings are used, meaning that no efficiently computable isomorphism exists between $\mathbb{G}$ and $\mathbb{H}$ [36]. For such pairings, the efficiency of $\mathbb{G}$ and $\mathbb{H}$ often differs by several factors [36,47]. Furthermore, we use the implicit representation used for group elements in [32]. Suppose $g' \in \mathbb{G}'$ is the generator of some group $\mathbb{G}' \in \{\mathbb{G}, \mathbb{H}, \mathbb{G}_T\}$, then we use $[x]_{\mathbb{G}'}$ to denote the element $(g')^x$.

2.4 Attribute-based encryption

Predicate family. A *predicate family* [11] 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 $\kappa = (p, \mathrm{par})$, where p is a natural number and par denote the rest of the entries.

Definition 2 (Attribute-based encryption (ABE) [4]). *An attribute-based 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:*

- $\mathrm{Setup}(\lambda, \mathrm{par}) \to (\mathrm{MPK}, \mathrm{MSK})$*: 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.*
- $\mathrm{KeyGen}(\mathrm{MSK}, y) \to \mathrm{SK}_y$*: On input the master secret key* MSK *and some* $y \in \mathcal{Y}_\kappa$*, this probabilistic algorithm generates a secret key* SK_y*.*
- $\mathrm{Encrypt}(\mathrm{MPK}, x, M) \to \mathrm{CT}_x$*: On input the master public key* MPK*, some* $x \in \mathcal{X}_\kappa$ *and message* M*, this probabilistic algorithm generates a ciphertext* CT_x*.*
- $\mathrm{Decrypt}(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x) \to M$*: 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 $P_\kappa(x, y) = 1$,

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

Ciphertext-policy ABE. A specific instance of ABE is ciphertext-policy ABE. In this type of ABE, the key predicate y is a set of attributes $\mathcal{S}$ over some universe of attributes $\mathcal{U}$, and the ciphertext predicate x is an access policy $\mathbb{A} = (\mathbf{A}, \rho)$, in this work represented as LSSS matrices (Definition 1).

Multi-authority ABE. In the multi-authority setting, the Setup is split in two algorithms: the GlobalSetup and the AuthoritySetup. The latter is run by each authority in the system. Furthermore, the security model allows the attacker to corrupt authorities. In Appendix A.1, the full definitions can be found.

2.5 Full security against chosen-plaintext attacks

Definition 3 (Full security against chosen-plaintext attacks (CPA) [4]). *We define the security game* IND-CPA(λ, par) *between challenger and attacker as follows:*

- ***Setup phase:*** *The challenger runs* Setup(λ) *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. In the co-selective security model, the attacker commits to all $y \in \mathcal{Y}_\kappa$ before the Setup phase. In the static security model, the attacker commits to $x^* \in \mathcal{X}_\kappa$ and all $y \in \mathcal{Y}_\kappa$ before the Setup phase.

2.6 The uber-assumption family

The security of many schemes, including those instantiated in the Agrawal-Chase framework [4], rely on q-type assumptions, which are complexity assumptions parametrized in one or more parameter. Many q-type assumptions can be captured in the uber-assumption framework by Boneh, Boyen and Goh [21,23]. In particular, they prove generic lower bounds on the complexity of any such q-type assumptions in the generic group model [51].

Definition 4 (The uber-assumption family [21,23]). *Let $e\colon \mathbb{G}\times\mathbb{H}\to\mathbb{G}_T$ be a pairing over three groups $\mathbb{G},\mathbb{H},\mathbb{G}_T$ of prime order p, and let $g\in\mathbb{G}, h\in\mathbb{H}$ be two generators. Let $n_{\mathbb{G}}, n_{\mathbb{H}}, n_{\mathbb{G}_T}, n_c \in \mathbb{N}$ be four positive integers. Suppose that, for all $\mathbb{G}'\in\{\mathbb{G},\mathbb{H},\mathbb{G}_T\}$, we have polynomials $\mathfrak{P}_{\mathbb{G}'}\in\mathbb{Z}_p[X_1,...,X_{n_c}]^{n_{\mathbb{G}'}}$. Let $\mathfrak{P}_T\in\mathbb{Z}_p[X_1,...,X_{n_c}]$ another polynomial. The challenger generates $\mathsf{x}_1,...,\mathsf{x}_{\mathsf{n_c}}\in_{\mathsf{R}}\mathbb{Z}_{\mathsf{p}}$, and outputs*

$$g^{\mathfrak{P}_{\mathbb{G}}(\mathsf{x}_1,...,\mathsf{x}_{\mathsf{n_c}})}, h^{\mathfrak{P}_{\mathbb{H}}(\mathsf{x}_1,...,\mathsf{x}_{\mathsf{n_c}})}, e(g,h)^{\mathfrak{P}_{\mathbb{G}_T}(\mathsf{x}_1,...,\mathsf{x}_{\mathsf{n_c}})}.$$

The challenger also flips a coin $\beta\in_R\mathbb{Z}_p$ and outputs $T\in_R\mathbb{G}_T$ if $\beta=0$ and $T=e(g,h)^{\mathfrak{P}_{\mathsf{T}}(\mathsf{x}_1,...,\mathsf{x}_{\mathsf{n_c}})}$ if $\beta=1$. The attacker outputs a guess β' for β. The advantage of the attacker is defined as $\mathsf{Adv}_{(n_{\mathbb{G}},n_{\mathbb{H}},n_{\mathbb{G}_T},n_c)\text{-DDH}} = |\Pr[\beta'=\beta]-\frac{1}{2}|$. The decisional $(n_{\mathbb{G}},n_{\mathbb{H}},n_{\mathbb{G}_T},n_c)$-Diffie-Hellman ($(n_{\mathbb{G}},n_{\mathbb{H}},n_{\mathbb{G}_T},n_c)$-DDH) assumption holds if all polynomial-time attackers have at most a negligible advantage, i.e.,

$$\mathsf{Adv}_{(n_{\mathbb{G}},n_{\mathbb{H}},n_{\mathbb{G}_T},n_c)\text{-DDH}} \leq \mathrm{negl}(\lambda).$$

Remark 1. We formulate the definition of the uber-assumption family in the type-III setting, i.e., in which the pairing is asymmetric. One can easily adapt the definition to cover symmetric pairings (where $\mathbb{G}=\mathbb{H}$) by setting $\mathfrak{P}_{\mathbb{G}}=\mathfrak{P}_{\mathbb{H}}$.

Boneh, Boyen and Goh show that, if $\mathfrak{P}_T$ is independent of $\mathfrak{P}_{\mathbb{G}_T}$ and all products of the polynomials in $\mathfrak{P}_{\mathbb{G}}$ with the polynomials in $\mathfrak{P}_{\mathbb{H}}$, the decisional $(n_{\mathbb{G}},n_{\mathbb{H}},n_{\mathbb{G}_T},n_c)$-Diffie-Hellman ($(n_{\mathbb{G}},n_{\mathbb{H}},n_{\mathbb{G}_T},n_c)$-DDH) assumption holds in the generic group model. We state Corollary 1 [23, §5.2] below.

Corollary 1 (Asymptotic lower bound for uber assumptions [23]). *Let p, $\mathfrak{P}_{\mathbb{G}'}$ and $\mathfrak{P}_T$ be as in Definition 4. Suppose $\mathfrak{P}_T$ is independent of $\mathfrak{P}_{\mathbb{G}_T}$ and all products of the polynomials in $\mathfrak{P}_{\mathbb{G}}$ with the polynomials in $\mathfrak{P}_{\mathbb{H}}$. Let $\deg_{\mathbb{G}'}$ be the maximum degree of the polynomials in $\mathfrak{P}_{\mathbb{G}'}$, let $\deg_T$ be the degree of $\mathfrak{P}_T$, and set $\deg=\max(\{\deg_{\mathbb{G}_T},\deg_T,\deg_{\mathbb{G}}+\deg_{\mathbb{H}}\})$. Then, any attacker $\mathcal{A}$ that can solve the decisional $(n_{\mathbb{G}},n_{\mathbb{H}},n_{\mathbb{G}_T},n_c)$-Diffie-Hellman problem in the generic group model must take time at least $\mathcal{O}(\sqrt{p/\deg}-n_c)$.*

3 Pair encoding schemes

To support the aforementioned practical extensions, we extend the definitions of pair encoding schemes and their associated security definition: the symbolic property. Intuitively, the most fine-grained definition [4] of pair encoding schemes (see Definition 5) considers schemes of the form

$$\mathrm{SK} = (h^{\mathbf{r}}, h^{\mathbf{k}(\alpha,\mathbf{r},\hat{\mathbf{r}},\mathbf{b},y)}), \qquad \mathrm{CT} = (M \cdot e(g,h)^{\alpha s}, g^{\mathbf{s}} = g^{(s,s_1,...,)}, g^{\mathbf{c}(\mathbf{s},\hat{\mathbf{s}},\mathbf{b},x)}),$$

where $\mathbf{r}, \mathbf{s}, \hat{\mathbf{r}}, \hat{\mathbf{s}}, \mathbf{k}, \mathbf{c}$ are vectors. Specifically, α is called the master-key variable, $\mathbf{r}$ and $\mathbf{s}$ are called the non-lone key and ciphertext variables, respectively, $\hat{\mathbf{r}}$ and $\hat{\mathbf{s}}$ are called the lone key and ciphertext variables, respectively, and $\mathbf{k}$ and $\mathbf{c}$ are the key and ciphertext polynomials, respectively. In particular, we distinguish between lone and non-lone variables to separate variables that occur in combination with a common variable (i.e., which are "non-lone") and those do not (i.e., which are "lone"). Roughly, the symbolic property considers the existence of matrices (for variables $\mathbf{b}$) and vectors (for the other variables) such that substituting the variables in the key and ciphertext polynomials with these matrices and vectors yields all-zero vectors upon evaluation (see Definition 6).

In this section, we first give the prior formulation of pair encoding schemes and the symbolic property, and then show how they can be generalized.

3.1 Prior formulation of pair encoding schemes

Pair encoding schemes. Throughout the years, the notion of pair encoding schemes has been defined and refined [11,12,2,4]. We provide the most refined definition below.

Definition 5 (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.*

- $\mathrm{Param}(\mathrm{par}) \to (n, \mathbf{b})$*: On input* par*, the algorithm outputs $n \in \mathbb{N}$ that specifies the number of common variables, which are denoted as $\mathbf{b} = (b_1, ..., b_n)$.*
- $\mathrm{EncKey}(y, p) \to (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \mathbf{b}, y))$*: On input $p \in \mathbb{N}$ and $y \in \mathcal{Y}_\kappa$, this algorithm outputs a vector of polynomials $\mathbf{k} = (k_1, ..., k_{m_3})$, with $m_3 \in \mathbb{N}$, 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 $\delta_i, \delta_{i,j}, \delta_{i,j,k} \in \mathbb{Z}_p$.
- $\mathrm{EncCt}(x, p) \to (w_1, w_2, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, x))$*: On input $p \in \mathbb{N}$ and $x \in \mathcal{X}_\kappa$, this algorithm outputs a vector of polynomials $\mathbf{c} = (c_1, ..., c_{w_3})$, with $w_3 \in \mathbb{N}$, defined over non-lone variables $\mathbf{s} = (s, s_1, s_2, ..., s_{w_1})$ 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 $\eta_{i,j}, \eta_{i,j,k} \in \mathbb{Z}_p$.

- $\mathrm{Pair}(x, y, p) \rightarrow (\mathbf{E}, \overline{\mathbf{E}})$*: 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, if 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 $\mathbf{s}\mathbf{E}\mathbf{k}^\intercal + \mathbf{c}\overline{\mathbf{E}}\mathbf{r}^\intercal = \alpha s$.

Symbolic security property. The symbolic security property is a powerful security notion for pair encoding schemes that is purely algebraic. Roughly, the notions of selective and co-selective symbolic security are based on the classical security notions of selective and co-selective security for ABE (Definition 3). Recall that, in these models, the attacker commits to the challenge access policy (resp. set of attributes). This is used in "program-and-cancel" proofs [57,48], in which the challenger embeds the policy (resp. set) in the public keys. In the simulation of the secret keys and challenge ciphertext, the components are programmed in a specific way, using that the set does not satisfy the policy (resp. policy is not satisfied by the set). Typically, the components that cannot be programmed are canceled by other non-programmable components. In the AC17 framework, this "programming" is replaced by "substitution", and the "canceling" is replaced by "evaluating to 0".

Definition 6 (Symbolic security property (Sym-Prop) [4]). *A pair encoding scheme $\Gamma = (\mathrm{Param}, \mathrm{EncKey}, \mathrm{EncCt}, \mathrm{Pair})$ for a predicate family $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \rightarrow \{0,1\}$ 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 that*

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

such that $\langle \mathbf{s}_0, \mathbf{a} \rangle \neq 0$, and if we substitute

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

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 $\mathrm{EncB}, \mathrm{EncR}, \mathrm{EncS}$ *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 $d_2', d_2'' \leq d_2$.

3.2 How the symbolic property and selective security are related

As mentioned, the selective symbolic property and selective security are strongly related in their approaches. More specifically, the evaluation of the polynomials

k_i and c_i to 0 after substituting the variables by the vectors and matrices is closely related to the "canceling" part of the "program-and-cancel" strategy used in selective-security proofs. The "programming" part of this proof strategy is related to the complexity assumption that is used in the reduction. Concretely, various input parameters to this complexity assumption are used to program the key and ciphertext components associated with the common and non-lone variables. They are programmed in such a way that the $e(g,h)^{\alpha s}$ part of the scheme can be programmed by the "testing value" of the complexity assumption. For example, consider the keys and ciphertexts of the Boneh-Boyen [20] scheme:

$$\mathrm{SK} = (h^{\alpha + r(b_0 + y b_1)}, h^r), \qquad \mathrm{CT} = (M \cdot e(g,h)^{\alpha s}, g^{s(b_0 + x b_1)}, g^s),$$

where x and y are identities, for which the associated PES is

$$\mathbf{k}(\alpha, r, (b_0, b_1)) = \alpha + r(b_0 + y b_1), \qquad \mathbf{c}(s, (b_0, b_1)) = s(b_0 + x b_1).$$

It satisfies the selective symbolic property, because for $x \neq y$, we can set

$$\mathbf{a} = 1, \qquad \mathbf{r} = \frac{1}{x-y}, \qquad \mathbf{b}_0 = -x, \qquad \mathbf{b}_1 = 1, \qquad \mathbf{s} = 1.$$

Analogously, in the selective security proof, we can make a reduction to the decisional bilinear Diffie-Hellman (DBDH) assumption, i.e., given $g^{\mathsf{x}}, h^{\mathsf{x}}, g^{\mathsf{y}}, h^{\mathsf{y}}, g^{\mathsf{z}}, h^{\mathsf{z}}$, determine whether some testing value T is equal to $e(g,h)^{\mathsf{xyz}}$ or not. We can program the master public key, and the secret key and ciphertext components associated with the non-lone variables in a similar way as in the symbolic property as follows:

$$e(g,h)^{\alpha} = e(g,h)^{\bar{\alpha}} \cdot e(g,h)^{\mathbf{a}\mathsf{xz}}, \qquad g^{b_0} = g^{\bar{b}_0} \cdot g^{\mathbf{b}_0\mathsf{z}}, \qquad g^{b_1} = g^{\bar{b}_1} \cdot g^{\mathbf{b}_1\mathsf{z}},$$
$$h^r = h^{\bar{r}} \cdot h^{\mathbf{r}\mathsf{x}}, \qquad g^s = g^{\bar{s}} \cdot g^{\mathbf{s}\mathsf{y}}.$$

Then, the secret key and ciphertext components associated with the polynomials can be programmed by using the inputs to the DBDH assumption and using that the polynomials evaluate to 0 for those inputs that are not part of the assumption. For example, the key component is simulated as follows:

$$h^{\alpha + r(b_0 + y b_1)} = h^{\bar{\alpha} + \mathbf{a}\mathsf{xz} + (\bar{r} + \mathbf{r}\mathsf{x})(\bar{b}_0 + \mathbf{b}_0\mathsf{z} + y(\bar{b}_1 + \mathbf{b}_1\mathsf{z}))}$$
$$= \underbrace{h^{\bar{\alpha} + \bar{r}(\bar{b}_0 + \mathbf{b}_0\mathsf{z} + y(\bar{b}_1 + \mathbf{b}_1\mathsf{z})) + \mathbf{r}\mathsf{x}(\bar{b}_0 + y\bar{b}_1)}}_{\Delta_1} \cdot h^{\mathbf{a}\mathsf{xz} + \mathbf{r}\mathsf{x}(\mathbf{b}_0\mathsf{z} + y\mathbf{b}_1\mathsf{z})} = \Delta_1 \cdot \underbrace{h^{(\mathbf{a} + \mathbf{r}(\mathbf{b}_0 + y\mathbf{b}_1))\mathsf{xz}}}_{=1},$$

such that Δ_1 can be programmed from $\bar{\alpha}, \bar{r}, \bar{b}_0, \bar{b}_1$ and the inputs to the DBDH assumption, and the remainder associated with h^{xz} (which cannot be part of the assumption) cancels because the polynomial $\alpha + r(b_0 + y b_1)$ evaluates to 0 when α, r, b_0, b_1 are substituted by $\mathbf{a}, \mathbf{r}, \mathbf{b}_0, \mathbf{b}_1$. Lastly, the blinding value is set to $e(g,h)^{\alpha s} = T \cdot e(g,h)^{\bar{\alpha} s} \cdot e(g,h)^{\alpha \bar{s}} \cdot e(g,h)^{\bar{\alpha}\bar{s}}$.

For our compiler, we generalize this approach. Roughly, we associate the public key variables with (parallel instances of) z, all lone key variables with (parallel instances of) xz, and all non-lone key variables with (parallel instances of) x, so that the key polynomials are associated with (parallel instances of) xz. Similarly, we associate the lone ciphertext variables with (parallel instances of) yz and the non-lone ciphertext variables with (parallel instances of) y, so that the ciphertext polynomials are associated with (parallel instances of) yz. Finally, the blinding value should be associated with xyz, so in the case that this is αs (as in the definition of PES), we require that α and s only use xz and y (and no parallel instances) of the inputs to the complexity assumption. Note that these parallel instances are related to the choices of d_1 and d_2, e.g., we require d_1 parallel instances of y to embed each entry of the substitution vector for a non-lone ciphertext variable. We show in Section 4 how to create such parallel instances in such a way that the assumption holds generically, while the parts of the keys and ciphertexts that do not cancel can be programmed as required.

3.3 Generalizing the definition of pair encoding schemes

In order to cover a larger class of schemes, we also give a more general definition of pair encoding schemes. Notably, decentralized schemes such as [41,49] cannot be covered by Definition 5. Consequently, we cannot benefit from the generic security as well as the generic conversion techniques that the pair encodings framework provides. Regardless, the proof techniques in [49] are strikingly similar to the proof techniques in works in the single-authority setting [57,48]. We use this observation to define our more general definitions of pair encoding schemes and the symbolic property. Concretely, for the definition of pair encodings, we extend the master key α and the associated encodings. We also explicitly include ciphertext polynomials that will be instantiated in the target group, and write the blinding value used to mask M in the scheme as a polynomial.

Definition 7 (Generalized pair encoding schemes (GPES)). *A generalized 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.*

- $\mathrm{Param}(\mathrm{par}) \to (n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b})$*: On input* par*, the algorithm outputs $n_\alpha, n_b \in \mathbb{N}$ that specify the number of master key variables and common variables, respectively, which are denoted as $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_{n_\alpha})$ and $\mathbf{b} = (b_1, ..., b_{n_b})$, respectively.*
- $\mathrm{EncKey}(y, p) \to (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, y))$*: 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*

$$k_i = \sum_{j \in [n_\alpha]} \delta_{i,j} \alpha_j + \sum_{j \in [m_2]} \hat{\delta}_{i,j} \hat{r}_j + \sum_{j \in [m_1], k \in [n_b]} \delta_{i,j,k} r_j b_k,$$

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

- $\mathrm{EncCt}(x,p) \rightarrow (w_1, w_2, w_2', c_M, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, x), \mathbf{c}'(\mathbf{s}, \tilde{\mathbf{s}}, \boldsymbol{\alpha}, x))$*: On input* $p \in \mathbb{N}$ *and* $x \in \mathcal{X}_\kappa$*, this algorithm outputs a blinding variable* c_M *and two vectors of polynomials* $\mathbf{c} = (c_1, ..., c_{w_3})$ *and* $\mathbf{c}' = (c_1', ..., c_{w_4}')$ *defined over non-lone variables* $\mathbf{s} = (s, s_1, s_2, ..., s_{w_1})$*, lone variables* $\hat{\mathbf{s}} = (\hat{s}_1, ..., \hat{s}_{w_2})$ *and special lone variables* $\tilde{\mathbf{s}} = (\tilde{s}_1, ..., \tilde{s}_{w_2'})$*. Specifically, the polynomial* c_i *is expressed as*

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

for all $i \in [w_3]$*, where* $\eta_{i,j}, \eta_{i,j,k} \in \mathbb{Z}_p$*, the polynomial* c_i' *is expressed as*

$$c_i' = \sum_{j \in [n_\alpha], j' \in \overline{[w_1]}} \eta_{i,j,j'}' \alpha_j s_{j'} + \sum_{j \in [w_2']} \hat{\eta}_{i,j}' \tilde{s}_j,$$

for all $i \in [w_4]$*, where* $\eta_{i,j,j'}', \hat{\eta}_{i,j}' \in \mathbb{Z}_p$*, and the variable* c_M *is expressed as*

$$c_M = \sum_{j \in [w_2']} \zeta_j \tilde{s}_j + \sum_{j \in [n_\alpha], j' \in \overline{[w_1]}} \zeta_{j,j'} \alpha_j s_{j'},$$

where $\zeta_j, \zeta_{j,j'} \in \mathbb{Z}_p$*.*
- $\mathrm{Pair}(x,y,p) \rightarrow (\mathbf{e}, \mathbf{E}, \overline{\mathbf{E}})$*: On input* p*,* x*, and* y*, this algorithm outputs a vector* $\mathbf{e} \in \mathbb{Z}_p^{w_4}$ *and 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* $\mathbf{e}\mathbf{c}'^\intercal + \mathbf{s}\mathbf{E}\mathbf{k}^\intercal + \mathbf{c}\overline{\mathbf{E}}\mathbf{r}^\intercal = c_M$*.*

3.4 Special symbolic property for GPES

To generalize the symbolic property, we also need to find proper substitutions for the new master-key variables and the ciphertext encodings $\mathbf{c}'$. In addition, we need to be able to account for static corruption of certain variables.

