Generic Transformations of Predicate Encodings: Constructions and Applications

Miguel Ambrona, Gilles Barthe, and Benedikt Schmidt

IMDEA Software Institute, Madrid, Spain
{miguel.ambrona,gilles.barthe,benedikt.schmidt}@imdea.org

Abstract. Predicate encodings (Wee, TCC 2014; Chen, Gay, Wee, EUROCRYPT 2015), are symmetric primitives that can be used for building predicate encryption schemes. We give an algebraic characterization of the notion of privacy from predicate encodings, and explore several of its consequences. Specifically, we propose more efficient predicate encodings for boolean formulae and arithmetic span programs, and generic optimizations of predicate encodings. We define new constructions to build boolean combination of predicate encodings. We formalize the relationship between predicate encodings and pair encodings (Attrapadung, EUROCRYPT 2014), another primitive that can be transformed generically into predicate encryption schemes, and compare our constructions for boolean combinations of pair encodings with existing similar constructions from pair encodings. Finally, we demonstrate that our results carry to tag-based encodings (Kim, Susilo, Guo, and Au, SCN 2016).

1 Introduction

Predicate Encryption (PE) [11, 22] is a form of public-key encryption that supports fine-grained access control for encrypted data. In predicate encryption, everyone can create ciphertexts while keys can only be created by the master key owner. Predicate encryption schemes use predicates to model (potentially complex) access control policies, and attributes are attached to both ciphertexts and secret keys. A predicate encryption scheme for a predicate P guarantees that decryption of a ciphertext ct_x with a secret key sk_y is allowed if and only if the attribute x associated to the ciphertext ct and the attribute y associated to the secret key sk verify the predicate P, i.e. $\mathsf{P}(x, y) = 1$. Predicate encryption schemes exist for several useful predicates, such as Zero Inner Product Encryption (ZIPE), where attributes are vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ and the predicate $\mathsf{P}(\boldsymbol{x}, \boldsymbol{y})$ is defined as $\boldsymbol{x}^\top \boldsymbol{y} = 0$. Predicate encryption subsumes several previously defined notions of public-key encryption. For example, Identity-Based Encryption (IBE) [30] can be obtained by defining $\mathsf{P}(x, y)$ as $x = y$ and Attribute-Based Encryption (ABE) [29] can be obtained similarly. More concretely, for Key-Policy ABE (KP-ABE), the attribute x is a boolean vector, the attribute y is a boolean function, and the predicate $\mathsf{P}(x, y)$ is defined as $y(x) = 1$. For Ciphertext-Policy ABE (CP-ABE), the roles of the attributes x and y are swapped.

Modular approaches for PE. In 2014, two independent works by Wee [33] and Attrapadung [4] proposed generic and unifying frameworks for obtaining efficient fully secure PE schemes for a large class of predicates. Both works use the dual system methodology introduced by Lewko and Waters [24, 32] and define a compiler that takes as input a relatively simple symmetric primitive and produces a fully secure PE construction. Wee introduced so-called *predicate encodings*, an information-theoretic primitive inspired from linear secret sharing. Attrapadung introduced so-called *pair encodings* and provided computational and information-theoretic security notions. These approaches greatly simplify the construction and analysis of predicate encryption schemes and share several advantages. First, they provide a good trade-off between *expressivity* and *performance*, while the security relies on standard and well studied assumptions. Second, they unify existing constructions into a single framework, i.e., previous PE constructions can be seen as instantiations of these new compilers with certain encodings. Third, building PE schemes by analyzing and building these simpler encodings is much easier than building PE schemes directly. Compared to full security for PE, the encodings must verify much weaker security requirements. The power of pair and predicate encodings is evidenced by the discovery of new constructions and efficiency improvements over existing ones. However, both approaches were designed over *composite order* bilinear groups. In 2015, Chen, Gay and Wee [12] and Attrapadung [5] respectively extended the predicate encoding and pair encoding compiler to the *prime order* setting. Next, Agrawal and Chase [1] improved on Attrapadung's work by relaxing the security requirement on *pair encodings* and thus, capturing new constructions. In addition, their work also brings both generic approaches closer together, because like in [12], the new compiler relies (in a black-box way) on Dual System Groups (DSG) [13, 14]. Additionally, Kim, Susilo, Guo, and Au [19] recently introduced a new generic framework for modular design of predicate encryption that improves on the performance of existing compilers. Their core primitive, *tag-based encodings*, is very similar to *predicate encodings*.

1.1 Our contributions

We pursue the study of predicate encodings and establish several general results and new constructions that broaden their scope and improve their efficiency. We also compare predicate encodings to pair and tag-based encodings.