For the master-key variables, we first observe that these occur as lone variables in the key encodings and as common variables in the ciphertext encodings $\mathbf{c}'$, meaning that we only have to be able to multiply them with non-lone ciphertext variables, and it is thus sufficient to substitute with vectors (rather than matrices, like the common variables). Because the non-lone ciphertext variables are substituted by vectors of length d_1, we therefore also substitute the master-key variables by vectors of length d_1, so that their inner product yields an integer. In addition to products of master-key variables and non-lone variables, the ciphertext encodings consist of special lone variables, which therefore also need to be substituted by integers.

To ensure that we can replace $e(g,h)^{c_M}$ with the testing value T, we additionally require that all master-key variables and non-lone ciphertext variables that occur in c_M are equal to $\mathbf{1}_1^{d_1}$. In this way, the products of the simulated components do not yield any parallel instances of xyz.

Finally, to support corruption, we need to ensure that none of the corrupted secret values (such as those related to the lone key variables) contains any input parameters to the complexity assumption. We ensure this by setting their corresponding substitution vectors/matrices to all-zero. Putting this together, this yields the following definition.

Definition 8 (Special symbolic property for GPES (Spec-Sym-Prop-G)). *A GPES* $\Gamma = (\mathrm{Param}, \mathrm{EncKey}, \mathrm{EncCt}, \mathrm{Pair})$ *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* $\kappa = (p, \mathrm{par})$*, and* $x \in \mathcal{X}_\kappa$ *and* $y \in \mathcal{Y}_\kappa$ *with* $P_\kappa(x, y) = 0$*, and optionally, there exist* $\mathfrak{a} \subsetneq [n_\alpha]$*,* $\mathfrak{b} \subsetneq [n_b]$ *(which we call corruptable variables), such that we have that*

- $\mathrm{EncB}(x, \mathfrak{a}, \mathfrak{b}) \to \mathbf{a}_1, ..., \mathbf{a}_{n_\alpha} \in \mathbb{Z}_p^{d_1}, \mathbf{B}_1, ..., \mathbf{B}_{n_b} \in \mathbb{Z}_p^{d_1 \times d_2}$*;*
- $\mathrm{EncR}(x, y) \to \mathbf{r}_1, ..., \mathbf{r}_{m_1} \in \mathbb{Z}_p^{d_2}, \hat{\mathbf{r}}_1, ..., \hat{\mathbf{r}}_{m_2} \in \mathbb{Z}_p^{d_1}$*;*
- $\mathrm{EncS}(x) \to \mathbf{s}_0, ..., \mathbf{s}_{w_1} \in \mathbb{Z}_p^{d_1}, \hat{\mathbf{s}}_1, ..., \hat{\mathbf{s}}_{w_2} \in \mathbb{Z}_p^{d_2}, \tilde{\mathbf{s}}_1, ..., \tilde{\mathbf{s}}_{w'_2} \in \mathbb{Z}_p$*;*

such that, if we substitute

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

for $i \in [w_1], i' \in [w_2], i'' \in [w'_2], j \in [n_b], k \in [m_1], k' \in [m_2], l \in [n_\alpha]$ *in all the polynomials of* $\mathbf{k}$*,* $\mathbf{c}$ *and* $\mathbf{c}'$ *(output by* EncKey *and* EncCt*, respectively), they evaluate to* $\mathbf{0}$*. Furthermore,*

- *for all* $j \in [n_\alpha] \setminus \mathfrak{a}, j' \in \overline{[w_1]}$ *with* $\zeta_{j,j'} \neq 0$*, we have that* $\mathbf{a}_j = \mathbf{s}_{j'} = \mathbf{1}_1^{d_1}$*;*
- *for* $j \in [w'_2]$ *with* $\zeta_j \neq 0$*, we have that* $\tilde{\mathbf{s}}_j = 1$*;*
- *for* $j \in \mathfrak{a}$*, we have* $\mathbf{a}_j = \mathbf{0}^{d_1}$*;*
- *and for* $j \in \mathfrak{b}$*, we have that* $\mathbf{B}_j = \mathbf{0}^{d_1 \times d_2}$*.*

Similarly, a GPES satisfies the special (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 GPES satisfies the special (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* $d'_2, d''_2 \leq d_2$*.*

Remark 2. PESs can be captured under our definition of generalized PES. That is, we can simply set $n_\alpha = 1$, $w_2, w_4 = 0$ and $C_M = \alpha s$. Furthermore, most existing PESs (e.g., [4,13]) satisfy the special (d_1, d_2)-selective symbolic property, because they satisfy the symbolic property, and $\mathbf{a} = \mathbf{s} = \mathbf{1}_1^{d_1}$. Therefore, these can be securely instantiated in the selective-security setting with our compiler.

3.5 Distribution of the encodings

We also give an explicit definition for the distribution of the encodings over the two source groups $\mathbb{G}$ and $\mathbb{H}$, and the target group $\mathbb{G}_T$ when they are instantiated in our new compiler. Such a distribution should ensure that the correctness of the GPES is preserved, such that the correctness of the ABE scheme is also guaranteed. In particular, for the correctness of the decryption algorithm, we require that each pair of key and ciphertext encodings that needs to be paired has one encoding in $\mathbb{G}$ and one in $\mathbb{H}$. Furthermore, to ensure that encryption can be performed correctly, the master public keys required in computing a ciphertext encoding element need to be in the same group.

Definition 9 (Distribution of the encodings over $\mathbb{G}$, $\mathbb{H}$ and $\mathbb{G}_T$). *Let $\Gamma = (\mathrm{Param}, \mathrm{EncKey}, \mathrm{EncCt}, \mathrm{Pair})$ be a GPES for a predicate family $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ and let $\mathbb{G}$, $\mathbb{H}$ and $\mathbb{G}_T$ be three groups. Let $\mathcal{E}$ denote the set of possible encodings and non-lone variables that can be sampled with* Param, EncKey *and* EncCt*, and let $\mathcal{E}' \subseteq \mathcal{E}$ denote its subset containing the master key variables $\boldsymbol{\alpha}$ and ciphertext encodings $\mathbf{c}'$. Then, we define $\mathfrak{D} \colon \mathcal{E} \to \{\mathbb{G}, \mathbb{H}, \mathbb{G}_T\}$ to be the distribution of Γ over $\mathbb{G}$, $\mathbb{H}$ and $\mathbb{G}_T$ such that the correctness of the encoding is preserved. This is the case, if 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*

- *$\mathfrak{D}(\mathcal{E}') = \{\mathbb{G}_T\}$, and $\mathfrak{D}(\mathcal{E} \setminus \mathcal{E}') = \{\mathbb{G}, \mathbb{H}\}$;*
- *for all $i \in [m_3]$, $j \in \overline{[w_1]}$, if $\mathfrak{D}(k_i) = \mathfrak{D}(s_j)$, then $\mathbf{E}_{j,i} = 0$;*
- *for all $i \in [w_3]$, $j \in \overline{[m_1]}$, if $\mathfrak{D}(c_i) = \mathfrak{D}(r_j)$, then $\overline{\mathbf{E}}_{i,j} = 0$;*
- *for all $k \in [n_b]$ for which there exist some $i \in [w_3], j \in \overline{[w_1]}$ with $\eta_{i,j,k} \neq 0$, we have $\mathfrak{D}(b_k) = \mathfrak{D}(c_i)$.*

3.6 Full-domain hashes and random oracles

Sometimes, some of the variables are generated implicitly by a full-domain hash (FDH). For example, this is done to support large universes (see e.g., [57,3]) or to link the keys together in decentralized schemes (see e.g., [41,49]). Instead of generating e.g., g^b in the Setup and including it in the master public key, it is generated by the hash. In this way, the master public key only needs to contain a description of the hash, and then, any parameter generated by the hash can be generated once it is needed. Our compiler and proof can be easily support the use of full-domain hashes. In that case, the security proof requires the hashes to be modeled as random oracles. In particular, the random oracles answer the queries exactly in the way that it does in a proof where the variable is not generated by an FDH. To capture such random oracle queries in the security proof, we also define a function $\mathcal{F}$ that maps each encoding variable to a natural number.

Definition 10 (FDH-generated encoding variables). *Let $\Gamma = (\mathrm{Param}, \mathrm{EncKey}, \mathrm{EncCt}, \mathrm{Pair})$ be a GPES for a predicate family $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$. Let $\mathcal{E}$ denote the set of possible encodings and non-lone variables that can be sampled with* Param, EncKey *and* EncCt*. Then, we define $\mathcal{F} \colon \mathcal{E} \to \mathbb{N}$ to be the mapping that assigns whether the encoding variables are generated by an FDH or not. If not, then the encoding variable is mapped to 0. Otherwise, it is mapped to any integer larger than 0. When the FDH is instantiated, it expects the index of the encoding variable as input, e.g., if $\mathcal{F}(b_{\mathrm{att}}) = 1$, then $\mathcal{H}_1$ expects* att *as input in the scheme, and outputs $[b_{\mathrm{att}}]_{\mathfrak{D}(b_{\mathrm{att}})}$.*

Furthermore, to ensure correctness of the scheme, we require the distribution over the two source groups to be such that, for any common variable b_k that is provided implicitly by a hash, and each associated encoding k_i and c_i, it holds that they are placed in the same group. Similarly, we can define such a restriction for the other variables. Furthermore, if a non-lone variable and a common variable occur together in a product in one of the polynomials, then it cannot be the case that both are generated by an FDH. (It is possible to generate at most one with an FDH, by computing, e.g., $\mathcal{H}(\text{att})^r$ or $\mathcal{H}(\text{GID})^{b_{\text{att}}}$, but not both.) We formalize these restrictions as follows.

Definition 11 (Correctness of variables generated by an FDH). *Let $\mathfrak{D}$ be as in Definition 9. Then, for any common variable b_k with $\mathcal{F} > 0$ (i.e., generated implicitly by the full-domain hash), it holds that:*

- *For all $i \in [m_3]$, if $\mathfrak{D}(k_i) \neq \mathfrak{D}(b_k)$, then $\delta_{i,j,k} = 0$ for all $j \in \overline{[m_1]}$;*
- *For all $i \in [w_3]$, if $\mathfrak{D}(c_i) \neq \mathfrak{D}(b_k)$, then $\eta_{i,j,k} = 0$ for all $j \in \overline{[w_1]}$.*

For any non-lone variable r_j or s_j with $\mathcal{F}(r_j), \mathcal{F}(s_j) > 0$, it holds that:

- *For all $i \in [m_3]$, if $\mathfrak{D}(k_i) \neq \mathfrak{D}(r_j)$, then $\delta_{i,j,k} = 0$ for all $k \in [n]$;*
- *For all $i \in [w_3]$, if $\mathfrak{D}(c_i) \neq \mathfrak{D}(s_j)$, then $\eta_{i,j,k} = 0$ for all $k \in [n]$;*
- *For all $i \in [m_3], k \in [n]$, if $\delta_{i,j,k} \neq 0$, then $\mathcal{F}(b_k) = 0$;*
- *For all $i \in [w_3], k \in [n]$, if $\eta_{i,j,k} \neq 0$, then $\mathcal{F}(b_k) = 0$.*

Furthermore, for each $i \in \mathbb{N}$ with $i > 0$, we require that all the encodings that are mapped to it, i.e., $\mathcal{F}^{-1}(i)$, are either all common variables, or all non-lone key variables, or all non-lone ciphertext variables.

3.7 Our complexity assumption

The last ingredient to our compiler is the complexity assumption. The assumption that we use to prove security generically is loosely based on the q-type assumptions used in works that prove selective security, e.g., [48, §A]. Roughly, this assumption creates several parallel instances of an assumption similar to the DBDH assumption, augmented with some additional inputs.

Definition 12 (The (d_1, d_2)-parallel DBDH assumption). *Let λ be the security parameter. Let $e\colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$ be a pairing over three groups $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ of prime order p, and let $g \in \mathbb{G}, h \in \mathbb{H}$ be two generators. The challenger generates $\mathsf{x}, \mathsf{y}, \mathsf{z}, \mathsf{c_i}, \mathsf{c'_j} \in_R \mathbb{Z}_p$ for all $\mathsf{i} \in [2, d_1], \mathsf{j} \in [2, d_2]$, sets $\mathsf{c_1} = \mathsf{c'_1} = 1$ and outputs for all $\mathbb{G}' \in \{\mathbb{G}, \mathbb{H}\}$:*

$$[\mathsf{xc_i}]_{\mathbb{G}'}\text{, for all } \mathsf{i} \in [d_1] \qquad \left[\frac{\mathsf{xzc_i}}{\mathsf{c_{i'}c'_j}}\right]_{\mathbb{G}'}\text{, for all } \mathsf{i}, \mathsf{i'} \in [d_1], \mathsf{i} \neq \mathsf{i'}, \mathsf{j} \in [d_2]$$

$$\left[\mathsf{yc'_j}\right]_{\mathbb{G}'}\text{, for all } \mathsf{j} \in [d_2] \qquad \left[\frac{\mathsf{yzc'_j}}{\mathsf{c_ic'_{j'}}}\right]_{\mathbb{G}'}\text{, for all } \mathsf{i} \in [d_1], \mathsf{j}, \mathsf{j'} \in [d_2], \mathsf{j} \neq \mathsf{j'}$$

$$\left[\frac{\mathsf{z}}{\mathsf{c_ic'_j}}\right]_{\mathbb{G}'}\text{, for all } \mathsf{i} \in [d_1], \mathsf{j} \in [d_2].$$

By setting $\mathsf{c}_1 = \mathsf{c}'_1 = 1$*, we also have that* $[\mathsf{x}]_{\mathbb{G}'}, [\mathsf{y}]_{\mathbb{G}'}, [\mathsf{z}]_{\mathbb{G}'}$ *are included in these terms. The challenger also flips a coin* $\beta \in_R \mathbb{Z}_p$ *and outputs* $T \in_R \mathbb{G}_T$ *if* $\beta = 0$ *and* $T = e(g,h)^{\mathsf{xyz}}$ *if* $\beta = 1$*. The attacker outputs a guess* β' *for* β*. The advantage of the attacker is defined as* $\mathsf{Adv}_{(d_1,d_2)\text{-pDBDH}} = |\Pr[\beta' = \beta] - \frac{1}{2}|$*. The* (d_1, d_2)*-parallel DBDH assumption (*(d_1, d_2)*-pDBDH) holds if all polynomial-time attackers have at most a negligible advantage, i.e.,* $\mathsf{Adv}_{(d_1,d_2)\text{-pDBDH}} \leq \text{negl}(\lambda)$*.*

We prove the following lemma in Appendix B.

Lemma 1. *The* (d_1, d_2)*-parallel DBDH assumption holds in the GGM.*

Remark 3. Interestingly, for $d_1 = d_2 = 1$, the (d_1, d_2)-parallel DBDH assumption is equivalent to the DBDH assumption. An advantage of this is that, if the GPES is such that the special selective symbolic property holds for $d_1 = d_2 = 1$, we automatically obtain an instantiation whose security relies on DBDH (see, e.g., the scheme in Appendix E.2). In contrast, the q-type assumption on which the Agrawal-Chase compiler relies does not satisfy this property.

4 Our generic compiler

Our new generic compiler instantiates the GPES into the pairing-friendly groups $\mathbb{G}, \mathbb{H}$ and $\mathbb{G}_T$ in the most obvious way. Roughly, the master public key, the secret keys and the ciphertexts have the following form:

$$\text{MPK} = (e(g,h)^{\boldsymbol{\alpha}}, (g')^{\mathbf{b}}), \qquad \text{SK} = (h^{\mathbf{r}}, h^{\mathbf{k}(\mathbf{r},\hat{\mathbf{r}},\boldsymbol{\alpha},\mathbf{b},y)}),$$
$$\text{CT} = (M \cdot e(g,h)^{c_M}, (g')^{\mathbf{c}(\mathbf{s},\hat{\mathbf{s}},\mathbf{b},x)}, e(g,h)^{\mathbf{c}'(\mathbf{s},\hat{\mathbf{s}},\boldsymbol{\alpha},x)}),$$

(where g' indicates that either $g' = g$ or $g' = h$ for each entry of the vector in the exponent). More concretely, we define our generic compiler as follows.

Definition 13 (Our generic compiler). *Let* $\Gamma = (\text{Param}, \text{EncKey}, \text{EncCt}, \text{Pair})$ *be a GPES for a predicate family* $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$*, let* $e \colon \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$ *be a pairing over three groups* $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ *of prime order* p*, let* $g \in \mathbb{G}, h \in \mathbb{H}$ *be two generators and let* $\mathfrak{D} \colon \mathcal{E} \to \{\mathbb{G}, \mathbb{H}\}$ *be a distribution of the encodings* F *the two source groups* $\mathbb{G}$ *and* $\mathbb{H}$*, and let* $\mathcal{F} \colon \mathcal{E} \to \mathbb{N}$ *be the mapping that maps the encoding variables to natural numbers. For each* $i \in \mathcal{F}(\mathcal{E}) \setminus \{0\}$*, let* $\mathcal{H}_i \colon \{0,1\}^* \to \mathbb{G}'$ *denote a full-domain hash modeled as a random oracle, where* $\mathbb{G}' = \mathfrak{D}(\mathcal{F}^{-1}(i))$ *is the group to which the associated encoding variables are mapped. Then, we define the ABE scheme for predicate family* P_κ *as follows:*

- $\text{Setup}(\lambda, \text{par}) \to (\text{MPK}, \text{MSK})$*: On input the security parameter* λ *and parameters* par*, this algorithm generates* $(n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b}) \leftarrow \text{Param}(\text{par})$*, sets* $\text{MSK} = (\boldsymbol{\alpha}, \{b_i \mid i \in [n_b] \wedge \mathcal{F}(b_i) = 0\})$ *as the master secret key, and outputs*

$$\text{MPK} = (A = \{[\alpha_i]_{\mathbb{G}_T}\}_{i \in [n_\alpha]}, \{[b_i]_{\mathfrak{D}(b_i)} \mid i \in [n_b] \wedge \mathcal{F}(b_i) = 0\})$$

as the master public key. The global parameters are $p, e, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, g, h$*.*

- KeyGen$(\mathrm{MSK}, y) \rightarrow \mathrm{SK}_y$: *On input the master secret key* MSK *and some* $y \in \mathcal{Y}_\kappa$*, this algorithm generates* $(m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, y)) \leftarrow \mathrm{EncKey}(y, p)$*, and outputs the secret key* SK_y *as*

$$\mathrm{SK}_y = (y, \{[r_j]_{\mathfrak{D}(r_j)} \mid j \in [m_1] \wedge \mathcal{F}(r_j) = 0\}, \{[k_i]_{\mathfrak{D}(k_i)}\}_{i \in [m_3]})$$

- Encrypt$(\mathrm{MPK}, x, M) \rightarrow \mathrm{CT}_x$: *On input the master public key* MPK*, some* $x \in \mathcal{X}_\kappa$ *and message* $M \in \mathbb{G}_T$*, this algorithm generates* $(w_1, w_2, w'_2, c_M, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, x), \mathbf{c}'(\mathbf{s}, \tilde{\mathbf{s}}, \boldsymbol{\alpha}, x)) \leftarrow \mathrm{EncCt}(x, p)$*, and outputs the ciphertext* CT_x *as*

$$\mathrm{CT}_x = (x, M \cdot e(g,h)^{c_M},$$
$$[s]_{\mathfrak{D}(s)}, \{[s_j]_{\mathfrak{D}(s_j)} \mid j \in \overline{[w_1]} \wedge \mathcal{F}(s_j) = 0\}, \{[c_i]_{\mathfrak{D}(c_i)}\}_{i \in [w_3]}, \{[c'_i]_{\mathbb{G}_T}\}_{i \in [w_4]}).$$

- Decrypt$(\mathrm{MPK}, \mathrm{SK}_y, \mathrm{CT}_x) \rightarrow M$: *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 first obtains* $(\mathbf{E}, \overline{\mathbf{E}}) \leftarrow \mathrm{Pair}(x, y, p)$*, sets*

$$\begin{aligned} \mathcal{P} = \{&(s_j, k_i, \mathbf{E}_{j,i}) \mid i \in [m_3], j \in \overline{[w_1]}, \mathbf{E}_{j,i} \neq 0 \wedge \mathfrak{D}(s_j) = \mathbb{G}\} \\ \cup\ \{&(k_i, s_j, \mathbf{E}_{j,i}) \mid i \in [m_3], j \in \overline{[w_1]}, \mathbf{E}_{j,i} \neq 0 \wedge \mathfrak{D}(s_j) = \mathbb{H}\} \\ \cup\ \{&(r_j, c_i, \overline{\mathbf{E}}_{i,j}) \mid i \in [w_3], j \in \overline{[m_1]}, \overline{\mathbf{E}}_{i,j} \neq 0 \wedge \mathfrak{D}(r_j) = \mathbb{G}\} \\ \cup\ \{&(c_i, r_j, \overline{\mathbf{E}}_{i,j}) \mid i \in [w_3], j \in \overline{[m_1]}, \overline{\mathbf{E}}_{i,j} \neq 0 \wedge \mathfrak{D}(r_j) = \mathbb{H}\}, \end{aligned}$$

and then retrieves

$$\prod_{i \in [n_\alpha]} [c'_i]_{\mathbb{G}_T}^{\mathbf{e}_i} \prod_{(\mathfrak{l}, \mathfrak{r}, \mathfrak{e}) \in \mathcal{P}} e([\mathfrak{l}]_{\mathbb{G}}, [\mathfrak{r}]_{\mathbb{H}})^{\mathfrak{e}} = e(g,h)^{\mathbf{e}\mathbf{c}'^{\intercal} + \mathbf{s}\mathbf{E}\mathbf{k}^{\intercal} + \mathbf{c}\overline{\mathbf{E}}\mathbf{r}^{\intercal}} = e(g,h)^{c_M}.$$

The correctness of the scheme is preserved under the correctness of the GPES and the preservation-of-correctness property of the distribution (Definition 9).

Theorem 1. *If* Γ *satisfies the special symbolic property (Definition 8), and the* (d_1, d_2)*-parallel DBDH assumption holds in the groups* $\mathbb{G}$*,* $\mathbb{H}$*, and* $\mathbb{G}_T$*, then the ABE scheme in Definition 13 is selectively secure. (If we allow corruption of variables, the scheme is also secure under static corruption of variables.)*

Proof (sketch). The full formal proof can be found in Appendix C. Intuitively, the security proof generalizes the strategy explained informally in Section 3.2. Specifically, each part of the key and ciphertext components that cannot be programmed with the inputs to the (d_1, d_2)-parallel DBDH are canceled by using the special symbolic property. The rest can be programmed by using similar—but possibly parallel instances of—inputs as in the example. Note that the target T is embedded in the ciphertext in the same way as in Section 3.2.

4.1 The new generic compiler in the multi-authority setting

Although our regular compiler can also prove security of multi-authority schemes, it does not explicitly consider multiple authorities. To convert the compiler to the multi-authority setting, we need to split the setup in the global setup and the authority setup, in which a subset of the parameters, associated with some authority, is generated. Furthermore, the key generation should be fragmented across authorities, meaning that it should be possible to split the key generation in independent parts. For this to work properly in practice, any non-lone key variable that occurs across multiple authorities needs to be generated by an FDH. By extension, for any such non-lone variables, the substituted vector as in the (special) symbolic property often depends on the entire $y \in \mathcal{Y}_\kappa$, rather than only the subset $y_\mathcal{A} \subseteq y$ that is relevant for one authority with identifier $\mathcal{A}$. In this case, we require the static security model. For the compiler in the multi-authority setting, we define the following two properties.

Definition 14 (Independent encodings). *Let* $\Gamma = (\mathrm{Param}, \mathrm{EncKey}, \mathrm{EncCt}, \mathrm{Pair})$ *be a GPES for a predicate family* $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$*, and let* $\mathcal{F}$ *be the FDH-generated encoding assignment mapping (Definition 10). Let* $\mathcal{A}_1, ..., \mathcal{A}_{n_{\mathrm{aut}}}$ *be* $n_{\mathrm{aut}} \in \mathbb{N}$ *authorities, such that* $\mathcal{Y}_{\kappa,\mathcal{A}_i} \subseteq \mathcal{Y}_\kappa$ *denotes the set of predicates managed by* $\mathcal{A}_i$*, which are disjoint, i.e.,* $\mathcal{Y}_{\kappa,\mathcal{A}_i} \cap \mathcal{Y}_{\kappa,\mathcal{A}_j} = \emptyset$ *for all* $i \neq j$*. The GPES has independent encodings, if the following holds:*

- *we can find mappings* $\mathfrak{A}_\alpha \colon [n_\alpha] \to [n_{\mathrm{aut}}]$ *and* $\mathfrak{A}_b \colon [n_b] \to [n_{\mathrm{aut}}]$*, where* $(n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b}) \leftarrow \mathrm{Param}(\mathrm{par})$*. Let* $\boldsymbol{\alpha}_{|l} = \{\alpha_i \mid i \in \mathfrak{A}_\alpha^{-1}(l)\}$ *and* $\mathbf{b}_{|l} = \{b_i \mid i \in \mathfrak{A}_b^{-1}(l)\}$ *for all authorities* $\mathcal{A}_l$*;*
- *for all* $y_{\mathrm{GID}} = \{y_{\mathrm{GID},\mathcal{A}_l}\}_{l \in [n_{\mathrm{aut}}]}$*, if we obtain* $(m_{1,l}, m_{2,l}, \mathbf{k}_l(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}_{|l}, \mathbf{b}_{|l}, y_{\mathrm{GID},\mathcal{A}_l}))$ $\leftarrow \mathrm{EncKey}(y_{\mathrm{GID},\mathcal{A}_l}, p)$ *for all* $y_{\mathrm{GID},\mathcal{A}_l}$*, then it should hold that running* $(m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, y_{\mathrm{GID}})) \leftarrow \mathrm{EncKey}(y_{\mathrm{GID}}, p)$ *yields* $\mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, y_{\mathrm{GID}}))$ *that is equivalent to* $\{\mathbf{k}_l(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}_l, \mathbf{b}_l, y_{\mathrm{GID},\mathcal{A}_l})\}_{l \in [n_{\mathrm{aut}}]}$*;*
- *for all* $l \in [n_{\mathrm{aut}}]$*, let* $\mathbf{r}_{|l} \subseteq \mathbf{r}$ *and* $\hat{\mathbf{r}}_{|l} \subseteq \hat{\mathbf{r}}$ *be the subsets of non-lone and lone key variables for which* $\mathbf{k}_l$ *has a non-zero coefficient. Then, for all* $r_j \in \mathbf{r}$ *for which* $l \neq l'$ *exist such that* $r_j \in \mathbf{r}_{|l} \cap \mathbf{r}_{|l'}$*, it should hold that* $\mathcal{F}(r_j) > 0$*, and similarly, for* $\hat{r}_j \in \hat{\mathbf{r}}$ *with* $l \neq l'$ *such that* $\hat{r}_j \in \mathbf{r}_{|l} \cap \mathbf{r}_{|l'}$*, we have* $\mathcal{F}(\hat{r}_j) > 0$*.*

Then, we convert the generic compiler in Definition 13 to the multi-authority setting as follows.

Definition 15 (Our multi-authority compiler). *Let* $\Gamma = (\mathrm{Param}, \mathrm{EncKey}, \mathrm{EncCt}, \mathrm{Pair})$ *be a GPES for a predicate family* $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ *as in Definition 13, with the additional property that its encodings are independent (Definition 14). Then, in the multi-authority setting, almost all algorithms are the same as in Definition 13, except that we replace the* Setup *and* KeyGen *by:*

- $\mathrm{GlobalSetup}(\lambda, \mathrm{par}) \to \mathrm{GP}$*: On input the security parameter* λ *and parameters* par*, this algorithm outputs global parameters* $\mathrm{GP} = (p, e, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, g, h)$*.*
- $\mathrm{AuthoritySetup}(\mathrm{GP}) \to (\mathcal{A}_l, \mathrm{MPK}_{\mathcal{A}_l}, \mathrm{MSK}_{\mathcal{A}_l})$*: On input the global domain parameters, this probabilistic algorithm outputs the authority identifier* $\mathcal{A}_l$*, sets* $\mathrm{MSK}_{\mathcal{A}_l} \leftarrow (\boldsymbol{\alpha}_{|l}, \{b_i \mid b_i \in \mathbf{b}_{|l} \wedge \mathcal{F}(b_i) = 0\})$*, and outputs*

$$\mathrm{MPK} = (A = \{[\alpha_i]_{\mathbb{G}_T} \mid \alpha_i \in \boldsymbol{\alpha}_{|l}\}, \{[b_i]_{\mathfrak{D}(b_i)} \mid b_i \in \mathbf{b}_{|l} \wedge \mathcal{F}(b_i) = 0\})$$

as the master public key. Note that $\boldsymbol{\alpha}_{|l}$ *and* $\mathbf{b}_{|l}$ *are as in Definition 14.*

- $\mathrm{KeyGen}(\mathcal{A}_l, \mathrm{MSK}_{\mathcal{A}_l}, \mathrm{GID}, y_{\mathrm{GID},\mathcal{A}_l}) \to \mathrm{SK}_{\mathrm{GID},\mathcal{A}_l,y_{\mathrm{GID},\mathcal{A}_l}}$*: On input the master secret key* $\mathrm{MSK}_{\mathcal{A}}$ *of authority* $\mathcal{A}_l$ *and some* $y_{\mathrm{GID},\mathcal{A}_l} \in \mathcal{Y}_{\kappa,\mathcal{A}_l}$ *for identifier* GID*, this algorithm generates* $(m_{1,l}, m_{2,l}, \mathbf{k}_l(\mathbf{r}_{|l}, \hat{\mathbf{r}}_{|l}, \boldsymbol{\alpha}_{|l}, \mathbf{b}_{|l}, y_{\mathrm{GID},\mathcal{A}_l})) \leftarrow \mathrm{EncKey}(y_{\mathrm{GID},\mathcal{A}_l}, p)$*, and outputs the secret key as*

$$\mathrm{SK}_{\mathrm{GID},\mathcal{A}_l,y_{\mathrm{GID},\mathcal{A}_l}} = (y_{\mathrm{GID},\mathcal{A}_l}, \{[r_j]_{\mathfrak{D}(r_j)} \mid r_j \in \mathbf{r}_{|l} \wedge \mathcal{F}(r_j) = 0\},$$
$$\{[k_{i,l}]_{\mathfrak{D}(k_{i,l})}\}_{i\in[m_{3,i}]}).$$

The security proof for the multi-authority compiler relies heavily on the proof for Theorem 1. This proof can be found in Appendix D.

Theorem 2. *If Γ has independent encodings and satisfies the special symbolic property (Definition 8), and the (d_1, d_2)-parallel DBDH assumption holds in $\mathbb{G}$, $\mathbb{H}$, and $\mathbb{G}_T$, then the scheme in Definition 13 is statically secure. The scheme is also secure under static corruption, if the special symbolic property holds for $\mathfrak{a} = \bigcup_{l\in\mathfrak{C}} \boldsymbol{\alpha}_{|l}$ and $\mathfrak{b} = \bigcup_{l\in\mathfrak{C}} \mathbf{b}_{|l}$, where $\mathfrak{C}$ denotes the set of corrupted authorities.*

5 New schemes

To illustrate the effectiveness of our new compiler, we give several new constructions (in this section and Appendix E). In particular, these constructions can be instantiated with our new compiler, while existing full-security compilers cannot instantiate them. In this section, we give a new decentralized large-universe CP-ABE scheme. In the proof, we use a different technique than the "zero-out lemma" as used in statically-secure decentralized ABE [49,30].

For all schemes, we assume that $\mathcal{F}$ maps the variables to 0 unless otherwise specified. We do not define mappings for $\mathfrak{D}$, as the proofs generalize to any such mapping that is correct. We also let $\mathbf{w}$ (with $w_1 = 1$) be the vector orthogonal to all $\mathbf{A}_j$ with $j \in \Upsilon$ (Definition 1). The access policy of each decentralized scheme is extended with another mapping $\tilde{\rho}\colon [n_1] \to [n_{\mathrm{aut}}]$, which maps each row to an authority, and similarly, we extend the attribute set with a mapping $\tilde{\rho}_{\mathcal{S}}\colon \mathcal{S} \to [n_{\mathrm{aut}}]$, which maps each attribute in the set to an authority. In the proofs for decentralized ABE, we require the entire key set $\mathcal{S}$ for the substitution vector of one or more key variables. Therefore, when instantiating it with the multi-authority compiler, these schemes are statically secure.

5.1 Decentralized CP-ABE supporting OT-type negations

We give a decentralized large-universe CP-ABE scheme that supports OT-type negations. Roughly, it is a decentralized variant of the TKN20 [52] scheme, for which a simpler variant can be found in Appendix E.4. In the proofs, we use a different technique than the "zero-out lemma" as used in statically-secure decentralized ABE [49,30]. Furthermore, we extend the definition of access structures (Definition 1) to include three additional mappings. In particular, we introduce another mapping $\tau\colon [n_1] \to [m]$ that maps the rows associated with the same attributes to different integers, i.e., $m = \max_{j\in[n_1]} |\rho^{-1}(\rho(j))|$, and τ is injective on the sub-domain $\rho^{-1}(\rho(j)) \subseteq [n_1]$. We also introduce the mapping ρ' that maps the rows of the policy matrix to 1 if the attribute in the policy is not negated and to 2 if it is negated, and a function ρ_{lab} that maps the rows of the policy matrix to the label universe.

Definition 16 (Decentralized large-universe CP-ABE with OT-type negations). *We define the GPES as follows.*

- $\text{Param}(\mathcal{L})$*: Let* $\{\mathcal{A}_l\}_{[n_{\text{aut}}]}$ *be the authorities. On input the label universe* $\mathcal{L}$*, we set* $n_\alpha = n_{\text{aut}}$ *and* $n_b = (1 + 2|\mathcal{L}|)n_{\text{aut}}$*, where* $\boldsymbol{\alpha} = \{\alpha_l\}_{l\in[n_{\text{aut}}]}$*, and* $\mathbf{b} = (\{b, \{b_{l,\text{lab},0}, b_{l,\text{lab},1}\}_{\text{lab}\in\mathcal{L}}\}_{l\in[n_{\text{aut}}]})$*. We also set* $\mathcal{F}(b_{l,\text{lab},i}) = 2l+i$ *for all* $l \in [n_{\text{aut}}], i \in \{0,1\}, \text{lab} \in \mathcal{L}$*. (The FDH expects* $\mathcal{A}_l$ *and* lab *as input.)*
- $\text{EncKey}((\mathcal{S}, \tilde{\rho}_{\mathcal{S}}), p)$*: Assume that, for each* $\text{lab} \in \mathcal{L}$*, there is at most one* $\text{att} \in \mathcal{U}$ *such that* $(\text{lab}, \text{att}) \in \mathcal{S}$*. We set* $m_1 = |\tilde{\rho}_{\mathcal{S}}(\mathcal{S})| + 1$*,* $m_2 = 0$*, and* $\mathbf{k} = (\{k_{1,l} = \alpha_l + r_{\text{GID}}b_l + r_l b'_l\}_{l\in\tilde{\rho}_{\mathcal{S}}(\mathcal{S})}, \{k_{2,(\text{lab},\text{att})} = r_{\tilde{\rho}_{\mathcal{S}}(\text{att})}(b_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab},0} + x_{\text{att}}b_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab},1})\}_{(\text{lab},\text{att})\in\mathcal{S}})$*, where* x_{att} *is the representation of* att *in* $\mathbb{Z}_p$*.*
- $\text{EncCt}((\mathbf{A}, \rho, \tilde{\rho}, \rho', \rho_{\text{lab}}, \tau), p)$*: We set* $w_1 = m+n_1$*,* $w_2 = n_2-1$*,* $w'_2 = n_2-1$*,* $C_M = \tilde{s}$*,*

$$\begin{gathered}\mathbf{c} = (\{c_{1,j} = \mu_j + s_j b_{\tilde{\rho}(j)}\}_{j\in[n_1]},\\ \{c_{2,j} = s_j b'_{\tilde{\rho}(j)} + s'_{\tau(j)}(b_{\tilde{\rho}(j),\rho_{\text{lab}}(j),0} + x_{\rho(j)}b_{\tilde{\rho}(j),\rho_{\text{lab}}(j),1})\}_{j\in\Psi},\\ \{c_{2,j} = s_j b'_{\tilde{\rho}(j)} + s'_{\tau(j)}b_{\tilde{\rho}(j),\rho_{\text{lab}}(j),1}, c_{3,j} = s_{\tau(j)}(b_{\tilde{\rho}(j),\text{lab},0} + x_{\rho(j)}b_{\tilde{\rho}(j),\rho_{\text{lab}}(j),1})\}_{j\in\overline{\Psi}})\end{gathered}$$

and $\mathbf{c}' = (\{c'_j = \lambda_j + \alpha_{\tilde{\rho}(j)}s_j\}_{j\in[n_1]})$*, where* $\lambda_j = A_{j,1}\tilde{s} + \sum_{k\in[2,n_2]} A_{j,k}\hat{v}_k$*, and* $\Psi = \{j \in [n_1] \mid \rho'(j) = 1\}$ *and* $\overline{\Psi} = [n_1] \setminus \Psi$ *(i.e., the set of rows associated with the non-negated and negated attributes, respectively), and* $\mathbf{s} = (\{s_j\}_{[n_1]}, \{s'_l\}_{l\in[m]})$*.*
- $\text{Pair}((\mathbf{A}, \rho, \tilde{\rho}, \rho', \rho_{\text{lab}}, \tau), (\mathcal{S}, \tilde{\rho}_{\mathcal{S}}), p)$*: If* $(\mathbf{A}, \rho, \tilde{\rho}, \rho', \rho_{\text{lab}}, \tau) \models \mathcal{S}$*, then this algorithm determines* $\Upsilon = \{j \in \Psi \mid (\rho_{\text{lab}}(j), \rho(j)) \in \mathcal{S}\}$*,* $\overline{\Upsilon} = \{j \in \overline{\Psi} \mid (\rho_{\text{lab}}(j), \rho(j)) \notin \mathcal{S} \wedge \exists(\rho_{\text{lab}}(j), \text{att}) \in \mathcal{S}\}$ *and* $\{\varepsilon_j \in \mathbb{Z}_p\}_{j\in\Upsilon\cup\overline{\Upsilon}}$ *so that* $\sum_{j\in\Upsilon\cup\overline{\Upsilon}} \varepsilon_j\lambda_j = \tilde{s}$ *(Definition 1), and outputs the vector* $\mathbf{e} = \sum_{j\in\Upsilon\cup\overline{\Upsilon}} \varepsilon_j \mathbf{1}^{w_4}_j$ *and matrices*

$$\begin{aligned}\mathbf{E} = -&\sum_{j\in\Upsilon\cup\overline{\Upsilon}} \varepsilon_j \mathbf{1}^{w_1\times m_3}_{(1,j),(1,\tilde{\rho}(j))} - \sum_{j\in\Upsilon} \varepsilon_j \mathbf{1}^{w_1\times m_3}_{(2,\tau(j)),(2,\rho(j))}\\ &-\sum_{j\in\overline{\Upsilon}} \frac{\varepsilon_j}{x_{\text{att}_j} - \rho(j)} \mathbf{1}^{w_1\times m_3}_{(2,\tau(j)),(2,\rho(j))} \text{ and}\\ \overline{\mathbf{E}} = &\sum_{j\in\Upsilon\cup\overline{\Upsilon}} \varepsilon_j \left(\mathbf{1}^{w_3\times m_1}_{(1,j),\text{GID}} + \mathbf{1}^{w_3\times m_1}_{(2,j),\tilde{\rho}(j)}\right) + \sum_{j\in\overline{\Upsilon}} \frac{\varepsilon_j}{x_{\text{att}_j} - \rho(j)} \mathbf{1}^{w_3\times m_1}_{(3,j),\tilde{\rho}(j)},\end{aligned}$$

where att_j *is such that* $(\rho_{\text{lab}}(j), \text{att}_j) \in \mathcal{S}$*.*

Lemma 2. *The GPES in Definition 16 satisfies the special selective symbolic property.*

Proof. Let $\mathfrak{C} \subseteq [n_{\text{aut}}]$ be a set of corrupted authorities, and $d_1 = n_1$ and $d_2 = n_2 + n_1 n_2 |\rho_{\text{lab}}(n_1)|$. For simple notation of the column indices, we use $(1, k)$ and $(2, j, k, \text{lab})$ (for all $j \in [n_1], k \in [n_2], \text{lab} \in \rho_{\text{lab}}(n_1)$), which are mapped injectively in the interval $[d_2]$. We define $\text{EncB}, \text{EncR}, \text{EncS}$ as follows:

- $\text{EncB}((\mathbf{A}, \rho, \rho', \tau), \mathfrak{a}, \mathfrak{b}) \rightarrow (\{\mathbf{a}_l, \mathbf{B}_l, \mathbf{B}_{l,\text{lab},0}, \mathbf{B}_{l,\text{lab},1}\}_{l \in [n_{\text{aut}}], \text{lab} \in \mathcal{L}})$, where where $\mathbf{a}_l = \mathbf{0}^{d_1}$ and $\mathbf{B}_l, \mathbf{B}'_l = \mathbf{0}^{d_1 \times d_2}$ for all $l \in \mathfrak{C}$, and let $\mathbf{v} \in \mathbb{Z}_p^{n_2}$ (with $v_1 = 1$) be the vector orthogonal to each row $j \in \tilde{\rho}^{-1}(\mathfrak{C})$ associated with a corrupted authority. For all $l \in [n_{\text{aut}}] \setminus \mathfrak{C}$, we set:

$$\mathbf{a}_l = \sum_{j \in \tilde{\rho}^{-1}(l), k \in [n_2]} A_{j,k} v_k \mathbf{1}_j^{d_1}, \qquad \mathbf{B}_l = \sum_{j \in \tilde{\rho}^{-1}(l), k \in [2, n_2]} A_{j,k} (\mathbf{1}_{j,(1,k)}^{d_1 \times d_2} + v_k \mathbf{1}_{j,(1,1)}^{d_1 \times d_2}),$$

$$\mathbf{B}'_l = \sum_{j \in \tilde{\rho}^{-1}(l), k \in [n_2]} A_{j,k} \mathbf{1}_{j,(1,k)}^{d_1 \times d_2},$$

$$\mathbf{B}_{l,\text{lab},0} = \sum_{j \in \Psi_{l,\text{lab}}, k \in [n_2]} A_{j,k} \left(\mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2} - x_{\rho(j)} \mathbf{1}_{\tau(j),(2,j,k,\text{lab})}^{d_1 \times d_2} \right)$$
$$- \sum_{j \in \overline{\Psi}_{l,\text{lab}}, k \in [n_2]} x_{\rho(j)} A_{j,k} \mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2},$$

$$\mathbf{B}_{l,\text{lab},1} = \sum_{j \in \Psi_{l,\text{lab}}, k \in [n_2]} A_{j,k} \mathbf{1}_{\tau(j),(2,j,k,\text{lab})}^{d_1 \times d_2} + \sum_{j \in \overline{\Psi}_{l,\text{lab}}, k \in [n_2]} A_{j,k} \mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2}$$

  where $\Psi_{l,\text{lab}} = \{j \in [n_1] \mid \tilde{\rho}(j) = l \wedge \rho_{\text{lab}}(j) = \text{lab} \wedge \rho'(j) = 1\}$ and $\overline{\Psi}_{l,\text{lab}} = \{j \in [n_1] \mid \tilde{\rho}(j) = l \wedge \rho_{\text{lab}}(j) = \text{lab} \wedge \rho'(j) = 0\}$.
- $\text{EncR}((\mathbf{A}, \rho, \rho', \tau), \mathcal{S}, \mathfrak{a}, \mathfrak{b}) \rightarrow (\mathbf{r}_{\text{GID}}, \{\mathbf{r}_l\}_{l \in \tilde{\rho}_{\mathcal{S}}(\mathcal{S})})$: Let $\mathbf{w} \in (1, w_2, ..., w_{n_2}) \in \mathbb{Z}_p^{n_2}$ be such that $\mathbf{A}_j \mathbf{w}^\intercal = 0$ for all $j \in [n_1]$ with either $(\rho_{\text{lab}}(j), \rho(j)) \in \mathcal{S}$ if $\rho'(j) = 1$ or $(\rho_{\text{lab}}(j), \text{att}) \in \mathcal{S}$ with $\text{att} \neq \rho(j)$ if $\rho'(j) = 0$ (Definition 1). Then, set $\mathbf{r}_{\text{GID}} = -\overline{\mathbf{1}}_1^{d_2} + \sum_{k \in [2, n_2]} w_k \overline{\mathbf{1}}_k^{d_2}$ and

$$\mathbf{r}_l = \sum_{k \in [n_2]} w_k \overline{\mathbf{1}}_{(1,k)}^{d_2} + \sum_{j \in \Psi_l \cap \overline{\Upsilon}, k \in [n_2], (\rho_{\text{lab}}(j), \text{att}) \in \mathcal{S}} \frac{w_k}{x_{\rho(j)} - x_{\text{att}}} \overline{\mathbf{1}}_{(2,j,k,\text{lab})}^{d_2},$$

  where $\Psi_l = \{j \in \tilde{\rho}^{-1}(l) \mid \rho'(j) = 1\}$ and $\overline{\Upsilon} = \{j \in [n_1] \mid (\rho_{\text{lab}}(j), \rho(j)) \notin \mathcal{S}\}$.
- $\text{EncS}((\mathbf{A}, \rho, \rho', \tau), \mathfrak{a}, \mathfrak{b}) \rightarrow (\{\mathbf{s}_j\}_{j \in [n_1]}, \{\mathbf{s}'_l\}_{l \in [m]}, \{\hat{\mathbf{v}}_k, \hat{\mathbf{v}}'_k\}_{k \in [2, n_2]}, \tilde{\mathbf{s}})$, where

$$\tilde{\mathbf{s}} = 1, \qquad \mathbf{s}'_l = -\mathbf{1}_l^{d_1}, \qquad \mathbf{s}_j = \mathbf{1}_j^{d_1}, \qquad \hat{\mathbf{v}}_k = v_k, \qquad \hat{\mathbf{v}}'_k = \overline{\mathbf{1}}_{(1,k)}^{d_2} + v_k \overline{\mathbf{1}}_{(1,1)}^{d_2}.$$

For these substitutions, the polynomials evaluate to $\mathbf{0}$ (see Appendix F). □

Remark 4. This is the first decentralized large-universe CP-ABE scheme that supports negations and that is almost completely unbounded (see Appendix H). (The only aspect in which it is bounded is the number of re-uses of a single label in the keys.) In contrast, the only other decentralized scheme that supports negations is the scheme by Okamoto and Takashima [44], which also supports OT-type negations and is fully secure, but is bounded in the label universe and the number of label re-uses in both the keys and ciphertexts.