Predicate encodings. We show that the information-theoretic definition of α-privacy used in [12, 33] is equivalent to an algebraic statement (furthermore independent of α) about the existence of solutions for a linear system of equations. Leveraging this result, we prove a representation theorem for predicate encodings: every triple of encoding functions implicitly defines a unique predicate for which it is a valid predicate encoding. Conversely, every predicate P that admits a predicate encoding is logically equivalent to the implicit predicate induced by its encoding functions. Moreover, our algebraic definition of privacy simplifies all subsequent results in the paper.

First, we define a generic optimization of predicate encodings that often leads to efficiency improvements and reduce the number of required group elements in keys and ciphertexts. We prove the soundness of the transformations and validate their benefits experimentally on examples from [12, 33]; we successfully apply these simplifications to reduce the size of keys and ciphertexts by up to 50% and to reduce the number of group operations needed in some of the existing encodings.

Second, we define generic methods for combining predicate encodings. We provide encoding transformations for the *disjunction*, *conjunction* and *negation* of predicates, and for the *dual* predicate.

Tag-based encodings. We show that our results on predicate encodings generalize to tag-based encodings. In particular, we give a purely algebraic characterization of the hiding property of tag-based encodings. Moreover, we demonstrate that the hiding property can be strengthened without any loss of generality, by requiring equality rather than statistical closeness of distributions.

Comparison of encodings. We compare the expressivity of the three core primitives (*predicate encodings*, *pair encodings* and *tag-based encodings*) corresponding to the three different modular frameworks. We provide an embedding that produces an information-theoretical pair encoding from every predicate encoding. Then, we use this encoding to compare our constructions to build boolean combination of predicate encodings with similar constructions for pair encodings that were introduced by [4].

In addition, we provide a transformation[1] from tag-based encodings into predicate encodings.

New constructions. We develop several new constructions of predicate encodings and predicate encryption:

- **Combining predicates.** We show how to combine our results to build *Dual-Policy Attribute-Based Encryption (DP-ABE)* [7, 8] in the frameworks of predicate encodings and tag-based encodings (Section 6.1). Additionally, we consider the idea of combining arbitrary encodings with a *broadcast encryption* encoding to achieve direct revocation of keys. The former encoding takes care of revocation, while the latter encodes the desired access structure.
- **Improved predicate encodings.** We provide new instances of predicate encodings that improve on best known predicate encodings proposed in [12] and have additional properties. (Section 6.2).
- **Extra features.** Finally, we show how to construct a weakly attribute-hiding predicate encoding for boolean formulas and how to enhance any predicate encoding with support for delegation. (Section 6.3).

[1] this transformation has side conditions, thus it is not universal, but all existing tag-based encodings (except one) satisfy these side conditions

Implementation and evaluation. We implement a general library for predicate encryption with support for the predicate encoding and pair encoding frameworks. Our library uses the Relic-Toolkit [3] for pairings with a 256-bits Barreto-Naehrig Curve [9]. We use our library for validating our constructions; experimental results are presented in the relevant sections. All the experiments were executed on a 8-core machine with 2.40GHz Intel Core i7-3630QM CPU and 8GB of RAM. Our scalability experiments show that predicate encodings can be used for real applications. The code is publicly available and open source[2].

1.2 Prior work

Predicate encodings have been introduced in [33] and we use a refined version that is defined in [12] as our starting point. Both variants use an information-theoretic definition of the hiding while we show that there is an equivalent algebraic definition. Another related work is [17], initiating a systematic study of the communication complexity of the so-called conditional secret disclosure primitive, which is closely related to predicate encodings.

Other works also optimize existing predicate encryption schemes, for example many works focus on going from composite order constructions to the more efficient prime order ones [5, 12, 23]. In [12] they also propose performance improvements on dual system groups. We believe our optimizations via predicate encodings complement other possible enhancements of predicate encryption.

Boolean combinations of predicates have also been considered in the setting of pair encodings. Attrapadung [7, 8] proposes generic transformations for conjunction and for the dual predicate, but neither for negation nor disjunction. We propose new transformations for conjunction and dual in the framework of predicate encodings and we also deal with negation and disjunction.

The main advantage of DP-ABE is the possibility of considering policies over *objective* attributes (associated to data) and policies over *subjective* attributes (associated to user credentials) at the same time. DP-ABE has been considered by Attrapadung in the pair encoding framework [7, 8]. To the best of our knowledge, we are the first to provide DP-ABE in the predicate encoding and tag-based encoding frameworks.

Revocation is a desirable property for PE and ABE schemes that has also been considered by many works in the literature. Revocation allows to invalidate a user's secret key in such a way that it becomes useless, even if its associated attribute satisfies the policy associated to the ciphertext. Some attempts [28] propose *indirect* revocation that requires that the master secret owner periodically updates secret keys for non-revoked users. Other attempts achieve *direct* revocation [6, 20, 26, 27], but either rely on strong assumptions or provide only selectively security. Our construction not only allows to achieve revocation in a *fully secure* framework, but it allows to add revocation to arbitrary predicate encodings.