6 Future work

This work gives room for further improvements in the simplified design of practical ABE schemes. Most obviously, it could be investigated whether the approaches used for our compiler also carry over to full-security compilers. Furthermore, since our new complexity assumption is structurally closer to the DBDH assumption, it would be valuable to investigate whether it can be reduced to DBDH and other well-studied non-parametrized assumptions such as the symmetric external Diffie-Hellman assumption. Lastly, our decentralized schemes could be used as inspiration for generic constructions of decentralized schemes, similarly as in the single-authority setting [13]. In this way, we can efficiently achieve properties such as non-monotonicity [7] in decentralized ABE.

7 Conclusion

We have introduced a new practical compiler for ABE, which uses the symbolic property to simplify the security proofs. Although in contrast to existing full-security compilers [11,12,2,4], ours proves selective security generically, it supports full-domain hashes, flexible instantiations in the pairing-friendly groups and multi-authority extensions. These properties are widely considered attractive for practice. Furthermore, the schemes produced by our compiler are a factor 2-3 more efficient than the schemes produced by full-security compilers. To illustrate the effectiveness of our compiler, we have given several new schemes—including the first decentralized large-universe CP-ABE scheme that supports negations and is almost completely unbounded—whose proofs are much less sizable and arguably simpler to verify than the security proofs of similar schemes [49,52].

Acknowledgments. The author would like to thank Greg Alpár for proofreading the paper.

References

1. Abe, M., Groth, J., Ohkubo, M., Tango, T.: Converting cryptographic schemes from symmetric to asymmetric bilinear groups. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO. LNCS, vol. 8616, pp. 241–260. Springer (2014)
2. Agrawal, S., Chase, M.: A study of pair encodings: Predicate encryption in prime order groups. In: Kushilevitz, E., Malkin, T. (eds.) TCC. LNCS, vol. 9563, pp. 259–288. Springer (2016)
3. Agrawal, S., Chase, M.: FAME: fast attribute-based message encryption. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) CCS. pp. 665–682. ACM (2017)
4. Agrawal, S., Chase, M.: Simplifying design and analysis of complex predicate encryption schemes. In: Coron, J., Nielsen, J.B. (eds.) EUROCRYPT. LNCS, vol. 10210, pp. 627–656. Springer (2017)

5. Akinyele, J.A., Garman, C., Miers, I., Pagano, M.W., Rushanan, M., Green, M., Rubin, A.D.: Charm: a framework for rapidly prototyping cryptosystems. J. Cryptogr. Eng. **3**(2), 111–128 (2013)
6. Akinyele, J.A., Green, M., Hohenberger, S.: Using SMT solvers to automate design tasks for encryption and signature schemes. In: Sadeghi, A., Gligor, V.D., Yung, M. (eds.) CCS. pp. 399–410. ACM (2013)
7. Ambrona, M.: Generic negation of pair encodings. In: Garay, J.A. (ed.) PKC. LNCS, vol. 12711, pp. 120–146. Springer (2021)
8. Ambrona, M., Barthe, G., Gay, R., Wee, H.: Attribute-based encryption in the generic group model: Automated proofs and new constructions. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) CCS. pp. 647–664. ACM (2017)
9. 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)
10. Ambrona, M., Gay, R.: Multi-authority abe, revisited. Cryptology ePrint Archive, Report 2021/1381 (2021)
11. Attrapadung, N.: Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT. LNCS, vol. 8441, pp. 557–577. Springer (2014)
12. Attrapadung, N.: Dual system encryption framework in prime-order groups via computational pair encodings. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT. LNCS, vol. 10032, pp. 591–623. Springer (2016)
13. Attrapadung, N.: Unbounded dynamic predicate compositions in attribute-based encryption. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT. LNCS, vol. 11476, pp. 34–67. Springer (2019)
14. Attrapadung, N., Hanaoka, G., Ogawa, K., Ohtake, G., Watanabe, H., Yamada, S.: Attribute-based encryption for range attributes. In: Zikas, V., Prisco, R.D. (eds.) SCN. LNCS, vol. 9841, pp. 42–61. Springer (2016)
15. Attrapadung, N., Tomida, J.: Unbounded dynamic predicate compositions in ABE from standard assumptions. In: ASIACRYPT. pp. 405–436. Springer (2020)
16. Attrapadung, N., Yamada, S.: Duality in ABE: converting attribute based encryption for dual predicate and dual policy via computational encodings. In: Nyberg, K. (ed.) CT-RSA. LNCS, vol. 9048, pp. 87–105. Springer (2015)
17. Beimel, A.: Secure Schemes for Secret Sharing and Key Distribution. Phd thesis, Ben Gurion University (1996)
18. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) CCS. pp. 62–73. ACM (1993)
19. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: S&P. pp. 321–334. IEEE (2007)
20. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT. LNCS, vol. 3027, pp. 223–238. Springer (2004)
21. Boneh, D., Boyen, X., Goh, E.J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT. LNCS, vol. 3494, pp. 440–456. Springer (2005)
22. 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)

23. Boyen, X.: The uber-assumption family – a unified complexity framework for bilinear groups. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing. LNCS, vol. 5209, pp. 39–56. Springer (2008)
24. Chase, M.: Multi-authority attribute-based encryption. In: Vadhan, S.P. (ed.) TCC. LNCS, vol. 4392, pp. 515–534. Springer (2007)
25. Chase, M., Chow, S.S.M.: Improving privacy and security in multi-authority attribute-based encryption. In: Al-Shaer, E., Jha, S., Keromytis, A.D. (eds.) CCS. pp. 121–130. ACM (2009)
26. Chatterjee, S., Koblitz, N., Menezes, A., Sarkar, P.: Another look at tightness II: practical issues in cryptography. In: Phan, R.C., Yung, M. (eds.) Mycrypt. LNCS, vol. 10311, pp. 21–55. Springer (2016)
27. Chen, J., Gay, R., Wee, H.: Improved dual system ABE in prime-order groups via predicate encodings. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT. LNCS, vol. 9057, pp. 595–624. Springer (2015)
28. Chen, J., Wee, H.: Fully, (almost) tightly secure IBE and dual system groups. In: Canetti, R., Garay, J.A. (eds.) CRYPTO. LNCS, vol. 8043, pp. 435–460. Springer (2013)
29. Chen, J., Wee, H.: Dual system groups and its applications — compact hibe and more. Cryptology ePrint Archive, Report 2014/265 (2014)
30. Datta, P., Komargodski, I., Waters, B.: Decentralized multi-authority abe for nc^1 from computational-bdh. Cryptology ePrint Archive, Report 2021/1325 (2021)
31. Datta, P., Komargodski, I., Waters, B.: Fully adaptive decentralized multi-authority abe. Cryptology ePrint Archive, Paper 2022/1311 (2022)
32. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.L.: An algebraic framework for diffie-hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO. LNCS, vol. 8043, pp. 129–147. Springer (2013)
33. ETSI: ETSI TS 103 458 (V1.1.1). Technical specification, European Telecommunications Standards Institute (ETSI) (2018)
34. ETSI: ETSI TS 103 532 (V1.1.1). Technical specification, European Telecommunications Standards Institute (ETSI) (2018)
35. The FENTEC project. https://github.com/fentec-project
36. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for cryptographers. Discret. Appl. Math. **156**(16), 3113–3121 (2008)
37. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) CCS. ACM (2006)
38. 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)
39. Kamara, S., Lauter, K.E.: Cryptographic cloud storage. In: Sion, R., Curtmola, R., Dietrich, S., Kiayias, A., Miret, J.M., Sako, K., Sebé, F. (eds.) RLCPS. LNCS, vol. 6054, pp. 136–149. Springer (2010)
40. Ladd, W., Venema, M., Verma, T.: Portunus: Re-imagining access control in distributed systems. Cryptology ePrint Archive, Paper 2023/094 (2023)
41. Lewko, A., Waters, B.: Decentralizing attribute-based encryption. In: EUROCRYPT. pp. 568–588. Springer (2011)
42. Lewko, A.B., Waters, B.: New proof methods for attribute-based encryption: Achieving full security through selective techniques. In: CRYPTO. pp. 180–198. Springer (2012)

43. Okamoto, T., Takashima, K.: Fully secure functional encryption with general relations from the decisional linear assumption. In: Rabin, T. (ed.) CRYPTO. LNCS, vol. 6223, pp. 191–208. Springer (2010)
44. Okamoto, T., Takashima, K.: Decentralized attribute-based signatures. In: Kurosawa, K., Hanaoka, G. (eds.) PKC. LNCS, vol. 7778, pp. 125–142. Springer (2013)
45. Okamoto, T., Takashima, K.: Decentralized attribute-based encryption and signatures. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **103-A**(1), 41–73 (2020)
46. de la Piedra, A., Venema, M., Alpár, G.: ABE squared. https://github.com/abecryptools/abe_squared
47. de la Piedra, A., Venema, M., Alpár, G.: ABE squared: Accurately benchmarking efficiency of attribute-based encryption. TCHES **2022**(2), 192—-239 (2022)
48. Rouselakis, Y., Waters, B.: Practical constructions and new proof methods for large universe attribute-based encryption. In: Sadeghi, A., Gligor, V.D., Yung, M. (eds.) CCS. pp. 463–474. ACM (2013)
49. Rouselakis, Y., Waters, B.: Efficient statically-secure large-universe multi-authority attribute-based encryption. In: Böhme, R., Okamoto, T. (eds.) FC. LNCS, vol. 8975, pp. 315–332. Springer (2015)
50. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT. LNCS, vol. 3494, pp. 457–473. Springer (2005)
51. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT. LNCS, vol. 1233, pp. 256–266. Springer (1997)
52. Tomida, J., Kawahara, Y., Nishimaki, R.: Fast, compact, and expressive attribute-based encryption. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC. LNCS, vol. 12110, pp. 3–33. Springer (2020)
53. Venema, M., Alpár, G.: A bunch of broken schemes: A simple yet powerful linear approach to analyzing security of attribute-based encryption. In: Paterson, K.G. (ed.) CT-RSA. LNCS, vol. 12704, pp. 100–125. Springer (2021)
54. Venema, M., Alpár, G.: TinyABE: Unrestricted ciphertext-policy attribute-based encryption for embedded devices and low-quality networks. In: Batina, L., Daemen, J. (eds.) AFRICACRYPT. LNCS, vol. 13503, pp. 103–129. Springer (2022)
55. Venema, M., Alpár, G., Hoepman, J.: Systematizing core properties of pairing-based attribute-based encryption to uncover remaining challenges in enforcing access control in practice. Cryptology ePrint Archive, Report 2021/1172 (2021)
56. Waters, B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO. LNCS, vol. 5677, pp. 619–636. Springer (2009)
57. Waters, B.: Ciphertext-policy attribute-based encryption - an expressive, efficient, and provably secure realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC. LNCS, vol. 6571, pp. 53–70. Springer (2011)
58. Wee, H.: Dual system encryption via predicate encodings. In: Lindell, Y. (ed.) TCC. LNCS, vol. 8349, pp. 616–637. Springer (2014)
59. Yamada, K., Attrapadung, N., Emura, K., Hanaoka, G., Tanaka, K.: Generic constructions for fully secure revocable attribute-based encryption. In: Foley, S.N., Gollmann, D., Snekkenes, E. (eds.) ESORICS. LNCS, vol. 10493, pp. 532–551. Springer (2017)
60. Zeutro: The OpenABE library - open source cryptographic library with attribute-based encryption implementations in C/C++. https://github.com/zeutro/openabe (2020)

A Other types of encryption and predicates

A.1 Multi-authority ABE

Definition 17 (Multi-authority ABE (MA-ABE)). *A multi-authority ABE scheme for a predicate family $P = \{P_\kappa\}_{\kappa \in \mathbb{N}^c}$ over a message space $\mathcal{M} = \{M_\lambda\}_{\lambda \in \mathbb{N}}$, for authorities $\mathcal{A}_1, ..., \mathcal{A}_{n_{\text{aut}}}$ with $\mathcal{Y}_{\kappa,\mathcal{A}_i} \subseteq \mathcal{Y}_\kappa$ and for all $i \neq j$, $\mathcal{Y}_{\kappa,\mathcal{A}_i} \cap \mathcal{Y}_{\kappa,\mathcal{A}_j} = \emptyset$, consists of five algorithms:*

- $\text{GlobalSetup}(\lambda, \text{par}) \rightarrow \text{GP}$*: On input the security parameter λ and parameters* par*, this algorithm generates the global domain parameters* GP*. In addition, κ is set to $\kappa = (p, \text{par})$, where p denotes a natural number.*
- $\text{AuthoritySetup}(\text{GP}) \rightarrow (\mathcal{A}, \text{MPK}_\mathcal{A}, \text{MSK}_\mathcal{A})$*: On input the global domain parameters, this probabilistic algorithm outputs the authority identifier $\mathcal{A}$, the master public key $\text{MPK}_\mathcal{A}$ and the master secret key $\text{MSK}_\mathcal{A}$.*
- $\text{KeyGen}(\mathcal{A}, \text{MSK}_\mathcal{A}, \text{GID}, y_{\text{GID},\mathcal{A}}) \rightarrow \text{SK}_{\text{GID},\mathcal{A},y_{\text{GID},\mathcal{A}}}$*: On input the authority identifier $\mathcal{A}$, the corresponding master secret key $\text{MSK}_\mathcal{A}$ and some $y_{\text{GID},\mathcal{A}} \in \mathcal{Y}_{\kappa,\mathcal{A}}$, for the user with global identifier* GID*, this probabilistic algorithm generates a secret key $\text{SK}_{\text{GID},\mathcal{A},y_{\text{GID},\mathcal{A}}}$.*
- $\text{Encrypt}(\{\mathcal{A}_i, \text{MPK}_{\mathcal{A}_i}\}_i, x, M) \rightarrow \text{CT}_x$*: On input a set of authority identifiers, the associated master public keys $\text{MPK}_{\mathcal{A}_i}$, some $x \in \mathcal{X}_\kappa$ and message M, this probabilistic algorithm generates a ciphertext $\text{CT}_{\{\mathcal{A}_i\}_i,x}$.*
- $\text{Decrypt}(\{\mathcal{A}, \text{MPK}_\mathcal{A}, \text{SK}_{\text{GID},\mathcal{A},y_{\text{GID},\mathcal{A}}}\}, \text{CT}_{\{\mathcal{A}_i\}_i,x}) \rightarrow M$*: On input a set of authority identifiers, the associated master public keys $\text{MPK}_\mathcal{A}$ and secret keys $\{\text{SK}_{\text{GID},\mathcal{A},y_{\text{GID},\mathcal{A}}}\}$ (where $y = \bigcup_\mathcal{A} y_{\text{GID},\mathcal{A}}$), and the ciphertext $\text{CT}_{\{\mathcal{A}_i\}_i,x}$, if $P_\kappa(x,y) = 1$, then it returns M. Otherwise, it returns an error message $\perp$.*

Security. The security model is similar to that of regular ABE (Definition 3). In the Setup phase, both the GlobalSetup and AuthoritySetup are run. Furthermore, a set of authorities $\mathfrak{C} \subseteq [n_{\text{aut}}]$ is corrupted before the Setup phase is run (which we call *static corruption*). In the Setup, the master secret keys corresponding to the corrupt authorities are also shared with the attacker. Then, we define $\mathcal{Y}_{\kappa,\mathfrak{C}} \subseteq \mathcal{Y}_\kappa$ to be the collective set of key predicates that the attacker controls, by corrupting the authorities and querying secret keys. In the challenge phase and second query phase, we have the additional restriction that the challenge x^* is such that $P_\kappa(x^*, y) = 0$ for all $y \in \mathcal{Y}_{\kappa,\mathfrak{C}}$.

Multi-authority ciphertext-policy ABE. A specific instance of multi-authority ABE is multi-authority CP-ABE, which is the multi-authority variant of CP-ABE. In this special subtype of CP-ABE, we add another function $\tilde{\rho}$ to the access policy $\mathbb{A} = (\mathbf{A}, \rho, \tilde{\rho})$, which maps the rows of the matrix to the corresponding authority identifiers, i.e., $\tilde{\rho} \colon [n_1] \rightarrow [n_{\text{aut}}]$.

A.2 Key-policy ABE

The dual version of CP-ABE is key-policy ABE (KP-ABE), in which the keys are associated with policies and the ciphertexts with attribute sets [37].

Multi-authority KP-ABE. Although multi-authority KP-ABE schemes exist [24,25], they are not decentralized in that the authorities do not require any coordination. In general, the reason why such a scheme cannot be created is because the policy is enforced on the keys. Therefore, the authorities need to coordinate to establish the particular policy associated with the key. By restricting the policies, this issue can be mitigated [24,25], but it cannot be resolved.

A.3 Identity-based broadcast encryption

A special case of CP-ABE is identity-based broadcast encryption (IBBE). In this type of encryption, the predicates in $\mathcal{Y}_\kappa$ are identities, i.e., $y \in \mathcal{Y}_\kappa$ is some identity, and $\mathcal{X}_\kappa$ are sets of identities, i.e., $\mathcal{S} \in \mathcal{X}_\kappa$ is such that $\mathcal{S}$ contains strings of identities.

Decentralized IBBE. In decentralized IBBE, we allow the identities to be managed by different authorities. In particular, we attach some label $l \in [n_{[}\mathrm{aut}]$ to the identity y, denoted as (y, l), and we include a similar assignment function $\tilde{\rho} \colon \mathcal{S} \to [n_{\mathrm{aut}}]$ to the set $\mathcal{S}$.

B Proof of Lemma 1

Proof. We show that the (d_1, d_2)-parallel DBDH assumption holds generically by showing that the assumption is a member of the uber-assumption family (Definition 4), and that Corollary 1 applies.

First, note that $n_c = d_1 + d_2 + 1$, $\mathfrak{P}_{\mathbb{G}} = \mathsf{xyz}$, $\mathfrak{P}_{\mathbb{G}_T} = 1$, and

$$\mathfrak{P}_{\mathbb{G}} = \mathfrak{P}_{\mathbb{H}} = \Bigg(1, \{\mathsf{xc_i}\}_{\mathsf{i} \in [d_1]}, \{\mathsf{yc'_j}\}_{\mathsf{j} \in [d_2]}, \left\{\frac{\mathsf{z}}{\mathsf{c_i c'_j}}\right\}_{\mathsf{i} \in [d_1], \mathsf{j} \in [d_2]}, \left\{\frac{\mathsf{z}}{\mathsf{c_i c'_j}}\right\}_{\mathsf{i} \in [d_1], \mathsf{j} \in [d_2]}, \\ \left\{\frac{\mathsf{xzc_i}}{\mathsf{c_{i'} c'_j}}\right\}_{\mathsf{i}, \mathsf{i'} \in [d_1], \mathsf{i} \neq \mathsf{i'}, \mathsf{j} \in [d_2]}, \left\{\frac{\mathsf{yzc'_j}}{\mathsf{c_i c'_{j'}}}\right\}_{\mathsf{i} \in [d_1], \mathsf{j}, \mathsf{j'} \in [d_2], \mathsf{j} \neq \mathsf{j'}}\Bigg),$$

which are all monomials.