[2] source code at https://github.com/miguel-ambrona/abe-relic

Policy hiding is another property of PE, and ABE in particular, that has been broadly studied. In this context, policies associated to ciphertexts are not attached to them and therefore, unauthorized users will only learn the fact that their key does not satisfy the policy, but nothing else. Policy Hiding has been considered in several works [11, 22]. The security of our construction improves on earlier works, thanks to the compiler from [12]. Our observation extends the expressivity of attribute-hiding predicate encryption for ZIPE proposed in [12] to support policy-hiding for boolean formulas.

In [12], the authors introduce the notion of *spatial encryption* predicate encodings. We generalize this notion and introduce a transformation that makes delegation possible for every predicate encoding.

Several works evaluate the suitability of ABE for different applications. For example, ABE has been used and benchmarked to enforce privacy of Electronic Medical Records (EMR) [2], in a system where healthcare organizations export EMRs to external storage locations. Other examples are Sieve [31] or Streamforce [15], systems that provide enforced access control for user data and stream data in untrusted clouds. In contrast to these works, we are the first to evaluate predicate encryption and ABE based on modern modular approaches such as the predicate encoding and pair encoding frameworks. The resulting schemes also satisfy a stronger security notion (full vs. selective security) compared to the previously mentioned evaluations. We focus on synthetic case studies, while other works analyze more realistic settings and integration of ABE into bigger systems. Combining our methods with these more practical case studies is a very interesting direction for future work.

2 Background

In this section, we first introduce some mathematical notation and then define *predicate encodings*, *tag-based encodings* and *pair encodings* the three primitives used in the three different modular frameworks for predicate encryption.

2.1 Notation

For finite sets S, we use $x \xleftarrow{\$} S$ to denote that x is uniformly sampled from S. We define $[n]$ as the range $\{1, \ldots, n\}$ for an arbitrary $n \in \mathbb{N}$. For a predicate $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$, we use $(x, y) \in \mathsf{P}$ as a shorthand for $\mathsf{P}(x, y) = 1$. We use the same conventions for matrix-representations of linear maps on finite-dimensional spaces. We define vectors $\boldsymbol{v} \in \mathbb{F}^n$ as column matrices and denote the *transpose* of a matrix A by $A^\top$. We use $\mathsf{diag}(\boldsymbol{v})$ to denote the diagonal matrix with main diagonal $\boldsymbol{v}$. We denote the identity matrix of dimension n by I_n, a zero vector of length n by $\mathbf{0}_n$ and a zero matrix of m rows and n columns by $\mathbf{0}_{m,n}$. Let S be a set of indices and A be a matrix. A_S denotes the matrix formed from the set of columns of A with indices is in S. We define the *colspan* of a matrix $M \in \mathbb{F}^{m \times n}$ as the set of all possible linear combinations columns of M. This is $\underset{span}{col}\langle M \rangle = \{M\boldsymbol{v} : \boldsymbol{v} \in \mathbb{F}^n\} \subseteq \mathbb{F}^m$. We analogously define the *rowspan* of a

matrix. We consider prime order bilinear groups $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t)$ and use g_1, g_2, g_t to denote their respective generators. The map e satisfies $e(g_1^a, g_2^b) = g_t^{ab}$ for every $a, b \in \mathbb{N}$. A bilinear group is said to be symmetric if $\mathbb{G}_1 = \mathbb{G}_2$, otherwise it is called asymmetric. We abuse of notation and write $g^{\boldsymbol{v}}$ to denote $(g^{\boldsymbol{v}_1}, \ldots, g^{\boldsymbol{v}_n})$ for a group element g and a vector $\boldsymbol{v} \in \mathbb{Z}_p^n$.

2.2 Predicate Encodings

Predicate encodings are information-theoretic primitives that can be used for building predicate encryption schemes [33]. We adopt the definition from [12], but prefer to use matrix notation.

Definition 1 (Predicate encoding). *Let* $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ *be a binary predicate over finite sets* $\mathcal{X}$ *and* $\mathcal{Y}$*. Given a prime* $p \in \mathbb{N}$*, and* $s, r, w \in \mathbb{N}$*, a* (s, r, w)*-predicate encoding for* P *consists of five deterministic algorithms* $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$*: the* sender encoding algorithm sE *maps* $x \in \mathcal{X}$ *into a matrix* $\mathsf{sE}_x \in \mathbb{Z}_p^{s \times w}$*, the* receiver encoding algorithm rE *maps* $y \in \mathcal{Y}$ *into a matrix* $\mathsf{rE}_y \in \mathbb{Z}_p^{r \times w}$*, the* key encoding algorithm kE *maps* $y \in \mathcal{Y}$ *into a vector* $\mathsf{kE}_y \in \mathbb{Z}_p^r$*, while the* sender *and* receiver decoding algorithms*, respectively* sD *and* rD*, map a pair* $(x, y) \in \mathcal{X} \times \mathcal{Y}$ *into vectors* $\mathsf{sD}_{x,y} \in \mathbb{Z}_p^s$ *and* $\mathsf{rD}_{x,y} \in \mathbb{Z}_p^r$ *respectively. We require that the following properties are satisfied:*

reconstructability: *for all* $(x, y) \in \mathsf{P}$, $\mathsf{sD}_{x,y}^\top \mathsf{sE}_x = \mathsf{rD}_{x,y}^\top \mathsf{rE}_y$ *and* $\mathsf{rD}_{x,y}^\top \mathsf{kE}_y = 1$*;*