We now show that $\mathfrak{P}_{\mathbb{G}_T} = \mathsf{xyz}$ is independent of all products of the polynomials in $\mathfrak{P}_{\mathbb{G}} = \mathfrak{P}_{\mathbb{H}}$ and $\mathfrak{P}_{\mathbb{G}_T} = 1$. Because all polynomials are monomials, it suffices to show that there is no product of polynomials in $\mathfrak{P}_{\mathbb{G}} = \mathfrak{P}_{\mathbb{H}}$ that is equal to xyz:

Term	**Range**	**Required**
$\mathsf{xc_i}$	$\mathsf{i} \in [d_1]$	$\frac{\mathsf{yz}}{\mathsf{c_i}}$
$\mathsf{yc'_j}$	$\mathsf{j} \in [d_2]$	$\frac{\mathsf{xz}}{\mathsf{c'_j}}$
$\frac{\mathsf{z}}{\mathsf{c_i c'_j}}$	$\mathsf{i} \in [d_1], \mathsf{j} \in [d_2]$	$\mathsf{xyc_i c'_j}$
$\frac{\mathsf{xzc_i}}{\mathsf{c_{i'} c'_j}}$	$\mathsf{i}, \mathsf{i'} \in [d_1], \mathsf{i} \neq \mathsf{i'}, \mathsf{j} \in [d_2]$	$\frac{\mathsf{yc_{i'} c'_j}}{\mathsf{c_i}}$
$\frac{\mathsf{yzc'_j}}{\mathsf{c_i c'_{j'}}}$	$\mathsf{i} \in [d_1], \mathsf{j}, \mathsf{j'} \in [d_2], \mathsf{j} \neq \mathsf{j'}$	$\frac{\mathsf{xc_i c'_{j'}}}{\mathsf{c'_j}}$.

Because none of the terms that are required to obtain xyz are provided by the assumption, xyz is indeed independent of the products of the polynomials. Hence, the assumption is secure in the generic group model. With Corollary 1, it follows that any attacker that can solve the problem in the generic group model must take time at least $\mathcal{O}(\sqrt{p/\deg} - n_c)$, where $\deg = d_1 + d_2 + 1$. □

C Proof of Theorem 1

Proof. Suppose some attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ exists that can break the scheme in Definition 13 with non-negligible advantage ε. We show that it can be used to construct an attacker $\mathcal{A}_{(d_1,d_2)\text{-pDBDH}}$ with non-negligible advantage in a security game with challenger $\mathcal{C}_{(d_1,d_2)\text{-pDBDH}}$ as well.

- **Initialization phase:** Attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ commits to $x^* \in \mathcal{X}_\kappa$, and corruptable $\mathfrak{a} \subsetneq [n_\alpha]$ and $\mathfrak{b} \subsetneq [n_b]$ as in the special symbolic property, and sends those to challenger $\mathcal{C}_{\text{PE,IND-CPA}}$. (Note that $\mathfrak{a} = \mathfrak{b} = \emptyset$, if we do not allow corruption.) Let $\mathcal{C}_{(d_1,d_2)\text{-pDBDH}}$ be a challenger that sends the terms of the assumption in Definition 12, where d_1, d_2 as in the special symbolic property (Definition 8). Let $\text{EncB}, \text{EncR}, \text{EncS}$ be the three algorithms that generate the necessary substitutions for the variables in the encodings.
- **Setup phase:** Challenger $\mathcal{C}_{\text{PE,IND-CPA}}$ runs $(\mathbf{a}_1, ..., \mathbf{a}_{n_\alpha}, \mathbf{B}_1, ..., \mathbf{B}_n) \leftarrow \text{EncB}(x^*)$ to obtain the necessary substitutions for the master public key MPK. It constructs the master public key as

$$\text{MPK} = \Bigg(\{A_j = e(g,h)^{\bar{\alpha}_j} \cdot \prod_{\mathsf{i} \in [d_1]} e([\mathsf{x}]_{\mathbb{G}}^{\mathsf{c_i}}, [\mathsf{z}]_{\mathbb{H}})^{(\mathbf{a}_j)_{\mathsf{i}}}\}_{j \in [n_\alpha] \setminus \mathfrak{a}},$$

$$\left\{ [b_k]_{\mathfrak{D}(b_k)} = [\bar{b}_k]_{\mathfrak{D}(b_k)} \cdot \prod_{\mathsf{i} \in [d_1], \mathsf{j} \in [d_2]} \left[(\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} \frac{\mathsf{z}}{\mathsf{c_i c'_j}} \right]_{\mathfrak{D}(b_k)} \right\}_{k \in [n] \setminus \mathfrak{b} | \mathcal{F}(b_k) = 0},$$

$$\{A_j = e(g,h)^{\bar{\alpha}_j}\}_{j \in \mathfrak{a}}, \left\{ [b_k]_{\mathfrak{D}(b_k)} = [\bar{b}_k]_{\mathfrak{D}(b_k)} \right\}_{k \in \mathfrak{b}} \Big),$$

  where $\bar{\alpha}, \bar{b}_k \in_R \mathbb{Z}_p$ for all $k \in [n]$. Note that $\left[(\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} \frac{\mathsf{z}}{\mathsf{c_i c'_j}} \right]_{\mathfrak{D}(b_k)}$ can be generated from the terms in the assumption by computing $\left[\frac{\mathsf{z}}{\mathsf{c_i c'_j}} \right]_{\mathfrak{D}(b_k)}^{(\mathbf{B}_k)_{\mathsf{i},\mathsf{j}}}$.
- **Random oracle query phase for $\mathcal{H}_i$:** If attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ queries the random oracle $\mathcal{H}_i$ the input corresponding to common variable b_k (with $\mathcal{F}(b_k) = i$), then it obtains $(\mathbf{a}_1, ..., \mathbf{a}_{n_\alpha}, \mathbf{B}_1, ..., \mathbf{B}_n) \leftarrow \text{EncB}(x^*)$ and outputs

$$[b_k]_{\mathfrak{D}(b_k)} = \begin{cases} [\bar{b}_k]_{\mathfrak{D}(b_k)} \cdot \prod_{\mathsf{i} \in [d_1], \mathsf{j} \in [d_2]} \left[(\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} \frac{\mathsf{z}}{\mathsf{c_i c'_j}} \right]_{\mathfrak{D}(b_k)} & \text{if } k \in [n] \setminus \mathfrak{b} \\ [\bar{b}_k]_{\mathfrak{D}(b_k)} & \text{if } k \in \mathfrak{b}, \end{cases}$$

where $\bar{b}_k \in_R \mathbb{Z}_p$. If the oracle is queried implicitly for non-lone variable r_j, it runs $(\mathbf{r}_1, ..., \mathbf{r}_{m_1}, \hat{\mathbf{r}}_1, ..., \hat{\mathbf{r}}_{m_2}) \leftarrow \mathrm{EncR}(x^*, y)$ and outputs

$$[r_j]_{\mathfrak{D}(r_j)} = [\bar{r}_j]_{\mathfrak{D}(r_j)} \cdot \prod_{\mathsf{i}\in[d_1]} [(\mathbf{r}_j)_{\mathsf{i}}\mathsf{x}\mathsf{c}_{\mathsf{i}}]_{\mathfrak{D}(r_j)},$$

and if it is queried for non-lone variable s_j, it runs $(\mathbf{s}_0, ..., \mathbf{s}_{w_1}, \hat{\mathbf{s}}_1, ..., \hat{\mathbf{s}}_{w_2}) \leftarrow \mathrm{EncS}(x^*)$ and outputs

$$[s_j]_{\mathfrak{D}(s_j)} = [\bar{s}_j]_{\mathfrak{D}(s_j)} \cdot \prod_{\mathsf{j}\in[d_2]} [(\mathbf{s}_j)_{\mathsf{j}}\mathsf{y}\mathsf{c}'_{\mathsf{j}}]_{\mathfrak{D}(s_j)},$$

where $\bar{r}_j, \bar{s}_j \in_R \mathbb{Z}_p$.

- **First query phase:** Attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ queries secret keys for $y \in \mathcal{Y}_\kappa$. Challenger $\mathcal{C}_{\text{PE,IND-CPA}}$ generates $(\mathbf{r}_1, ..., \mathbf{r}_{m_1}, \hat{\mathbf{r}}_1, ..., \hat{\mathbf{r}}_{m_2}) \leftarrow \mathrm{EncR}(x^*, y)$ and $\bar{r}_j \in_R \mathbb{Z}_p$ for all $j \in [m_1]$ and programs the secret key as

$$\mathrm{SK}_y = (y, \{[r_j]_{\mathfrak{D}(r_j)} = [\bar{r}_j]_{\mathfrak{D}(r_j)} \cdot \prod_{\mathsf{i}\in[d_1]} [(\mathbf{r}_j)_{\mathsf{i}}\mathsf{x}\mathsf{c}_{\mathsf{i}}]_{\mathfrak{D}(r_j)}\}_{j\in[m_1]}, \{[k_i]_{\mathfrak{D}(k_i)}\}_{i\in[m_3]}),$$

such that $[k_i]_{\mathfrak{D}(k_i)} = [\sum_{j\in[n_\alpha]} \delta_{i,j}\alpha_j + \sum_{j\in[m_2]} \delta_{i,j}\hat{r}_j + \sum_{j\in[m_1],k\in[n]} \delta_{i,j,k} r_j b_k]_{\mathfrak{D}(k_i)}$ is programmed by implicitly setting

$$[\alpha_j]_{\mathfrak{D}(\alpha_j)} = [\bar{\alpha}_j]_{\mathfrak{D}(\alpha_j)} \cdot \prod_{\mathsf{j}\in[d_2]} \left[(\mathbf{a}_j)_{\mathsf{j}} \frac{\mathsf{x}\mathsf{z}}{\mathsf{c}'_{\mathsf{j}}}\right]_{\mathfrak{D}(\alpha_j)} \quad \text{for all } j \in [n_\alpha],$$

$$[\hat{r}_j]_{\mathfrak{D}(\hat{r}_j)} = \prod_{\mathsf{j}\in[d_2]} \left[(\hat{\mathbf{r}}_j)_{\mathsf{j}} \frac{\mathsf{x}\mathsf{z}}{\mathsf{c}'_{\mathsf{j}}}\right]_{\mathfrak{D}(\hat{r}_j)} \quad \text{for all } j \in [d_1],$$

$$[r_j b_k]_{\mathfrak{D}(b_k)} = [\bar{r}_j b_k]_{\mathfrak{D}(b_k)} \cdot [r_j \bar{b}_k]_{\mathfrak{D}(b_k)} \cdot \prod_{\mathsf{i},\mathsf{i}'\in[d_1],\mathsf{j}\in[d_2]} \left[(\mathbf{r}_j)_{\mathsf{i}'} (\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} \frac{\mathsf{z}\mathsf{x}\mathsf{c}_{\mathsf{i}'}}{\mathsf{c}_{\mathsf{i}}\mathsf{c}'_{\mathsf{j}}}\right]_{\mathfrak{D}(b_k)}$$

for all $j \in [d_2]$, such that the only terms we cannot program with the terms of the (d_1, d_2)-pDBDH assumption are those with $\frac{\mathsf{x}\mathsf{z}}{\mathsf{c}'_{\mathsf{j}}}$ for all $\mathsf{j} \in [d_2]$, which includes the rightmost terms of $[r_j b_k]_{\mathfrak{D}(b_k)}$ for $\mathsf{i} = \mathsf{i}'$, i.e.,

$$\prod_{\mathsf{i}\in[d_1],\mathsf{j}\in[d_2]} \left[(\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} (\mathbf{r}_j)_{\mathsf{i}} \frac{\mathsf{z}\mathsf{x}\mathsf{c}_{\mathsf{i}}}{\mathsf{c}_{\mathsf{i}}\mathsf{c}'_{\mathsf{j}}}\right]_{\mathfrak{D}(b_k)} = \prod_{\mathsf{i}\in[d_1],\mathsf{j}\in[d_2]} \left[(\mathbf{B}_k\mathbf{r}_j^{\intercal})_{\mathsf{j}} \frac{\mathsf{x}\mathsf{z}}{\mathsf{c}'_{\mathsf{j}}}\right]_{\mathfrak{D}(b_k)}.$$

For these terms, it follows from the selective property that these are canceled in the simulation of k_i, because

$$k_i = \sum_{j\in[n_\alpha]} \delta_{i,j}\mathbf{a}_j + \sum_{j\in[m_2]} \hat{\delta}_{i,j}\hat{\mathbf{r}}_j + \sum_{j\in[m_1],k\in[n]} \delta_{i,j,k}\mathbf{B}_k\mathbf{r}_j^{\intercal} = \mathbf{0}^{d_2},$$

from which it follows that for all $\mathsf{j} \in [d_2]$:

$$\left(\sum_{j \in [n_\alpha]} \delta_{i,j} (\mathbf{a}_j)_{\mathsf{j}} + \sum_{j \in [m_2]} \hat{\delta}_{i,j} (\hat{\mathbf{r}}_j)_{\mathsf{j}} + \sum_{j \in [m_1], k \in [n]} \delta_{i,j,k} (\mathbf{B}_k \mathbf{r}_j^\intercal)_{\mathsf{j}} \right) \frac{\mathsf{xz}}{\mathsf{c}'_{\mathsf{j}}}$$
$$= \sum_{j \in [n_\alpha]} \delta_{i,j} (\mathbf{a}_j)_{\mathsf{j}} \frac{\mathsf{xz}}{\mathsf{c}'_{\mathsf{j}}} + \sum_{j \in [m_2]} \hat{\delta}_{i,j} (\hat{\mathbf{r}}_j)_{\mathsf{j}} \frac{\mathsf{xz}}{\mathsf{c}'_{\mathsf{j}}} + \sum_{j \in [m_1], k \in [n]} \delta_{i,j,k} (\mathbf{B}_k \mathbf{r}_j^\intercal)_{\mathsf{j}} \frac{\mathsf{xz}}{\mathsf{c}'_{\mathsf{j}}} = 0.$$

Note that the rest of the terms can be programmed as follows:

$$[\bar{r}_j b_k]_{\mathfrak{D}(b_k)} = [b_k]_{\mathfrak{D}(b_k)}^{\bar{r}_j}, \qquad [r_j \bar{b}_k]_{\mathfrak{D}(b_k)} = [r_j]_{\mathfrak{D}(b_k)}^{\bar{b}_k},$$
$$\prod_{\mathsf{i},\mathsf{i}' \in [d_1], \mathsf{i} \neq \mathsf{i}', \mathsf{j} \in [d_2]} \left[(\mathbf{r}_j)_{\mathsf{i}'} (\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} \frac{\mathsf{zxc}_{\mathsf{i}'}}{\mathsf{c}_{\mathsf{i}} \mathsf{c}'_{\mathsf{j}}} \right]_{\mathfrak{D}(b_k)} = \prod_{\mathsf{i},\mathsf{i}' \in [d_1], \mathsf{i} \neq \mathsf{i}', \mathsf{j} \in [d_2]} \left[\frac{\mathsf{xzc}_{\mathsf{i}'}}{\mathsf{c}_{\mathsf{i}} \mathsf{c}'_{\mathsf{j}}} \right]_{\mathfrak{D}(b_k)}^{(\mathbf{r}_j)_{\mathsf{i}'} (\mathbf{B}_k)_{\mathsf{i},\mathsf{j}}}.$$

- **Challenge phase:** Attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ sends two messages M_0 and M_1 of equal length in $\mathcal{M}_\lambda$ to challenger $\mathcal{C}_{\text{PE,IND-CPA}}$. The challenger flips a coin, i.e., $\beta \in_R \{0,1\}$, encrypts M_β under x^* and sends the resulting ciphertext CT_{x^*} to the attacker as follows. First, it runs $(\mathbf{s}_0, ..., \mathbf{s}_{w_1}, \hat{\mathbf{s}}_1, ..., \hat{\mathbf{s}}_{w_2}) \leftarrow \text{EncS}(x^*)$, and it sets

$$\{[s_j]_{\mathfrak{D}(s_j)} = [\bar{s}_j]_{\mathfrak{D}(s_j)} \cdot \prod_{\mathsf{j} \in [d_2]} [(\mathbf{s}_j)_{\mathsf{j}} \mathsf{yc}'_{\mathsf{j}}]_{\mathfrak{D}(s_j)} \}_{j \in \overline{[w_1]}},$$

such that $[c_i]_{\mathfrak{D}(c_i)} = [\sum_{j \in [w_2]} \eta_{i,j} \hat{s}_j + \sum_{j \in \overline{[w_1]}, k \in [n]} \eta_{i,j,k} s_j b_k]_{\mathfrak{D}(c_i)}$ can be programmed by implicitly setting

$$\{[\hat{s}_j]_{\mathfrak{D}(s_j)} = \prod_{\mathsf{i} \in [d_1]} \left[(\hat{\mathbf{s}}_j)_{\mathsf{i}} \frac{\mathsf{yz}}{\mathsf{c}_{\mathsf{i}}} \right]_{\mathfrak{D}(s_j)} \}_{j \in [w_2]},$$

$$[s_j b_k]_{\mathfrak{D}(c_i)} = [\bar{s}_j b_k]_{\mathfrak{D}(c_i)} \cdot [s_j \bar{b}_k]_{\mathfrak{D}(c_i)} \cdot \prod_{\mathsf{i} \in [d_1], \mathsf{j}, \mathsf{j}' \in [d_2]} \left[(\mathbf{s}_j)_{\mathsf{j}'} (\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} \frac{\mathsf{yzc}'_{\mathsf{j}'}}{\mathsf{c}_{\mathsf{i}} \mathsf{c}'_{\mathsf{j}}} \right]_{\mathfrak{D}(b_k)}.$$

Only the terms with $\frac{\mathsf{yz}}{\mathsf{c}_{\mathsf{i}}}$ (i.e., for $\mathsf{j} = \mathsf{j}'$) cannot be programmed from the terms in the assumption, but are canceled in the simulation of $[c_i]_{\mathfrak{D}(c_i)}$, because

$$c_i = \sum_{j \in [w_2]} \eta_{i,j} \hat{\mathbf{s}}_j + \sum_{j \in \overline{[w_1]}, k \in [n]} \eta_{i,j,k} \mathbf{s}_j \mathbf{B}_k = \mathbf{0}^{d_1},$$

from which it follows that, for all $\mathsf{i} \in [d_1]$, we have

$$\left(\sum_{j \in [w_2]} \eta_{i,j} (\hat{\mathbf{s}}_j)_{\mathsf{i}} + \sum_{j \in \overline{[w_1]}, k \in [n]} \eta_{i,j,k} (\mathbf{s}_j \mathbf{B}_k)_{\mathsf{i}} \right) \frac{\mathsf{yz}}{\mathsf{c}_{\mathsf{i}}}$$
$$= \sum_{j \in [w_2]} \eta_{i,j} (\hat{\mathbf{s}}_j)_{\mathsf{i}} \frac{\mathsf{yz}}{\mathsf{c}_{\mathsf{i}}} + \sum_{j \in \overline{[w_1]}, k \in [n]} \eta_{i,j,k} (\mathbf{s}_j \mathbf{B}_k)_{\mathsf{i}} \frac{\mathsf{yz}}{\mathsf{c}_{\mathsf{i}}} = 0.$$

The other terms can be programmed by computing

$$[\bar{s}_j b_k]_{\mathfrak{D}(c_i)} = [b_k]^{\bar{s}_j}_{\mathfrak{D}(c_i)}, \qquad [s_j \bar{b}_k]_{\mathfrak{D}(c_i)} = [s_j]^{\bar{b}_k}_{\mathfrak{D}(c_i)},$$

$$\prod_{\mathsf{i} \in [d_1], \mathsf{j}, \mathsf{j}' \in [d_2], \mathsf{j} \neq \mathsf{j}'} \left[(\mathbf{s}_j)_{\mathsf{j}'} (\mathbf{B}_k)_{\mathsf{i},\mathsf{j}} \frac{\mathsf{yzc}'_{\mathsf{j}'}}{\mathsf{c}_\mathsf{i} \mathsf{c}'_\mathsf{j}} \right]_{\mathfrak{D}(b_k)} = \prod_{\mathsf{i} \in [d_1], \mathsf{j}, \mathsf{j}' \in [d_2], \mathsf{j} \neq \mathsf{j}'} \left[\frac{\mathsf{yzc}'_{\mathsf{j}'}}{\mathsf{c}_\mathsf{i} \mathsf{c}'_\mathsf{j}} \right]^{(\mathbf{s}_j)_{\mathsf{j}'} (\mathbf{B}_k)_{\mathsf{i},\mathsf{j}}}_{\mathfrak{D}(b_k)}.$$

Then, $[c'_i]_{\mathbb{G}_T}$ can be programmed by implicitly setting $\tilde{s}_j = \mathsf{xyz}$, by using that we can compute $\left[\mathsf{xyz} \frac{\mathsf{c}'_{\mathsf{j}'}}{\mathsf{c}'_\mathsf{j}}\right]_{\mathbb{G}_T} = e([\mathsf{xc}_\mathsf{i}]_{\mathbb{G}}, \left[\frac{\mathsf{yzc}'_\mathsf{j}}{\mathsf{c}_\mathsf{i} \mathsf{c}'_{\mathsf{j}'}}\right]_{\mathbb{H}})$ for all $\mathsf{j} \neq \mathsf{j}' \in [d_2]$, and

$$c'_i = \sum_{j \in [n_\alpha], j' \in \overline{[w_1]}} \eta'_{i,j,j'} \boldsymbol{\alpha}_j \mathbf{s}^\intercal_{j'} + \sum_{j \in [w'_2]} \hat{\eta}'_{i,j} \tilde{\mathbf{s}}_j = 0.$$