α-privacy: *for all* $(x, y) \notin \mathsf{P}, \alpha \in \mathbb{Z}_p$*,*

$$\boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w;\ \textit{return}\ (\mathsf{sE}_x \boldsymbol{w},\ \mathsf{rE}_y \boldsymbol{w} + \alpha \cdot \mathsf{kE}_y) \ \equiv\ \boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w;\ \textit{return}\ (\mathsf{sE}_x \boldsymbol{w},\ \mathsf{rE}_y \boldsymbol{w})$$

where $\equiv$ *denotes equality of distributions.*

Reconstructability allows to recover α from $(x, y, \mathsf{sE}_x \boldsymbol{w}, \mathsf{rE}_y \boldsymbol{w} + \alpha \cdot \mathsf{kE}_y)$ if $(x, y) \in \mathsf{P}$. *Privacy* ensures that α is perfectly hidden for such tuples if $(x, y) \notin \mathsf{P}$.

Example 1 (IBE predicate encoding). Let $\mathcal{X} = \mathcal{Y} = \mathbb{Z}_p$ and let $s = r = 1$, $w = 2$. We define the encoding functions as follows:

$$\mathsf{sE}_x = \begin{pmatrix} x & 1 \end{pmatrix} \qquad \mathsf{sD}_{x,y} = \begin{pmatrix} 1 \end{pmatrix}$$
$$\mathsf{rE}_y = \begin{pmatrix} y & 1 \end{pmatrix} \qquad \mathsf{rD}_{x,y} = \begin{pmatrix} 1 \end{pmatrix}$$
$$\mathsf{kE}_y = \begin{pmatrix} 1 \end{pmatrix}$$

The above is a predicate encoding for identity-based encryption, i.e., for the predicate $\mathsf{P}(x, y) = 1$ iff $x = y$. Note that $\begin{pmatrix} x & 1 \end{pmatrix} = \begin{pmatrix} y & 1 \end{pmatrix}$ when $x = y$, so reconstructability is satisfied. On the other hand, α-privacy follows from the fact that if $x \neq y$, $x \cdot w_1 + w_2$ and $y \cdot w_1 + w_2$ are pair-wise independent. ■

Predicate encryption from predicate encodings. We try to provide some intuition on how predicate encodings are compiled to predicate encryption schemes by the compiler from [12]. The master keys, ciphertexts and secret keys have the following form:

$$\mathsf{msk} = g_2^{\alpha} \qquad \mathsf{ct}_x = (g_1^s, g_1^{s \cdot \mathsf{sE}_x \boldsymbol{w}}, e(g_1, g_2)^{\alpha s} \cdot m)$$
$$\mathsf{mpk} = (g_1, g_1^{\boldsymbol{w}}, g_2, g_2^{\boldsymbol{w}}, e(g_1, g_2)^{\alpha}) \qquad \mathsf{sk}_y = (g_2^r, g_2^{\alpha \cdot \mathsf{kE}_y + r \cdot \mathsf{rE}_y \boldsymbol{w}})$$

The encrypted message $m \in \mathbb{G}_t$ is blinded by a random factor $e(g_1, g_2)^{\alpha s}$. The so-called *reconstruction* property of predicate encodings ensures that this blinding factor can be recovered for a pair $(\mathsf{ct}_x, \mathsf{sk}_y)$ if $\mathsf{P}(x, y) = 1$. More concretely, for all pairs (x, y) such that $\mathsf{P}(x, y) = 1$, because multiplying by matrices $\mathsf{sD}_{x,y}$,$\mathsf{rD}_{x,y}$ is a linear operation, it is possible operate in the exponent and compute

$$g_1^{s \cdot \mathsf{sD}_{x,y}^{\top} \mathsf{sE}_x \boldsymbol{w}} \quad \text{and} \quad g_2^{\mathsf{rD}_{x,y}^{\top} (\alpha \cdot \mathsf{kE}_y + r \cdot \mathsf{rE}_y \boldsymbol{w})},$$

obtaining $g_1^{s\beta}$ and $g_2^{\alpha + r\beta}$ for $\beta = \mathsf{sD}_{x,y}^{\top} \mathsf{sE}_x \boldsymbol{w} = \mathsf{rD}_{x,y}^{\top} \mathsf{rE}_y \boldsymbol{w}$ (note that knowing the value of β is nor necessary). Now, it is simple to recover $e(g_1, g_2)^{\alpha s}$ from $e(g_1^s, g_2^{\alpha + r\beta})$ and $e(g_1^{s\beta}, g_2^r)$. Security is ensured by the *α-privacy* property of the encoding together with decisional assumptions about dual system groups. Intuitively, the α-privacy property states that given certain values derived from the output of the encoding functions for random input, α remains information-theoretic hidden.