Hence,

$$\begin{aligned} [c'_i]_{\mathbb{G}_T} = & \prod_{j \in [n_\alpha], j' \in \overline{[w_1]}} [s_{j'}]^{\eta'_{i,j,j'} \bar{\alpha}_j}_{\mathbb{G}_T} \cdot \prod_{j \in [n_\alpha], j' \in \overline{[w_1]}} [\alpha_j]^{\eta'_{i,j,j'} \bar{s}_{j'}}_{\mathbb{G}_T} \\ & \cdot \prod_{j \in [n_\alpha], j' \in \overline{[w_1]}, \mathsf{j}, \mathsf{j}' \in [d_2]} \left[(\mathbf{a}_j)_\mathsf{j} (\mathbf{s}_{j'})_{\mathsf{j}'} \mathsf{yc}'_{\mathsf{j}'} \frac{\mathsf{xz}}{\mathsf{c}'_\mathsf{j}} \right]^{\eta'_{i,j,j'}}_{\mathbb{G}_T} \cdot \prod_{j \in [w'_2]} [\tilde{\mathsf{s}} \mathsf{xyz}]^{\hat{\eta}'_{i,j}}_{\mathbb{G}_T} \\ = & \prod_{j \in [n_\alpha], j' \in \overline{[w_1]}} \left([s_{j'}]^{\eta'_{i,j,j'} \bar{\alpha}_j}_{\mathbb{G}_T} \cdot A_j^{\eta'_{i,j,j'} \bar{s}_{j'}} \cdot \prod_{\mathsf{j}, \mathsf{j}' \in [d_2], \mathsf{j} \neq \mathsf{j}'} \left[\mathsf{xyz} \frac{\mathsf{c}'_{\mathsf{j}'}}{\mathsf{c}'_\mathsf{j}} \right]^{\eta'_{i,j,j'} (\mathbf{a}_j)_\mathsf{j} (\mathbf{s}_{j'})_{\mathsf{j}'}}_{\mathbb{G}_T} \right) \\ & \cdot [\mathsf{xyz}]^{\sum_{j \in [n_\alpha], j' \in \overline{[w_1]}} \eta'_{i,j,j'} (\mathbf{a}_j) (\mathbf{s}_{j'})^\intercal + \sum_{j \in [w'_2]} \hat{\eta}'_{i,j} \tilde{\mathbf{s}}}_{\mathbb{G}_T} \end{aligned}$$

where $[s_{j'}]_{\mathbb{G}_T}$ can be computed from $[s_{j'}]_{\mathfrak{D}(s_{j'})}$. Furthermore, let $\mathcal{J}_1 = \{(j, j') \in [n_\alpha] \times \overline{[w_1]} \mid \zeta_{j,j'} \neq 0\}$ and $\mathcal{J}_2 = \{j \in [w_2] \mid \zeta_j \neq 0\}$. Then, we let

$$\begin{aligned} C &= M_\beta \cdot \prod_{(j,j') \in \mathcal{J}_1} A_j^{\zeta_{j,j'} s_{j'}} \prod_{j \in \mathcal{J}_2} e(g,h)^{\zeta_j \hat{s}_j} \\ &= M_\beta \cdot \prod_{(j,j') \in \mathcal{J}_1} [s_{j'}]^{\bar{\alpha}_j \zeta_{j,j'}}_{\mathbb{G}_T} \cdot \prod_{(j,j') \in \mathcal{J}_1} [\mathsf{xyz}]^{\zeta_{j,j'}}_{\mathbb{G}_T} \prod_{j \in \mathcal{J}_2} [\mathsf{xyz}]^{\zeta_j}_{\mathbb{G}_T} \\ &= M_\beta \cdot \prod_{(j,j') \in \mathcal{J}_1} [s_{j'}]^{\bar{\alpha}_j \zeta_{j,j'}}_{\mathbb{G}_T} \cdot T^{\sum_{(j,j') \in \mathcal{J}_1} \zeta_{j,j'} + \sum_{j \in \mathcal{J}_2} \zeta_j}, \end{aligned}$$

which is well-formed if $T = e(g,h)^{\mathsf{xyz}}$. It outputs the ciphertext as

$$\mathrm{CT}^*_{x^*} = (x^*, C, [s]_{\mathfrak{D}(s)}, \{[s_j]_{\mathfrak{D}(s_j)}\}_{j \in \overline{[w_1]}}, \{[c_i]_{\mathfrak{D}(c_i)}\}_{i \in [w_3]}, \{[c'_i]_{\mathbb{G}_T}\}_{i \in [w_4]})$$

- **Second query phase:** This phase is identical to the first query phase.
- **Decision phase:** Attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ outputs a guess β' for β. If $\beta' = \beta$, then attacker $\mathcal{A}_{(d_1,d_2)\text{-pDBDH}}$ concludes that the ciphertext was well-formed. Thus, it outputs that $T = e(g,h)^{\mathsf{xyz}}$, and otherwise, it outputs that $T \in_R \mathbb{G}_T$.

The probability that attacker $\mathcal{A}_{(d_1,d_2)\text{-pDBDH}}$ guesses correctly when $T = e(g,h)^{\mathsf{xyz}}$ holds corresponds to the success probability of attacker $\mathcal{A}_{\text{PE,IND-CPA}}$, i.e., $\varepsilon + \frac{1}{2}$. If $T \in_R \mathbb{G}_T$ holds, then attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ guesses at random, and thus, attacker $\mathcal{A}_{(d_1,d_2)\text{-pDBDH}}$ also guesses at random. Hence, the advantage

$$\mathsf{Adv}_{\mathcal{A}_{(d_1,d_2)\text{-pDBDH}}} = \varepsilon + \frac{1}{2} - \frac{1}{2} = \varepsilon$$

is non-negligible.

D Proof of Theorem 2

We prove a slightly more extended version of Theorem 2:

Theorem 3. *If Γ has independent encodings and satisfies the special symbolic property (Definition 8), and the (d_1,d_2)-parallel DBDH assumption holds in the groups $\mathbb{G}$, $\mathbb{H}$, and $\mathbb{G}_T$, then the ABE scheme in Construction 13 is statically secure. If the key-variable substitutions are independent, then the scheme is even selectively secure. The scheme is also secure under static corruption, if the special symbolic property holds for $\mathfrak{a} = \bigcup_{l \in \mathfrak{C}} \boldsymbol{\alpha}_{|l}$ and $\mathfrak{b} = \bigcup_{l \in \mathfrak{C}} \mathbf{b}_{|l}$, where $\mathfrak{C}$ denotes the set of corrupted authorities.*

In particular, we define the key-variable substitutions to be independent as follows.

Definition 18 (Independent key-variable substitutions). *Let $\Gamma = (\text{Param}, \text{EncKey}, \text{EncCt}, \text{Pair})$ be a GPES for a predicate family $P_\kappa \colon \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ with independent encodings (Definition 14) for authorities $\mathcal{A}_1, ..., \mathcal{A}_{n_{\text{aut}}}$, for which the special symbolic property holds (Definition 8). Let $\mathbf{r}_{|l}, \hat{\mathbf{r}}_{|l}$ be as in Definition 14. The GPES has independent key-variable substitutions, if for all $x \in \mathcal{X}_\kappa$ and $y_{\text{GID}} \in \mathcal{Y}_\kappa$ with $y_{\text{GID}} = \{y_{\text{GID},\mathcal{A}_l}\}_l$, running $(\mathbf{r}_1, ..., \mathbf{r}_{m_1}, \hat{\mathbf{r}}_1, ..., \hat{\mathbf{r}}_{m_2}) \leftarrow \text{EncR}(x, y_{\text{GID}})$ yields the same substitutions as running EncR on x and $y_{\text{GID},\mathcal{A}_l}$ for each $\mathcal{A}_l$ and considering the outputs for $\mathbf{r}_{|l}$ and $\hat{\mathbf{r}}_{|l}$.*

We now prove Theorem 3.

Proof. For a big part of this proof, we rely on the security proof for Theorem 1. In particular, we use that, for Γ, the scheme produced by the multi-authority compiler is indistinguishable from the scheme produced with the regular compiler in Definition 13, i.e., all key and ciphertext components are simulated in the same way as in the proof of the regular compiler.

Second, we need to assume that $\mathfrak{a} = \bigcup_{l \in \mathfrak{C}} \boldsymbol{\alpha}_{|l}$ and $\mathfrak{b} = \bigcup_{l \in \mathfrak{C}} \mathbf{b}_{|l}$, where $\mathfrak{C}$ denotes the set of corrupted authorities. Then, the challenger can share the master secret keys of the corrupted authorities, i.e., $\text{MSK}_{\mathcal{A}_l} = (\boldsymbol{\alpha}_{|l}, \{b_i \mid b_i \in \mathbf{b}_{|l} \wedge \mathcal{F}(b_i) = 0\})$, for which each $\alpha_i \in \boldsymbol{\alpha}_{|l}$ is such that $\alpha_i = \bar{\alpha}_i$ and each $b_i = \bar{b}_i$, which are known to the challenger.

Furthermore, we distinguish between the cases that the key-variable substitutions are independent or not. If they are not independent, then we use the static-security model to reduce the multi-authority scheme produced by Definition 15 to the instantiation of Γ in the regular compiler in Definition 13. In the static-security model, the attacker $\mathcal{A}_{\text{PE,IND-CPA}}$ commits to $x^* \in \mathcal{X}_\kappa$ and all $y_{\text{GID}} \in \mathcal{Y}_\kappa$ that they are going to query in the query phases. In this case, we can simply run EncR on the whole y_{GID}, and substitute the various key encoding variables for the resulting vectors. If the key-variable substitutions are independent, then we can run EncR for each $y_{\text{GID},\mathcal{A}_l}$ and thus simulate each $\text{SK}_{\text{GID},\mathcal{A}_l,y_{\text{GID},\mathcal{A}_l}}$ independently from the other secret keys for GID. In this case, we do not require the static-security model. □

E More schemes

In addition to the scheme in Section 5, we provide some more new schemes in this appendix.

E.1 More efficient decentralized large-universe CP-ABE from FDH

We first give a scheme that is similar to the Rouselakis-Waters decentralized scheme (RW15) [49], but has a more efficient decryption algorithm. In part, to achieve this, we use the multi-use techniques by Agrawal and Chase [4]. In particular, we introduce another mapping $\tau\colon [n_1] \to [m]$ that maps the rows associated with the same attributes to different integers, i.e., $m = \max_{j\in[n_1]} |\rho^{-1}(\rho(j))|$, and τ is injective on the sub-domain $\rho^{-1}(\rho(j)) \subseteq [n_1]$.

Definition 19 (Decentralized large-universe CP-ABE from FDH). *We define the GPES as follows.*

- $\text{Param}(\text{par}) \to (n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b})$*: Let* $\{\mathcal{A}_l\}_{[n_{\text{aut}}]}$ *be the authorities, and* $n_\alpha = n_{\text{aut}}$ *and* $n_b = 2n_{\text{aut}} + |\mathcal{U}|$, $\boldsymbol{\alpha} = (\{\alpha_l\}_{l\in[n_{\text{aut}}]}$, *and* $\mathbf{b} = (\{b_l, b'_l\}_{l\in[n_{\text{aut}}]}, \{b_{\text{att}}\}_{\text{att}\in\mathcal{U}})$, *where* $\mathcal{U}$ *denotes the universe. We set* $\mathcal{F}(b_{\text{att}}) = 1$ *for all* $\text{att} \in \mathcal{U}$.
- $\text{EncKey}((\mathcal{S}, \tilde{\rho}_{\mathcal{S}}), p) \to (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, (\mathcal{S}, \tilde{\rho}_{\mathcal{S}})))$*: We set* $m_1 = |\tilde{\rho}_{\mathcal{S}}(\mathcal{S})| + 1$, $m_2 = 0$, *and* $\mathbf{k} = (\{\{k_{1,l} = \alpha_l + r_{\text{GID}} b_l + r_l b'_l\}_{l\in\tilde{\rho}_{\mathcal{S}}(\mathcal{S})}, \{k_{2,\text{att}} = r_{\tilde{\rho}_{\mathcal{S}}(\text{att})} b_{\text{att}}\}_{\text{att}\in\mathcal{S}})$, *and* $\mathcal{F}(r_{\text{GID}}) = 2$.
- $\text{EncCt}((\mathbf{A}, \rho, \tilde{\rho}, \tau), p) \to (w_1, w_2, w'_2, c_M, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, (\mathbf{A}, \rho, \tilde{\rho}, \tau)), \mathbf{c}'(\mathbf{s}, \tilde{\mathbf{s}}, \boldsymbol{\alpha}, (\mathbf{A}, \rho, \tilde{\rho}, \tau)))$*: We set* $w_1 = m + n_1$, $w_2 = n_2 - 1$, $w'_2 = n_2 - 1$, $C_M = \tilde{s}$, $\mathbf{c} = (\{c_{1,j} = \mu_j + s_j b_{\tilde{\rho}(j)}, c_{2,j} = s_j b'_{\tilde{\rho}(j)} + s'_{\tau(j)} b_{\rho(j)}\}_{j\in[n_1]})$ *and* $\mathbf{c}' = (\{c'_j = \lambda_j + \alpha_{\tilde{\rho}(j)} s_j\}_{j\in[n_1]})$, *where* $\lambda_j = A_{j,1}\tilde{s} + \sum_{k\in[2,n_2]} A_{j,k}\hat{v}_k$ *and* $\mu_j = \sum_{k\in[2,n_2]} A_{j,k}\hat{v}'_k$.

Remark 5. The decryption of this scheme is more efficient than RW15 [49], because it reduces the number of required pairing operations from $|\tilde{\rho}([n_1])| + |\Upsilon|$ to $|\tilde{\rho}([n_1])| + m$, where $m = 1$ if ρ is injective.

Lemma 3. *The GPES in Definition 19 satisfies the special selective symbolic property.*

Proof. Let $\mathfrak{C} \subseteq [n_{\text{aut}}]$ be a set of corrupted authorities, and $d_1 = n_1$ and $d_2 = n_2$.

- $\text{EncB}((\mathbf{A},\rho,\tilde{\rho},\tau)) \rightarrow (\{\mathbf{a}_l,\mathbf{B}_l,\mathbf{B}'_l\}_{l\in[n_{\text{aut}}]},\{\mathbf{B}_{\text{att}}\}_{\text{att}\in\mathcal{U}})$, where $\mathbf{a}_l = \mathbf{0}^{d_1}$ and $\mathbf{B}_l,\mathbf{B}'_l,\mathbf{B}_{\text{att}} = \mathbf{0}^{d_1\times d_2}$ for all $l \in \mathfrak{C}$ and $\text{att} \notin \rho([n_1])$, and let $\mathbf{v} \in \mathbb{Z}_p^{n_2}$ (with $v_1 = 1$) be the vector orthogonal to each row $j \in \tilde{\rho}^{-1}(\mathfrak{C})$ associated with a corrupted authority. For all $l \in [n_{\text{aut}}] \setminus \mathfrak{C}$, we set:

$$\mathbf{a}_l = \sum_{j\in\tilde{\rho}^{-1}(l),k\in[n_2]} A_{j,k}v_k\mathbf{1}_j^{d_1}, \qquad \mathbf{B}_l = \sum_{j\in\tilde{\rho}^{-1}(l),k\in[2,n_2]} A_{j,k}(\mathbf{1}_{j,k}^{d_1\times d_2} + v_k\mathbf{1}_{j,1}^{d_1\times d_2}),$$
$$\mathbf{B}'_l = \sum_{j\in\tilde{\rho}^{-1}(l),k\in[n_2]} A_{j,k}\mathbf{1}_{j,k}^{d_1\times d_2}, \{\mathbf{B}_{\text{att}} = \sum_{j\in\rho^{-1}(\text{att}),k\in[n_2]} A_{j,k}\mathbf{1}_{\tau(j),k}^{d_1\times d_2}\}_{\text{att}\in\rho([n_1])}.$$

- $\text{EncR}((\mathbf{A},\rho,\tilde{\rho},\tau),(\mathcal{S},\tilde{\rho}_{\mathcal{S}})) \rightarrow (\mathbf{r}_{\text{GID}},\{\mathbf{r}_l\}_{l\in\tilde{\rho}_{\mathcal{S}}(\mathcal{S})})$, where

$$\mathbf{r}_{\text{GID}} = -\overline{\mathbf{1}}_1^{d_2} + \sum_{k\in[2,n_2]} w_k\overline{\mathbf{1}}_k^{d_2}, \qquad \mathbf{r}_l = -\sum_{k\in[n_2]} w_k\overline{\mathbf{1}}_k^{d_2}.$$

- $\text{EncS}((\mathbf{A},\rho,\tilde{\rho},\tau)) \rightarrow (\{\mathbf{s}_j\}_{j\in[n_1]},\{\mathbf{s}'_l\}_{l\in[m]},\{\hat{\mathbf{v}}_k,\hat{\mathbf{v}}'_k\}_{k\in[2,n_2]},\tilde{\mathbf{s}})$, where

$$\tilde{\mathbf{s}} = 1, \qquad \mathbf{s}_j = -\mathbf{1}_j^{d_1}, \qquad \mathbf{s}'_l = \mathbf{1}_l^{d_1}, \qquad \hat{\mathbf{v}}_k = v_k, \qquad \hat{\mathbf{v}}'_k = \overline{\mathbf{1}}_k^{d_2} + v_k\overline{\mathbf{1}}_1^{d_2}.$$

For these substitutions, the polynomials evaluate to $\mathbf{0}$ (see Appendix G). Note that, instead of applying the zero-out lemma to simulate $e(g,h)^{c'_j}$ for all $j \in [n_1]$, we introduce another vector $\mathbf{v}$ that is orthogonal to all rows associated with corrupted authorities, which we embed in $\hat{\mathbf{v}}$. □

Remark 6. By slightly adapting the scheme and setting the number of authorities to 1, we can obtain a single-authority scheme with an attribute-wise key generation [55]. With such a key generation, a single user can request secret keys for different attributes at different points in time, rather than all at once. We give a more concrete version of such a scheme in Appendix E.3.

E.2 Large-universe ABE scheme from DBDH

We use the ideas behind the CP-ABE schemes from DBDH in [57] and [30] to obtain a large-universe CP-ABE scheme from DBDH. This follows from any instantiation with our generic compiler, because $d_1 = d_2 = 1$. Recall that, for this case, the (d_1,d_2)-parallel DBDH assumption is equivalent to DBDH.

Definition 20 (Large-universe CP-ABE from DBDH). *We define the GPES as follows.*

- $\text{Param}(\text{par}) \rightarrow (n_\alpha,n_b,\boldsymbol{\alpha},\mathbf{b})$*: Let* $N_2 \in \mathbb{N}$ *be the maximum on the number of columns of LSSS matrix* $\mathbf{A}$*. We set* $n_\alpha = 1$ *and* $n_b = 1 + N_2 \cdot |\mathcal{U}|$*, where* $\boldsymbol{\alpha} = \alpha$*, and* $\mathbf{b} = (b,\{b_{\text{att},k}\}_{\text{att}\in\mathcal{U},k\in[N_2]})$*, where* $\mathcal{U}$ *denotes the universe of attributes. We also set* $\mathcal{F}(b) = 0$*, and* $\mathcal{F}(b_{\text{att},k}) = 1$ *for all* $\text{att} \in \mathcal{U}, k \in [N_2]$.
- $\text{EncKey}(\mathcal{S},p) \rightarrow (m_1,m_2,\mathbf{k}(\mathbf{r},\hat{\mathbf{r}},\boldsymbol{\alpha},\mathbf{b},\mathcal{S}))$*: We set* $m_1 = N_2$*,* $m_2 = 0$*, and* $\mathbf{k} = (k_1 = \alpha + rb, \{k_{2,\text{att}} = rb_{\text{att},1} + \sum_{k\in[2,N_2]} r_k b_{\text{att},k}\}_{\text{att}\in\mathcal{S}})$.

- $\mathrm{EncCt}((\mathbf{A},\rho),p) \rightarrow (w_1, w_2, w_2', c_M, \mathbf{c}(\mathbf{s},\hat{\mathbf{s}},\mathbf{b},(\mathbf{A},\rho)), \mathbf{c}'(\mathbf{s},\tilde{\mathbf{s}},\boldsymbol{\alpha},(\mathbf{A},\rho)))$*: Assume that* $n_2 < N_2$ *and* ρ *is injective. We set* $w_1 = 1$*,* $w_2 = n_2 - 1$*,* $w_2' = 0$*,* $C_M = \alpha s$*,* $\mathbf{c} = (\{c_{1,j,1} = A_{j,1}sb + sb_{\rho(j),1}\}_{j\in[n_1]}, \{c_{1,j,k} = A_{j,k}\hat{v}_j + sb_{\rho(j),k}\}_{j\in[n_1],k\in[2,n_2]})$ *and* $\mathbf{c}' = \emptyset$*.*

Lemma 4. *The GPES in Definition 20 satisfies the special selective symbolic property.*

Proof. Let $d_1 = d_2 = 1$. We define EncB, EncR, EncS as follows:

- $\mathrm{EncB}(\mathbf{A},\rho) \rightarrow (\mathbf{a},\mathbf{B},\{\mathbf{B}_{\mathrm{att},k}\}_{\mathrm{att}\in\mathcal{U},k\in[N_2]})$, where $\mathbf{a} = 1$, $\mathbf{B} = -1$, and $\mathbf{B}_{\mathrm{att},k} = 0$ for all att $\notin \rho(n_1)$, and $\mathbf{B}_{\mathrm{att},k} = A_{\rho^{-1}(\mathrm{att}),k}$.
- $\mathrm{EncR}((\mathbf{A},\rho),\mathcal{S}) \rightarrow (\mathbf{r}_1,...,\mathbf{r}_{n_2})$: Let $\mathbf{w} \in (1, w_2, ..., w_{n_2}) \in \mathbb{Z}_p^{n_2}$ be such that $\mathbf{A}_j\mathbf{w}^\intercal = 0$ for all $j \in [n_1]$ with $\rho(j) \in \mathcal{S}$ (Definition 1). Then, set $\mathbf{r}_1 = 1$, and $\mathbf{r}_k = w_k$ for all $k \in [2, n_2]$;.
- $\mathrm{EncS}((\mathbf{A},\rho)) \rightarrow (\mathbf{s}_0, \hat{\mathbf{v}}_1, ..., \hat{\mathbf{v}}_{w_2})$, where $\mathbf{s}_0 = 1$, and $\hat{\mathbf{v}}_j = -1$. □

E.3 CP-ABE with attribute-wise key generation

We give an example of a CP-ABE scheme with an attribute-wise key generation (as first introduced by Venema et al. [55]). In such schemes, the secret keys can be generated for different attributes at different points in time, for the same user. We use the compiler for the multi-authority setting for this functionality, which uses the static-security model to prove security, while keys may be requested for different attributes at different stages in time. To the best of our knowledge, this is the first (single-authority) CP-ABE scheme that explicitly enjoys this property, although we use the techniques of [49] to formalize it. Note, however, that the scheme is structurally closer to the unbounded CP-ABE scheme without random oracles [48].

Definition 21 (CP-ABE scheme with attribute-wise key generation). *We define the GPES as follows.*

- $\mathrm{Param}(\mathrm{par}) \rightarrow (n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b})$*: We set* $n_\alpha = 1$ *and* $n_b = 4$*, where* $\boldsymbol{\alpha} = \alpha$*, and* $\mathbf{b} = (b, b', b_0, b_1)$*, and* $\mathcal{F}(b) = \mathcal{F}(b') = \mathcal{F}(b_0) = \mathcal{F}(b_1) = 0$*.*
- $\mathrm{EncKey}(\mathcal{S},p) \rightarrow (m_1, m_2, \mathbf{k}(\mathbf{r},\hat{\mathbf{r}},\boldsymbol{\alpha},\mathbf{b},\mathcal{S}))$*: We set* $m_1 = |\mathcal{S}|+1$ *and* $m_2 = 0$*, where* $\mathbf{k} = (\{k_1 = \alpha + r_{\mathrm{GID}}b, k_{\mathrm{att}} = r_{\mathrm{GID}}b' + r_{\mathrm{att}}(b_0 + x_{\mathrm{att}}b_1)\}_{\mathrm{att}\in\mathcal{S}})$ *such that* x_{att} *is the integer representation of* att *in* $\mathbb{Z}_p$*, and* $\mathcal{F}(r_{\mathrm{GID}}) = 1$*.*
- $\mathrm{EncCt}((\mathbf{A},\rho),p) \rightarrow (w_1, w_2, w_2', c_M, \mathbf{c}(\mathbf{s},\hat{\mathbf{s}},\mathbf{b},(\mathbf{A},\rho)), \mathbf{c}'(\mathbf{s},\tilde{\mathbf{s}},\boldsymbol{\alpha},(\mathbf{A},\rho)))$*: We set* $w_1 = n_1$*,* $w_2 = n_2 - 1$*,* $w_2' = 0$*,* $C_M = \alpha s$*,*

$$\mathbf{c} = (\{c_{1,j} = \lambda_j + s_jb', c_{2,j} = s_j(b_0 + \rho(j)b_1)\}_{j\in[n_1]},$$

and $\mathbf{c}' = \emptyset$*, where* $\lambda_j = A_{j,1}sb + \sum_{k\in[2,n_2]} A_{j,k}\hat{v}_k$*.*

Lemma 5. *The GPES in Definition 21 satisfies the special selective symbolic property.*

Proof. Let $d_1 = n_1 + 1$ and $d_2 = (n_1 + 1)n_2$. For the simple representation of column indices, we use $(1, k)$ for all $k \in [n_2]$ and $(2, j, k)$ for all $j \in [n_1]$ and $k \in [n_2]$. These are mapped injectively in the interval $[d_2]$. For the row indices, we start counting at 0.

- $\text{EncB}((\mathbf{A}, \rho)) \rightarrow (\mathbf{a}, \mathbf{B}, \mathbf{B}', \mathbf{B}_0, \mathbf{B}_1)$, where $\mathbf{a} = \mathbf{1}^{d_1}_0$, and

$$\mathbf{B} = \mathbf{1}^{d_1 \times d_2}_{0,(1,1)}, \qquad \mathbf{B}' = \sum_{j \in [n_1], k \in [n_2]} A_{j,k} \mathbf{1}^{d_1 \times d_2}_{j,(1,k)},$$

$$\mathbf{B}_1 = \sum_{j \in [n_1], k \in [n_2]} A_{j,k} \mathbf{1}^{d_1 \times d_2}_{j,(2,j,k)}, \qquad \mathbf{B}_0 = -\sum_{j \in [n_1], k \in [n_2]} A_{j,k} \rho(j) \mathbf{1}^{d_1 \times d_2}_{j,(2,j,k)}.$$

- $\text{EncR}((\mathbf{A}, \rho), \mathcal{S}) \rightarrow (\mathbf{r}_{\text{GID}}, \{\mathbf{r}_{\text{att}}\}_{\text{att} \in \mathcal{S}})$, where

$$\mathbf{r}_{\text{GID}} = \sum_{k \in [n_2]} w_k \overline{\mathbf{1}}^{d_2}_{(1,k)}, \qquad \mathbf{r}_{\text{att}} = \sum_{j \in \overline{\Upsilon}, k \in [n_2]} \frac{w_k \overline{\mathbf{1}}^{d_2}_{(2,j,k)}}{x_{\text{att}} - \rho(j)},$$

  such that $\mathbf{w}$ (with $w_1 = 1$) is the vector orthogonal to all $\mathbf{A}_j$ with $j \in \Upsilon$ (Definition 1), and $\overline{\Upsilon} = [n_1] \setminus \Upsilon$.
- $\text{EncS}((\mathbf{A}, \rho)) \rightarrow (\mathbf{s}_0, \{\mathbf{s}_j\}_{j \in [n_1]}, \{\hat{\mathbf{v}}_k\}_{k \in [2, n_2]})$, where

$$\mathbf{s} = \mathbf{1}^{d_1}_0, \qquad \left\{\mathbf{s}_j = \mathbf{1}^{d_1}_j\right\}_{j \in [n_1]}, \qquad \hat{\mathbf{v}}_k = \overline{\mathbf{1}}^{d_2}_{(1,k)}.$$

Note that, in this proof, we require the knowledge of the entire key set $\mathcal{S}$ for the substitution vector of $\mathbf{r}_{\text{GID}}$. Therefore, when instantiating it with the multi-authority compiler (where we set the number of authorities to 1), the scheme is statically secure. □

E.4 CP-ABE with OT-type negations

We use the ideas behind the CP-ABE schemes supporting OT-type negations in [13] and [52] to obtain a GPES for large-universe CP-ABE scheme with OT-type negations. In this scheme, we represent each attribute as a label-value pair, e.g., "name" and "Alice", where the labels come from the label universe $\mathcal{L}$ and the values from the attribute universe $\mathcal{U}$. Furthermore, the policy $(\mathbf{A}, \rho, \rho_{\text{lab}}, \rho', \tau)$ consists of three additional mappings ρ_{lab}, ρ' and τ. Here, $\rho_{\text{lab}} \colon [n_1] \rightarrow \mathcal{L}$ maps the rows to the corresponding labels. Then, $\rho' \colon [n_1] \rightarrow \{0, 1\}$ is such that $\rho'(j) = 0$ indicates that the attribute corresponding with the j-th row is negated, and $\rho'(j) = 1$ indicates that it is not negated. The mapping $\tau \colon [n_1] \rightarrow [m]$ maps the rows corresponding to the same labels to different integers, i.e., $m = \max_{j \in [n_1]} |\rho^{-1}_{\text{lab}}((\rho_{\text{lab}}(j)))|$ is the maximum number of occurrences of one specific attribute, and τ is injective on each sub-domain $\rho^{-1}_{\text{lab}}(\rho_{\text{lab}}(j)) \subseteq [n_1]$.

Definition 22 (Large-universe CP-ABE with OT-type negations). *We define the GPES as follows.*

- $\mathrm{Param}(\mathrm{par}) \to (n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b})$*: We set* $n_\alpha = 1$ *and* $n_b = 1+2|\mathcal{L}|$*, where* $\boldsymbol{\alpha} = \alpha$*, and* $\mathbf{b} = (b, \{b_{\mathrm{lab},0}, b_{\mathrm{lab},1}\}_{\mathrm{lab}\in\mathcal{L}})$*. We also set* $\mathcal{F}(b) = 0$*, and* $\mathcal{F}(b_{\mathrm{lab},0}) = 1$ *and* $\mathcal{F}(b_{\mathrm{lab},1}) = 2$ *for all* $\mathrm{lab} \in \mathcal{L}$*.*
- $\mathrm{EncKey}(\mathcal{S}, p) \to (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, \mathcal{S}))$*: Assume that, for each* $\mathrm{lab} \in \mathcal{L}$*, there is at most one* $\mathrm{att} \in \mathcal{U}$ *such that* $(\mathrm{lab}, \mathrm{att}) \in \mathcal{S}$*. We set* $m_1 = 1$*,* $m_2 = 0$*, and* $\mathbf{k} = (k_1 = \alpha + rb, \{k_{2,(\mathrm{lab},\mathrm{att})} = r(b_{\mathrm{lab},0} + x_{\mathrm{att}} b_{\mathrm{lab},1})\}_{(\mathrm{lab},\mathrm{att})\in\mathcal{S}})$*, where* x_{att} *is the representation of* att *in* $\mathbb{Z}_p$*.*
- $\mathrm{EncCt}((\mathbf{A}, \rho, \rho', \tau), p) \to (w_1, w_2, w_2', c_M, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, (\mathbf{A}, \rho, \rho', \tau)), \mathbf{c}'(\mathbf{s}, \tilde{\mathbf{s}}, \boldsymbol{\alpha}, (\mathbf{A}, \rho, \rho', \tau)))$*: We set* $w_1 = m$*,* $w_2 = n_2 - 1$*,* $w_2' = 0$*,* $C_M = \alpha s$*,*

$$\mathbf{c} = (\{c_{1,j} = \lambda_j + s_{\tau(j)}(b_{\mathrm{lab},0} + x_{\rho(j)} b_{\rho_{\mathrm{lab}}(j),1})\}_{j\in\Psi},$$
$$\{c_{1,j} = \lambda_j + s_{\tau(j)} b_{\rho_{\mathrm{lab}}(j),1}, c_{2,j} = s_{\tau(j)}(b_{\mathrm{lab},0} + \rho(j) b_{\rho_{\mathrm{lab}}(j),1})\}_{j\in\overline{\Psi}})$$

  and $\mathbf{c}' = \emptyset$*, where* $\lambda_j = A_{j,1} sb + \sum_{k\in[2,n_2]} A_{j,k}\hat{v}_k$*, and* $\Psi = \{j \in [n_1] \mid \rho'(j) = 1\}$ *and* $\overline{\Psi} = [n_1] \setminus \Psi$ *(i.e., the set of rows associated with the positive and negative attributes, respectively).*

Lemma 6. *The GPES in Definition 22 satisfies the special selective symbolic property.*

Proof. Let $d_1 = m$ and $d_2 = n_2 + n_1 n_2 |\rho_{\mathrm{lab}}(n_1)|$. For simple notation of the column indices, we use $(1, k)$ (for all $k \in [n_2]$) and $(2, j, k, \mathrm{lab})$ (for all $j \in [n_1], k \in [n_2], \mathrm{lab} \in \rho_{\mathrm{lab}}(n_1)$), which are mapped injectively in the interval $[d_2]$. We define EncB, EncR, EncS as follows:

- $\mathrm{EncB}((\mathbf{A}, \rho, \rho', \tau)) \to (\mathbf{a}, \mathbf{B}, \{\mathbf{B}_{\mathrm{lab},0}, \mathbf{B}_{\mathrm{lab},1}\}_{\mathrm{lab}\in\mathcal{L}})$, where $\mathbf{a} = \mathbf{1}_1^{d_1}$, $\mathbf{B} = \mathbf{1}_{1,(1,1)}^{d_1\times d_2}$, and

$$\begin{aligned}\mathbf{B}_{\mathrm{lab},0} = &\sum_{j\in\Psi_{\mathrm{lab}}, k\in[n_2]} A_{j,k} v_k \left(\mathbf{1}_{\tau(j),(1,k)}^{d_1\times d_2} - \rho(j)\mathbf{1}_{\tau(j),(2,j,k,\mathrm{lab})}^{d_1\times d_2}\right)\\ &- \sum_{j\in\overline{\Psi}_{\mathrm{lab}}, k\in[n_2]} \rho(j) A_{j,k} v_k \mathbf{1}_{\tau(j),(1,k)}^{d_1\times d_2},\end{aligned}$$

$$\mathbf{B}_{\mathrm{lab},1} = \sum_{j\in\Psi_{\mathrm{lab}}, k\in[n_2]} A_{j,k} v_k \mathbf{1}_{\tau(j),(2,j,k,\mathrm{lab})}^{d_1\times d_2} + \sum_{j\in\overline{\Psi}_{\mathrm{lab}}, k\in[n_2]} A_{j,k} v_k \mathbf{1}_{\tau(j),(1,k)}^{d_1\times d_2},$$

  where $\Psi_{\mathrm{lab}} = \{j \in [n_1] \mid \rho_{\mathrm{lab}}(j) = \mathrm{lab} \wedge \rho'(j) = 1\}$ and $\overline{\Psi}_{\mathrm{lab}} = \{j \in [n_1] \mid \rho_{\mathrm{lab}}(j) = \mathrm{lab} \wedge \rho'(j) = 0\}$.
- $\mathrm{EncR}((\mathbf{A}, \rho, \rho', \tau), \mathcal{S}) \to (\mathbf{r})$: Let $\mathbf{w} \in (1, w_2, ..., w_{n_2}) \in \mathbb{Z}_p^{n_2}$ be such that $\mathbf{A}_j \mathbf{w}^{\intercal} = 0$ for all $j \in [n_1]$ with either $(\rho_{\mathrm{lab}}(j), \rho(j)) \in \mathcal{S}$ if $\rho'(j) = 1$ or $(\rho_{\mathrm{lab}}(j), \mathrm{att}) \in \mathcal{S}$ with $\mathrm{att} \neq \rho(j)$ if $\rho'(j) = 0$ (Definition 1). Then, set

$$\mathbf{r} = \sum_{k\in[n_2]} w_k \overline{\mathbf{1}}_{(1,k)}^{d_2} + \sum_{j\in\Psi\cap\overline{\Upsilon}, k\in[n_2], (\mathrm{lab},\mathrm{att})\in\mathcal{S}|\mathrm{lab}=\rho_{\mathrm{lab}}(j)} \frac{w_k}{\rho(j) - x_{\mathrm{att}}} \overline{\mathbf{1}}_{(2,j,k,\mathrm{lab})}^{d_2},$$

  where $\Psi = \{j \in [n_1] \mid \rho'(j) = 1\}$ and $\overline{\Upsilon} = \{j \in [n_1] \mid (\rho_{\mathrm{lab}}(j), \rho(j)) \notin \mathcal{S}\}$.
- $\mathrm{EncS}((\mathbf{A}, \rho, \rho', \tau)) \to (\mathbf{s}_0, \mathbf{s}_1, ..., \mathbf{s}_m, \hat{\mathbf{v}}_1, ..., \hat{\mathbf{v}}_{w_2})$, where $\mathbf{s}_0 = \mathbf{1}_1^{d_1}$, $\mathbf{s}_l = \mathbf{1}_l^{d_1}$ for all $l \in [m]$ and $\hat{\mathbf{v}}_k = \overline{\mathbf{1}}_{(1,k)}^{d_2}$ for all $k \in [n_2]$. □

KP-ABE variant. Because the structure of the PES fits in the AC17 class of PESs, we can use the conversion techniques in [4] to obtain the KP-ABE variant of this scheme.

E.5 Decentralized "unbounded" CP-ABE scheme

We also give a decentralized variant of the unbounded single-authority CP-ABE scheme by Rouselakis and Waters [48].

Definition 23 (Decentralized "unbounded" CP-ABE). *We define the GPES as follows.*

- $\mathrm{Param}(\mathrm{par}) \rightarrow (n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b})$*: Let* $\{\mathcal{A}_l\}_{[n_{\mathrm{aut}}]}$ *be the authorities. We set* $n_\alpha = n_{\mathrm{aut}}$ *and* $n_b = 4n_{\mathrm{aut}}$*, where* $\boldsymbol{\alpha} = \{\alpha_l\}_{l \in [n_{\mathrm{aut}}]}$*, and* $\mathbf{b} = (\{b_l, b'_l, b_{l,0}, b_{l,1}\}_{l \in [n_{\mathrm{aut}}]})$.
- $\mathrm{EncKey}((\mathcal{S}, \tilde{\rho}_{\mathcal{S}}), p) \rightarrow (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, (\mathcal{S}, \tilde{\rho}_{\mathcal{S}})))$*: We set* $m_1 = |\mathcal{S}| + 1$ *and* $m_2 = 0$*, where* $\mathbf{k} = (\{k_{1,l} = \alpha_l + r_{\mathrm{GID}} b_l\}_{l \in \tilde{\rho}([n_1])}, \{k_{2,\mathrm{att}} = r_{\mathrm{GID}} b'_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att})} + r_{\mathrm{att}}(b_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),0} + x_{\mathrm{att}} b_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),1})\}_{\mathrm{att} \in \mathcal{S}})$ *such that* x_{att} *is the integer representation of* att *in* $\mathbb{Z}_p$*, and* $\mathcal{F}(r_{\mathrm{GID}}) = 1$.
- $\mathrm{EncCt}((\mathbf{A}, \rho, \tilde{\rho}), p) \rightarrow (w_1, w_2, w'_2, c_M, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, (\mathbf{A}, \rho, \tilde{\rho})), \mathbf{c}'(\mathbf{s}, \tilde{\mathbf{s}}, \boldsymbol{\alpha}, (\mathbf{A}, \rho, \tilde{\rho})))$*: We set* $w_1 = n_1$*,* $w_2 = n_2 - 1$*,* $w'_2 = n_2 - 1$*,* $C_M = \tilde{s}$*,*

  $$\mathbf{c} = (\{c_{1,j} = \mu_j + s_j(b_{\tilde{\rho}(j)} + b'_{\tilde{\rho}(j)}), c_{2,j} = s_j(b_{\tilde{\rho}(j),0} + \rho(j) b_{\tilde{\rho}(j),1})\}_{j \in [n_1]}),$$

  and $\mathbf{c}' = (\{\lambda_j + \alpha_{\tilde{\rho}(j)} s_j\}_{j \in [n_1]})$*, where* $\lambda_j = A_{j,1}\hat{s} + \sum_{k \in [2,n_2]} A_{j,k}\hat{v}_k$ *and* $\mu_j = \sum_{k \in [2,n_2]} A_{j,k}\hat{v}'_k$.

Lemma 7. *The GPES in Definition 23 satisfies the special selective symbolic property.*

Proof. Let $\mathfrak{C} \subseteq [n_{\mathrm{aut}}]$ be a set of corrupted authorities, and set $d_1 = n_1$ and $d_2 = (n_1 + 1)n_2$. For the simple representation of column indices, we use $(1, k)$ for all $k \in [n_2]$ and $(2, j, k)$ for all $j \in [n_1]$ and $k \in [n_2]$. These are mapped injectively in the interval $[d_2]$.

- $\mathrm{EncB}((\mathbf{A}, \rho, \tilde{\rho})) \rightarrow (\{\mathbf{a}_l, \mathbf{B}_l, \mathbf{B}'_l, \mathbf{B}_{l,0}, \mathbf{B}_{l,1}\}_{l \in [n_{\mathrm{aut}}]})$, $\mathbf{a}_l = \mathbf{0}^{d_1}$ and $\mathbf{B}_l, \mathbf{B}'_l = \mathbf{0}^{d_1 \times d_2}$ for all $l \in \mathfrak{C}$, and let $\mathbf{v} \in \mathbb{Z}_p^{n_2}$ (with $v_1 = 1$) be the vector orthogonal to each row $j \in \tilde{\rho}^{-1}(\mathfrak{C})$ associated with a corrupted authority. For all $l \in [n_{\mathrm{aut}}] \setminus \mathfrak{C}$, we set:

  $$\mathbf{a}_l = \sum_{j \in \tilde{\rho}^{-1}(l), k \in [n_2]} \mathbf{1}_j^{d_1}, \qquad \mathbf{B}_l = \sum_{j \in \tilde{\rho}^{-1}(l), k \in [n_2]} A_{j,k} v_k \mathbf{1}_{j,(1,1)}^{d_1 \times d_2},$$

  $$\mathbf{B}'_l = \sum_{j \in \tilde{\rho}^{-1}(l), k \in [n_2]} A_{j,k} \mathbf{1}_{j,(1,k)}^{d_1 \times d_2}, \qquad \mathbf{B}_{l,1} = \sum_{j \in [n_1], k \in [n_2]} A_{j,k} \mathbf{1}_{j,(2,j,k)}^{d_1 \times d_2},$$

  $$\mathbf{B}_{l,0} = -\sum_{j \in [n_1], k \in [n_2]} A_{j,k} \rho(j) \mathbf{1}_{j,(2,j,k)}^{d_1 \times d_2},$$

- $\mathrm{EncR}((\mathbf{A}, \rho, \tilde{\rho}), (\mathcal{S}, \tilde{\rho}_{\mathcal{S}})) \rightarrow (\mathbf{r}_{\mathrm{GID}}, \{\mathbf{r}_{\mathrm{att}}\}_{\mathrm{att} \in \mathcal{S}})$, where

$$\mathbf{r}_{\mathrm{GID}} = \sum_{k \in [n_2]} w_k \overline{\mathbf{1}}^{d_2}_{(1,k)}, \qquad \mathbf{r}_{\mathrm{att}} = \sum_{j \in \overline{\Upsilon}, k \in [n_2]} \frac{w_k \overline{\mathbf{1}}^{d_2}_{(2,j,k)}}{x_{\mathrm{att}} - \rho(j)},$$

  such that $\mathbf{w}$ (with $w_1 = 1$) is the vector orthogonal to all $\mathbf{A}_j$ with $j \in \Upsilon$ (Definition 1), and $\overline{\Upsilon} = [n_1] \setminus \Upsilon$.
- $\mathrm{EncS}((\mathbf{A}, \rho, \tilde{\rho})) \rightarrow (\{\mathbf{s}_j\}_{j \in [n_1]}, \{\hat{\mathbf{v}}_k, \hat{\mathbf{v}}'_k\}_{k \in [2,n_2]}, \tilde{\mathbf{s}})$, where

$$\tilde{\mathbf{s}} = 1, \qquad \mathbf{s}_j = \mathbf{1}^{d_1}_j, \qquad \hat{\mathbf{v}}_k = v_k, \qquad \hat{\mathbf{v}}'_k = \overline{\mathbf{1}}^{d_2}_{(1,k)} + v_k \overline{\mathbf{1}}^{d_2}_{(1,1)}.$$

Note that, in this proof, we require the knowledge of the entire key set $\mathcal{S}$ for the substitution vector of $\mathbf{r}_{\mathrm{GID}}$. Therefore, when instantiating it with the multi-authority compiler, the scheme is statically secure. □

E.6 Decentralized identity-based broadcast encryption scheme

We also give a decentralized identity-based broadcast encryption scheme with short ciphertexts based on TinyABE [54].