2.3 Tag-based encodings

Tag-based encodings is a new primitive defined in a very recent work [19] that defines a new generic framework (using prime order groups) for modular design of predicate encryption.

Definition 2 (Tag-based encoding). *Let $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ be a binary predicate over finite sets $\mathcal{X}$ and $\mathcal{Y}$. Given a prime $p \in \mathbb{N}$, and $c, k, h \in \mathbb{N}$, a (c, k, h)-tag-based encoding encoding for P consists of two deterministic algorithms $(\mathsf{cE}, \mathsf{kE})$: the* ciphertext encoding algorithm cE *maps $x \in \mathcal{X}$ into a matrix $\mathsf{cE}_x \in \mathbb{Z}_p^{c \times h}$ and the* key encoding algorithm kE *maps $y \in \mathcal{Y}$ into a matrix $\mathsf{kE}_y \in \mathbb{Z}_p^{k \times h}$. We require that the following properties are satisfied:*

reconstructability: *for all $(x, y) \in \mathsf{P}$, there exists an efficient algorithm that on input (x, y) computes vectors $\boldsymbol{m}_c \in \mathbb{Z}_p^c$, $\boldsymbol{m}_k \in \mathbb{Z}_p^k$ such that*

$$\boldsymbol{m}_c^{\top} \mathsf{cE}_x = \boldsymbol{m}_k^{\top} \mathsf{kE}_y \neq \boldsymbol{0}_h^{\top}$$

***h*-hiding:** *for all $(x, y) \notin \mathsf{P}$,*

$$\boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^h;\ \textit{return}\ (\mathsf{cE}_x \boldsymbol{h},\ \mathsf{kE}_y \boldsymbol{h}) \quad \approx_s \quad \boldsymbol{h}, \boldsymbol{h}' \xleftarrow{\$} \mathbb{Z}_p^h;\ \textit{return}\ (\mathsf{cE}_x \boldsymbol{h},\ \mathsf{kE}_y \boldsymbol{h}')$$

where $\approx_s$ denotes negligible statistical distance between distributions.

The compiler proposed in [19] uses similar ideas to the one for predicate encodings. However, it does not rely on dual system groups and can be instantiated with symmetric bilinear maps. The message is blinded and ciphertexts and keys contain a set of group elements that are enough to recover the blinding factor only when the predicate is true. This compiler has the advantage that some elements of ciphertexts and keys are $\mathbb{Z}_p$ values and not group elements, which reduces the storage size.

2.4 Pair Encodings

Attrapadung [4, 5] proposes an independent modular framework for predicate encryption, based on a primitive called *pair encoding*. For our purposes, it suffices to consider a more restrictive, information-theoretic, notion of pair encodings.

Definition 3 (Information-theoretic pair encoding). *Let* $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ *be a binary predicate over finite sets* $\mathcal{X}$ *and* $\mathcal{Y}$*. Given a prime* $p \in \mathbb{N}$*, and* $c, k, l, m, n \in \mathbb{N}$*, let* $\boldsymbol{h} = (h_1, \ldots, h_n)$*,* $\boldsymbol{s} = (s_0, s_1, \ldots, s_l)$ *and* $\boldsymbol{r} = (\alpha, r_1, \ldots, r_m)$ *be sets of variables. An information-theoretic* (c, k, n)*-pair encoding scheme for* P *consists of three deterministic algorithms* $(\mathsf{Enc1}, \mathsf{Enc2}, \mathsf{Pair})$*: the* ciphertext encoding algorithm $\mathsf{Enc1}$ *maps a value* $x \in \mathcal{X}$ *into a list of polynomials* $\boldsymbol{c}_x \in \mathbb{Z}_p[\boldsymbol{s}, \boldsymbol{h}]^c$*, the* key encoding algorithm $\mathsf{Enc2}$ *maps a value* $y \in \mathcal{Y}$ *into a list of polynomials* $\boldsymbol{k}_y \in \mathbb{Z}_p[\boldsymbol{r}, \boldsymbol{h}]^k$ *and the* decoding algorithm Pair *maps a pair* $(x, y) \in \mathcal{X} \times \mathcal{Y}$ *into a matrix* $E_{x,y} \in \mathbb{Z}_p^{c \times k}$*. We require that the following properties are satisfied:*

polynomial constraints:

- *For every* $x \in \mathcal{X}$ *and every* $f \in \mathsf{Enc1}(x)$*,* $f = f(\boldsymbol{s}, \boldsymbol{h})$ *only contains monomials of the form* s_i *or* $s_i h_j$*,* $i \in [0, l]$*,* $j \in [n]$*.*
- *For every* $y \in \mathcal{Y}$ *and every* $f \in \mathsf{Enc2}(y)$*,* $f = f(\boldsymbol{r}, \boldsymbol{h})$ *only contains monomials of the form* α*,* r_i *or* $r_i h_j$*,* $i \in [m]$*,* $j \in [n]$*.*

reconstructability: *for all* $(x, y) \in \mathsf{P}$ *and all* $\boldsymbol{c}_x \leftarrow \mathsf{Enc1}(x)$*,* $\boldsymbol{k}_y \leftarrow \mathsf{Enc2}(y)$*,* $E_{x,y} \leftarrow \mathsf{Pair}(x, y)$*, the following polynomial equality holds* $\boldsymbol{c}_x^\top E_{x,y} \boldsymbol{k}_y = \alpha s_0$*.*

perfect security: *for all* $(x, y) \notin \mathsf{P}$ *and all* $\boldsymbol{c}_x \leftarrow \mathsf{Enc1}(x)$*,* $\boldsymbol{k}_y \leftarrow \mathsf{Enc2}(y)$*,*

$$\boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^n;\ \boldsymbol{r} \xleftarrow{\$} (\mathbb{Z}_p^*)^m;\ \boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_p^{l+1}; \qquad \mathsf{return}\ (\boldsymbol{c}_x(\boldsymbol{s}, \boldsymbol{h}),\ \boldsymbol{k}_y(0, \boldsymbol{r}, \boldsymbol{h})) \quad \equiv$$

$$\boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^n;\ \boldsymbol{r} \xleftarrow{\$} (\mathbb{Z}_p^*)^m;\ \boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_p^{l+1}; \alpha \xleftarrow{\$} \mathbb{Z}_p; \quad \mathsf{return}\ (\boldsymbol{c}_x(\boldsymbol{s}, \boldsymbol{h}),\ \boldsymbol{k}_y(\alpha, \boldsymbol{r}, \boldsymbol{h}))$$

where $\equiv$ *denotes equality of distributions.*

The compiler from pair encodings follows similar ideas to the other compilers. The message is blinded by a random factor and ciphertexts and keys contain all the information necessary to recover this blinded factor, only when the predicate

holds. The compiler from pair encodings requires to compute a polynomial number of pairings during decryption, unlike the compilers for predicate encodings and tag-based encodings that need[3] 6 and 8 pairings respectively.

3 Predicate encodings: properties and consequences

In this section, we present a purely algebraic (and independent of α) characterization of the α-privacy property. It simplifies both the analysis and the construction of predicate encodings. In particular, we use our characterization to define and prove a new optimization of predicate encodings, i.e., a transformation that makes the encoding functions smaller while preserving the predicate. Additionally, we unify the reconstructability and privacy properties and show that they are mutually exclusive and complementary, i.e., for every $(x, y) \in \mathcal{X} \times \mathcal{Y}$, one and only one of the two conditions holds. This unified treatment facilitates the construction and study of predicate encodings.

3.1 Algebraic properties of predicate encodings

The following theorem captures two essential properties of predicate encodings: first, privacy admits a purely algebraic characterization (furthermore independent of α) given in terms of existence of solutions of a linear system of equations. Second, reconstructability and privacy, when viewed as properties of a single pair (x, y), negate each other; i.e. a pair (x, y) always satisfies exactly one of the two properties.

Theorem 1 (Algebraic characterization of privacy). *Let $p \in \mathbb{N}$ be a prime, let $s, r, w \in \mathbb{N}$ and let $S \in \mathbb{Z}_p^{s\times w}$, $R \in \mathbb{Z}_p^{r\times w}$, $\boldsymbol{k} \in \mathbb{Z}_p^r$. The following are equivalent:*

- ***α-privacy*** *For every $\alpha \in \mathbb{Z}_p$,*

$$\boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (S\boldsymbol{w},\, R\boldsymbol{w} + \alpha \cdot \boldsymbol{k}) \quad \equiv \quad \boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (S\boldsymbol{w},\, R\boldsymbol{w})$$

- ***(algebraic) privacy*** *There exists $\boldsymbol{w} \in \mathbb{Z}_p^w$ such that $S\boldsymbol{w} = \boldsymbol{0}_s$ and $R\boldsymbol{w} = \boldsymbol{k}$*
- ***non-reconstructability*** *For every $\boldsymbol{s} \in \mathbb{Z}_p^s$ and $\boldsymbol{r} \in \mathbb{Z}_p^r$, either $\boldsymbol{s}^\top S \neq \boldsymbol{r}^\top R$ or $\boldsymbol{r}^\top \boldsymbol{k} \neq 1$.*