Definition 24 (Decentralized IBBE). *We define the GPES as follows.*

- $\mathrm{Param}(\mathrm{par}) \rightarrow (n_\alpha, n_b, \boldsymbol{\alpha}, \mathbf{b})$*: Let* $\{\mathcal{A}_l\}_{[n_{\mathrm{aut}}]}$ *be the authorities. We set* $n_\alpha = n_{\mathrm{aut}}$ *and* $n_b = 2n$*, where* $n \in \mathbb{N}$*, where* $\boldsymbol{\alpha} = \{\alpha_l\}_{l \in [n_{\mathrm{aut}}]}$*, and* $\mathbf{b} = (\{b_{i,0}, b_{i,1}\}_{i \in [n]})$*. We set* $\mathcal{F}(b_{i,0}) = 1$ *and* $\mathcal{F}(b_{i,1}) = 2$*, where the hash expects* $i \in [n]$ *as input.*
- $\mathrm{EncKey}((y, l), p) \rightarrow (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \boldsymbol{\alpha}, \mathbf{b}, (y, l)))$*: We set* $m_1 = n$ *and* $m_2 = 0$*, where* $\mathbf{k} = (\{k_i = \alpha_l + r_i(b_{i,0} + y b_{i,1})\}_{i \in [n]})$ *such that* y *is the integer representation of the identity of the user in* $\mathbb{Z}_p$*.*
- $\mathrm{EncCt}((\mathcal{S}, \tilde{\rho}), p) \rightarrow (w_1, w_2, w'_2, c_M, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}, (\mathcal{S}, \tilde{\rho})), \mathbf{c}'(\mathbf{s}, \tilde{\mathbf{s}}, \boldsymbol{\alpha}, (\mathcal{S}, \tilde{\rho})))$*: Let* $n' = \left\lceil \frac{|\mathcal{S}|}{n} \right\rceil$*, and let* $\tau \colon \mathcal{S} \rightarrow [n']$ *be a function that maps the identities in* $\mathcal{S}$ *to* n' *partitions such that* $|\tau^{-1}(j)| \leq n$ *for all* $j \in [n']$*. Let* $\hat{\tau} \colon \mathcal{S} \rightarrow [n]$ *be a function that maps the identities in* $\mathcal{S}$ *that are assigned to the same partition to unique integers in* $[n]$*, i.e.,* $\hat{\tau}$ *is injective on* $\tau^{-1}(l)$ *for all* $l \in [n']$*. Let* $\mathcal{S}_j = \{x \in \mathcal{S} \mid \tau(x) = j\}$*. We set* $w_1 = n'$*,* $w_2 = 0$*,* $w'_2 = \sum_{j \in [n']} |\mathcal{N}_j|$*,* $C_M = \tilde{s}$*,*

$$\mathbf{c} = \left(\sum_{j \in [n']} s_j \sum_{x \in \mathcal{S}_j} (b_{\hat{\tau}(x),0} + x b_{\hat{\tau}(x),1}) \right),$$

  and $\mathbf{c}' = (\{\tilde{s} + \alpha_l s_j\}_{j \in [n'], l \in \mathcal{N}_j})$*, where* $\mathcal{N}_j = \{l \in [n_{\mathrm{aut}}] \mid x \in \mathcal{S}_j \wedge \tilde{\rho}(x) = l\}$*.*

Lemma 8. *The GPES in Definition 24 satisfies the special selective symbolic property.*

Proof. Let $\mathfrak{C} \subseteq [n_{\mathrm{aut}}]$ be a set of corrupted authorities, and set $d_1 = n'$ and $d_2 = n'$. For all $i \in [n]$, let $\chi_i = \{x \in \mathcal{S} \mid \hat{\tau}(x) = i\}$ be the set of identities in $\mathcal{S}$ that are mapped to $i \in [n]$ with $\hat{\tau}$.

- EncB$((\mathcal{S}, \tilde{\rho})) \rightarrow (\{\mathbf{a}_l, \mathbf{B}_{i,0}, \mathbf{B}_{i,1}\}_{l \in [n_{\text{aut}}], i \in [n]})$, $\mathbf{a}_l = \mathbf{0}^{d_1}$ for all $l \in \mathfrak{C}$. For all $l \in [n_{\text{aut}}] \setminus \mathfrak{C}$ and $i \in [n]$, we set $\hat{\mathcal{N}_l} = \{j \in [n'] \mid l \in \mathcal{N}_j\}$ and:
$$\mathbf{a}_l = \sum_{j \in \hat{\mathcal{N}_l}} \mathbf{1}_j^{d_1}, \quad \mathbf{B}_{i,1} = \sum_{x \in \chi_i} \mathbf{1}_{\tau(x),\tau(x)}^{d_1 \times d_2}, \quad \mathbf{B}_{i,0} = -\sum_{x \in \chi_i} x \mathbf{1}_{\tau(x),\tau(x)}^{d_1 \times d_2}.$$
- EncR$((\mathcal{S}, \tilde{\rho}), (y, l)) \rightarrow (\{\mathbf{r}_i\}_{i \in [n]})$, where
$$\mathbf{r}_i = \sum_{x \in \chi_i} \frac{\overline{\mathbf{1}}_{\tau(x)}^{d_2}}{x - y}.$$
- EncS$((\mathcal{S}, \tilde{\rho}) \rightarrow (\{\mathbf{s}_j\}_{j \in [n']}, \tilde{\mathbf{s}})$, where
$$\tilde{\mathbf{s}} = 1, \quad \mathbf{s}_j = \mathbf{1}_j^{d_1}.$$

□

Remark 7. The decryption costs are quadratic in the size of $\mathcal{S}$. To lower the decryption costs, we can also split the ciphertext in parts, i.e., set $\mathbf{c} = (\{c_j = \sum_{j \in [n']} s_j \sum_{x \in \mathcal{S}_j} (b_{\hat{\tau}(x),0} + x b_{\hat{\tau}(x),1})\}_{j \in [n']})$. Then, the decryption costs are only quadratic in n. However, this increases the encryption costs and the ciphertext size.

F Verification of the proof of Lemma 2

We show that, indeed, the polynomials evaluate to $\mathbf{0}$. Note that the evaluations for $k_{1,l}$, $c_{1,j}$ and c'_j are virtually the same as in Appendix G. For the other key polynomials, we have the following evaluations. For $k_{2,(\text{lab},\text{att})}$, we first compute $\mathbf{B}_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab},0} + x_{\text{att}} \mathbf{B}_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab},1}$, which is

$$\sum_{j \in \Psi_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab}}, k \in [n_2]} A_{j,k} \left(\mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2} - \rho(j) \mathbf{1}_{\tau(j),(2,j,k,\text{lab})}^{d_1 \times d_2} \right)$$
$$- \sum_{j \in \overline{\Psi}_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab}}, k \in [n_2]} \rho(j) A_{j,k} \mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2}$$
$$+ x_{\text{att}} \left(\sum_{j \in \Psi_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab}}, k \in [n_2]} A_{j,k} \mathbf{1}_{\tau(j),(2,j,k,\text{lab})}^{d_1 \times d_2} + \sum_{j \in \overline{\Psi}_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab}}, k \in [n_2]} A_{j,k} \mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2} \right)$$
$$= \sum_{j \in \Psi_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab}}, k \in [n_2]} A_{j,k} \left(\mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2} + (x_{\text{att}} - \rho(j)) \mathbf{1}_{\tau(j),(2,j,k,\text{lab})}^{d_1 \times d_2} \right)$$
$$+ \sum_{j \in \overline{\Psi}_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab}}, k \in [n_2]} (x_{\text{att}} - \rho(j)) A_{j,k} \mathbf{1}_{\tau(j),(1,k)}^{d_1 \times d_2}.$$

Then,

$$k_{2,(\text{lab},\text{att})} = r_{\tilde{\rho}_{\mathcal{S}}(\text{att})} (b_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab},0} + x_{\text{att}} b_{\tilde{\rho}_{\mathcal{S}}(\text{att}),\text{lab},1})$$

$$= \left(\sum_{k \in [n_2]} w_k \overline{\mathbf{1}}^{d_2}_{(1,k)} + \sum_{j \in \Psi_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att})} \cap \overline{\Upsilon}, k \in [n_2], (\mathrm{lab}, \mathrm{att}) \in \mathcal{S} | \mathrm{lab} = \rho_{\mathrm{lab}}(j)} \frac{w_k}{\rho(j) - x_{\mathrm{att}}} \overline{\mathbf{1}}^{d_2}_{(2,j,k,\mathrm{lab})} \right)$$

$$\left(\sum_{j \in \Psi_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),\mathrm{lab}}, k \in [n_2]} A_{j,k} \left(\mathbf{1}^{d_1 \times d_2}_{\tau(j),(1,k)} + (x_{\mathrm{att}} - \rho(j)) \mathbf{1}^{d_1 \times d_2}_{\tau(j),(2,j,k,\mathrm{lab})} \right) \right.$$

$$\left. + \sum_{j \in \overline{\Psi}_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),\mathrm{lab}}, k \in [n_2]} (x_{\mathrm{att}} - \rho(j)) A_{j,k} \mathbf{1}^{d_1 \times d_2}_{\tau(j),(1,k)} \right)$$

$$= \sum_{j \in \Psi_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),\mathrm{lab}}, k \in [n_2]} A_{j,k} w_k \left(\mathbf{1}^{d_1}_{\tau(j)} + \frac{(x_{\mathrm{att}} - \rho(j))}{\rho(j) - x_{\mathrm{att}}} \mathbf{1}^{d_1}_{\tau(j)} \right)$$

$$+ \sum_{j \in \overline{\Psi}_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),\mathrm{lab}}, k \in [n_2]} (x_{\mathrm{att}} - \rho(j)) A_{j,k} w_k \mathbf{1}^{d_1}_{\tau(j)},$$

which is either $\mathbf{0}^{d_1}$, because $\mathbf{A}_j \mathbf{w}^{\intercal} = 0$ (which is the case if either $(\rho_{\mathrm{lab}}(j), \rho(j)) \in \mathcal{S}$ if $\rho'(j) = 1$ or $(\rho_{\mathrm{lab}}(j), \mathrm{att}) \in \mathcal{S}$ with $\mathrm{att} \neq \rho(j)$ if $\rho'(j) = 0$), or it holds for $j \in \Psi_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),\mathrm{lab}}$ that $\rho(j) \neq x_{\mathrm{att}}$, and for $j \in \overline{\Psi}_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att}),\mathrm{lab}}$ that $x_{\mathrm{att}} = \rho(j)$, meaning that those j for which $\mathbf{A}_j \mathbf{w}^{\intercal} \neq 0$, we have

$$\sum_{k \in [n_2]} A_{j,k} w_k \left(\mathbf{1}^{d_1}_{\tau(j)} + \underbrace{\frac{(x_{\mathrm{att}} - \rho(j))}{\rho(j) - x_{\mathrm{att}}}}_{=-1} \mathbf{1}^{d_1}_{\tau(j)} \right) = \mathbf{0}^{d_1}$$

$$\sum_{k \in [n_2]} \underbrace{(x_{\mathrm{att}} - \rho(j))}_{=0} A_{j,k} w_k \mathbf{1}^{d_1}_{\tau(j)} = \mathbf{0}^{d_1}.$$

For the ciphertext polynomials, we have the following evaluations. We first note that the combination $s'_{\tau(j)}$ and $b_{\tilde{\rho}(j),\rho_{\mathrm{lab}}(j),0} + \rho(j) b_{\tilde{\rho}(j),\rho_{\mathrm{lab}}(j),1}$ yields

$$- \left(\sum_{k \in [n_2]} A_{j,k} \left(\overline{\mathbf{1}}^{d_2}_{(1,k)} - \rho(j) \overline{\mathbf{1}}^{d_2}_{(2,j,k,\rho_{\mathrm{lab}}(j))} \right) + \rho(j) \sum_{k \in [n_2]} A_{j,k} \overline{\mathbf{1}}^{d_2}_{(2,j,k,\rho_{\mathrm{lab}}(j))} \right)$$

$$= - \sum_{k \in [n_2]} A_{j,k} \overline{\mathbf{1}}^{d_2}_{(1,k)}$$

if $\rho'(j) = 1$, and otherwise

$$- \sum_{k \in [n_2]} \rho(j) A_{j,k} \overline{\mathbf{1}}^{d_2}_{(1,k)} + \rho(j) \sum_{k \in [n_2]} A_{j,k} \overline{\mathbf{1}}^{d_2}_{(1,k)} = \mathbf{0}^{d_2},$$

from which it directly follows that for $j \in \overline{\Psi}$, $c_{3,j} = \mathbf{0}^{d_2}$. For $j \in \Psi$, we have

$$c_{2,j} = s_j b'_{\tilde{\rho}(j)} + s'_{\tau(j)} (b_{\mathrm{lab},0} + x_{\rho(j)} b_{\rho_{\mathrm{lab}}(j),1})$$

$$
\begin{aligned}
&= \mathbf{1}_j^{d_1} \left(\sum_{j\in\tilde{\rho}^{-1}(\tilde{\rho}(j)),k\in[n_2]} A_{j,k}\mathbf{1}^{d_1\times d_2}_{j,(1,k)} \right) - \sum_{k\in[n_2]} A_{j,k}\overline{\mathbf{1}}^{d_2}_{(1,k)} \\
&\quad = \sum_{k\in[n_2]} A_{j,k}\overline{\mathbf{1}}^{d_2}_{(1,k)} - \sum_{k\in[n_2]} A_{j,k}\overline{\mathbf{1}}^{d_2}_{(1,k)} = \mathbf{0}^{d_2}.
\end{aligned}
$$

For $j \in \overline{\Psi}$, we have

$$
\begin{aligned}
c_{2,j} &= s_j b'_{\tilde{\rho}(j)} + s'_{\tau(j)} b_{\tilde{\rho}(j),\rho_{\mathrm{lab}}(j),1} \\
&= \mathbf{1}_j^{d_1} \left(\sum_{j\in\tilde{\rho}^{-1}(\tilde{\rho}(j)),k\in[n_2]} A_{j,k}\mathbf{1}^{d_1\times d_2}_{j,(1,k)} \right) - \sum_{k\in[n_2]} A_{j,k}\overline{\mathbf{1}}^{d_2}_{(1,k)} \\
&= \sum_{k\in[n_2]} A_{j,k}\overline{\mathbf{1}}^{d_2}_{(1,k)} - \sum_{k\in[n_2]} A_{j,k}\overline{\mathbf{1}}^{d_2}_{(1,k)} = \mathbf{0}^{d_2}.
\end{aligned}
$$

G Verification of the proof of Lemma 3

We show that, indeed, the polynomials evaluate to $\mathbf{0}$. For the key polynomials, we have the following evaluations.

$$
\begin{aligned}
k_{1,l} &= \alpha_l + r_{\mathrm{GID}} b_l + r_l b'_l \\
&= \sum_{j\in\tilde{\rho}^{-1}(l),k\in[n_2]} A_{j,k} v_k \mathbf{1}_j^{d_1} \\
&\quad + \left(-\overline{\mathbf{1}}_1^{d_2} + \sum_{k\in[2,n_2]} w_k \overline{\mathbf{1}}_k^{d_2} \right) \left(\sum_{j\in\tilde{\rho}^{-1}(l),k\in[2,n_2]} A_{j,k}(\mathbf{1}^{d_1\times d_2}_{j,k} + v_k \mathbf{1}^{d_1\times d_2}_{j,1}) \right) \\
&\quad - \left(\sum_{k\in[n_2]} w_k \overline{\mathbf{1}}_k^{d_2} \right) \left(\sum_{j\in\tilde{\rho}^{-1}(l),k\in[n_2]} A_{j,k}\mathbf{1}^{d_1\times d_2}_{j,k} \right) \\
&= \sum_{j\in\tilde{\rho}^{-1}(l),k\in[n_2]} A_{j,k} v_k \mathbf{1}_j^{d_1} - \sum_{j\in\tilde{\rho}^{-1}(l),k\in[2,n_2]} A_{j,k} v_k \mathbf{1}_j^{d_1} + \sum_{j\in\tilde{\rho}^{-1}(l),k\in[2,n_2]} A_{j,k} w_k \mathbf{1}_j^{d_1} \\
&\quad - \sum_{j\in\tilde{\rho}^{-1}(l),k\in[n_2]} A_{j,k} w_k \mathbf{1}_j^{d_1} \\
&= \sum_{j\in\tilde{\rho}^{-1}(l)} A_{j,1} v_1 \mathbf{1}_j^{d_1} - \sum_{j\in\tilde{\rho}^{-1}(l)} A_{j,1} w_1 \mathbf{1}_j^{d_1} = \mathbf{0}^{d_1},
\end{aligned}
$$

because $v_1 = w_1 = 1$. Then, for att $\notin \rho([n_1])$, we trivially have that $k_{2,\mathrm{att}}$ evaluates to $\mathbf{0}^{d_1}$, and for att $\in \rho([n_1])$, we have that $\mathbf{A}_j \mathbf{w}^\intercal = 0$, so

$$
k_{2,\mathrm{att}} = r_{\tilde{\rho}_{\mathcal{S}}(\mathrm{att})} b_{\mathrm{att}} = - \left(\sum_{k\in[n_2]} w_k \overline{\mathbf{1}}_k^{d_2} \right) \left(\sum_{j\in\rho^{-1}(\mathrm{att}),k\in[n_2]} A_{j,k}\mathbf{1}^{d_1\times d_2}_{\tau(j),k} \right)
$$

$$= \sum_{j\in\rho^{-1}(\mathrm{att}),k\in[n_2]} A_{j,k} w_k \mathbf{1}^{d_1}_{\tau(j)} = \sum_{j\in\rho^{-1}(\mathrm{att})} \mathbf{A}_j \mathbf{w}^{\intercal} \mathbf{1}^{d_1}_{\tau(j)} = \mathbf{0}^{d_1}.$$

For the ciphertext polynomials, we have the following evaluations.

$$c_{1,j} = \mu_j + s_j b_{\tilde{\rho}(j)} = \sum_{k\in[2,n_2]} A_{j,k} \hat{v}'_k + s_j b_{\tilde{\rho}(j)}$$

$$= \sum_{k\in[2,n_2]} A_{j,k} \left(\overline{\mathbf{1}}^{d_2}_k + v_k \overline{\mathbf{1}}^{d_2}_1\right) - \mathbf{1}^{d_1}_j \sum_{j\in\tilde{\rho}^{-1}(\tilde{\rho}(j)),k\in[2,n_2]} A_{j,k}(\mathbf{1}^{d_1\times d_2}_{j,k} + v_k \mathbf{1}^{d_1\times d_2}_{j,1})$$

$$= \sum_{k\in[2,n_2]} A_{j,k} \left(\overline{\mathbf{1}}^{d_2}_k + v_k \overline{\mathbf{1}}^{d_2}_1\right) - \sum_{k\in[2,n_2]} A_{j,k}(\overline{\mathbf{1}}^{d_2}_k + v_k \overline{\mathbf{1}}^{d_2}_1) = \mathbf{0}^{d_2}.$$

$$c_{2,j} = s_j b'_{\tilde{\rho}(j)} + s'_{\tau(j)} b_{\rho(j)}$$

$$= -\mathbf{1}^{d_1}_j \left(\sum_{j\in\tilde{\rho}^{-1}(l),k\in[n_2]} A_{j,k} \mathbf{1}^{d_1\times d_2}_{j,k} \right) + \mathbf{1}^{d_1}_{\tau(j)} \left(\sum_{j\in\rho^{-1}(\mathrm{att}),k\in[n_2]} A_{j,k} \mathbf{1}^{d_1\times d_2}_{\tau(j),k} \right)$$

$$= -\sum_{k\in[n_2]} A_{j,k} \overline{\mathbf{1}}^{d_2}_k + \sum_{k\in[n_2]} A_{j,k} \overline{\mathbf{1}}^{d_2}_k = \mathbf{0}^{d_2}.$$

$$c'_j = \lambda_j + \alpha_{\tilde{\rho}(j)} s_j = A_{j,1}\tilde{s} + \sum_{k\in[2,n_2]} A_{j,k} \hat{v}_k + \alpha_{\tilde{\rho}(j)} s_j$$

$$= A_{j,1} + \sum_{k\in[2,n_2]} A_{j,k} v_k - \mathbf{1}^{d_1}_j \left(\sum_{j\in\tilde{\rho}^{-1}(\tilde{\rho}(j)),k\in[n_2]} A_{j,k} v_k \mathbf{1}^{d_1}_j \right)$$

$$= \sum_{k\in[n_2]} A_{j,k} v_k - \sum_{k\in[n_2]} A_{j,k} v_k = 0.$$

H Comparison of decentralized schemes

We compare the qualitative features of our newly proposed decentralized schemes with existing multi-authority schemes in Table 1. It shows that our scheme in Definition 16 is the first non-monotone large-universe scheme of this kind that supports unbounded policies.

Table 1: Comparison of the properties of all pairing-based decentralized CP-ABE schemes supporting monotone span programs. For each scheme that is decentralized (i.e., it requires no coordination among the authorities), we consider whether the scheme supports negations in the policies, large universes (LU) and unbounded policies (UP) (i.e., it supports unbounded re-use of attributes in the policies and unbounded policy lengths). We also consider whether the scheme is fully, selectively or statically secure and whether it is secure under a non-parametrized assumption (NPA).

Scheme	**Negations**	**LU**	**UP**	**Security**	**NPA**
LW11 [41]	✗	✗	✗	Full	✓
OT13,OT20 [44,45]	✓	✓	✗	Full	✓
RW15 [49]	✗	✓	✓	Static	✗
DKW21 [30]	✗	✗	✗	Static	✓
AG21 [10]	✗	✓	✗	Selective	✓
DKW22 [31]	✗	✗	✗	Full	✓
Definition 16	✓	✓	✓	Static	✗
Definition 19	✗	✓	✓	Static	✗
Definition 23	✗	✓	✓	Static	✗

Table of Contents