Proof. We first prove that α-privacy is equivalent to algebraic privacy. Note that the fact that $\forall\, \alpha \in \mathbb{Z}_p$,

$$\boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (S\boldsymbol{w},\, R\boldsymbol{w} + \alpha \cdot \boldsymbol{k}) \quad \equiv \quad \boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (S\boldsymbol{w},\, R\boldsymbol{w})$$

is equivalent to the existence of a bijection ρ_α such that for all $\boldsymbol{w} \in \mathbb{Z}_p^w$, $S\boldsymbol{w} = S \cdot \rho_\alpha(\boldsymbol{w}) \ \wedge\ R\boldsymbol{w} + \alpha \cdot \boldsymbol{k} = R \cdot \rho_\alpha(\boldsymbol{w})$. By linearity, it can be rewritten as

$$S(\rho_\alpha(\boldsymbol{w}) - \boldsymbol{w}) = \boldsymbol{0}_s \quad \wedge \quad \alpha \cdot \boldsymbol{k} = R(\rho_\alpha(\boldsymbol{w}) - \boldsymbol{w})$$

[3] decryption in the framework of predicate encodings needs 4 pairings under SXDH assumption or 6 under DLIN, in the framework of tag-based encodings decryption requires 8 pairings and the assumption is DLIN

Now, the existence of such a bijection is equivalent to the existence of a solution for the following (parametric in α) linear system on $\boldsymbol{w}$: $S\boldsymbol{w} = \mathbf{0}_s \ \wedge \ R\boldsymbol{w} = \alpha \cdot \boldsymbol{k}$. To see this, note that if ρ_α is such a bijection, $\rho_\alpha(\boldsymbol{w}_0) - \boldsymbol{w}_0$ is a solution of the system for every $\boldsymbol{w}_0 \in \mathbb{Z}_p^w$. On the other hand, if $\boldsymbol{w}^*$ is a solution of the system, the bijection $\rho_\alpha(\boldsymbol{w}) = \boldsymbol{w} + \boldsymbol{w}^*$ satisfies the required identities. To conclude, note that the above system has a solution iff the following (independent of α) does:

$$S\boldsymbol{w} = \mathbf{0}_s \quad \wedge \quad R\boldsymbol{w} = \boldsymbol{k}$$

Next, we prove the equivalence between algebraic privacy and non-reconstructability. We use the following helping lemma from [10, Claim 2]: for every field $\mathbb{F}$, let $A \in \mathbb{F}^{m \times n}$ and $\boldsymbol{b} \in \mathbb{F}^n$ be matrices with coefficients in $\mathbb{F}$, the following two statements are equivalent:

- for every $\boldsymbol{a} \in \mathbb{F}^m$, $\boldsymbol{b}^\top \neq \boldsymbol{a}^\top A$;
- there exists $\boldsymbol{z} \in \mathbb{F}^n$ such that $\boldsymbol{z}^\top \boldsymbol{b} = 1$ and $A\boldsymbol{z} = \mathbf{0}_m$.

Assume that algebraic privacy does not hold, i.e., for every $\boldsymbol{w} \in \mathbb{Z}_p^w$, either $S\boldsymbol{w} \neq \mathbf{0}_s$ or $R\boldsymbol{w} \neq \boldsymbol{k}$. Equivalently, for every $\boldsymbol{w} \in \mathbb{Z}_p^w$

$$\begin{pmatrix} \mathbf{0}_s \\ \boldsymbol{k} \end{pmatrix} \neq \begin{pmatrix} -S \\ R \end{pmatrix} \boldsymbol{w}$$

which is equivalent (by our helping lemma) to the existence of $(\boldsymbol{z}_1, \boldsymbol{z}_2) \in \mathbb{Z}_p^s \times \mathbb{Z}_p^r$ such that

$$\begin{pmatrix} \boldsymbol{z}_1^\top & \boldsymbol{z}_2^\top \end{pmatrix} \begin{pmatrix} \mathbf{0}_s \\ \boldsymbol{k} \end{pmatrix} = 1 \quad \wedge \quad \begin{pmatrix} \boldsymbol{z}_1^\top & \boldsymbol{z}_2^\top \end{pmatrix} \begin{pmatrix} -S \\ R \end{pmatrix} = \mathbf{0}_w^\top$$

This is, there exists $\boldsymbol{z}_1 \in \mathbb{Z}_p^s$, $\boldsymbol{z}_2 \in \mathbb{Z}_p^r$ such that $\boldsymbol{z}_1^\top S = \boldsymbol{z}_2^\top R \ \wedge \ \boldsymbol{z}_2^\top \boldsymbol{k} = 1$, which is exactly reconstructability. The proof follows from the fact all the steps are equivalences. □

Our next result is a representation theorem. It is based on the notion of partial encoding; informally, a partial encoding consists of the first three algorithms of a predicate encoding; it is not attached to any specific predicate, nor is required to satisfy any property.

Definition 4 (Partial encoding). *Let $\mathcal{X}$ and $\mathcal{Y}$ be finite sets. Let $p \in \mathbb{N}$ be a prime and $s, r, w \in \mathbb{N}$. A (s, r, w)-partial encoding is given by three deterministic algorithms $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE})$: sE maps $x \in \mathcal{X}$ into a matrix $\mathsf{sE}_x \in \mathbb{Z}_p^{s \times w}$, and rE, kE map $y \in \mathcal{Y}$ into a matrix $\mathsf{rE}_y \in \mathbb{Z}_p^{r \times w}$ and a vector $\mathsf{kE}_y \in \mathbb{Z}_p^r$ respectively.*

The representation theorem shows that there exists an embedding from partial encodings to predicate encodings, and that every predicate encoding lies the image of the embedding.

Theorem 2 (Representation theorem). *Let $\mathcal{X}$ and $\mathcal{Y}$ be finite sets. Let $p \in \mathbb{N}$ be a prime and $s, r, w \in \mathbb{N}$. Every (s, r, w)-partial encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE})$ for $\mathcal{X}$ and $\mathcal{Y}$ induces a predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for the following predicate (henceforth coined implicit predicate):*

$$\mathsf{Pred}(x, y) \triangleq \ \forall \boldsymbol{w} \in \mathbb{Z}_p^w, \quad \mathsf{sE}_x \boldsymbol{w} \neq \mathbf{0}_s \ \vee \ \mathsf{rE}_y \boldsymbol{w} \neq \mathsf{kE}_y$$

Moreover, if $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ *is a predicate encoding for* P*, then for every* $(x, y) \in \mathcal{X} \times \mathcal{Y}$*,* $\mathsf{P}(x, y) \Leftrightarrow \mathsf{Pred}(x, y)$*.*

Example 2 (Implicit predicate of IBE predicate encoding). If we consider the following partial encoding functions corresponding to the encoding presented in Example 1:

$$\mathsf{sE}_x = \begin{pmatrix} x & 1 \end{pmatrix} \qquad \mathsf{rE}_y = \begin{pmatrix} y & 1 \end{pmatrix} \qquad \mathsf{kE}_y = \begin{pmatrix} 1 \end{pmatrix}$$

our Theorem 2 guarantees that it is a valid predicate encoding for the implicit predicate:

$$\mathsf{Pred}(x, y) = 1 \text{ iff } \forall (w_1, w_2) \in \mathbb{Z}_p^2,\ x \cdot w_1 + w_2 \neq 0 \ \vee \ y \cdot w_1 + w_2 \neq 1$$

A simple analysis shows that the above predicate is equivalent to $x = y$. ■

A consequence of Theorem 2 is that a predicate P over $\mathcal{X}$ and $\mathcal{Y}$ can be instantiated by a (s, r, w)-predicate encoding iff there exist $\mathcal{X}$-indexed and $\mathcal{Y}$-indexed matrices $S_x \in \mathbb{Z}_p^{s \times w}$ and $R_y \in \mathbb{Z}_p^{r \times w}$ and $\mathcal{Y}$-indexed vectors $\boldsymbol{k}_y \in \mathbb{Z}_p^r$ such that:

$$\mathsf{P}(x, y) = 1 \text{ iff } \begin{pmatrix} \mathbf{0}_s \\ \boldsymbol{k}_y \end{pmatrix} \notin \underset{span}{col} \left\langle \begin{matrix} S_x \\ R_y \end{matrix} \right\rangle$$

This is helpful to analyze the expressivity of predicate encodings of certain size.

Example 3. Let $\mathcal{X}$ and $\mathcal{Y}$ be finite sets, let $n \in \mathbb{N}$, we will characterize all the predicates that can be achieved from a $(1, 1, n)$-partial encoding, say $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE})$. Note that for every pair (x, y), sE_x and rE_y are vectors of length n, while kE_y is a single element. Say,

$$\mathsf{sE}_x = (f_1(x), \ldots, f_n(x)) \qquad \mathsf{rE}_y = (g_1(y), \ldots, g_n(y)) \qquad \mathsf{kE}_y = h(y)$$

for certain functions $f_i : \mathcal{X} \to \mathbb{Z}_p$, $g_i, h : \mathcal{Y} \to \mathbb{Z}_p$ for every $i \in [n]$. Theorem 2 guarantees that the above is a valid predicate encoding for the predicate

$$\mathsf{P}(x, y) = 1 \text{ iff } h(y) \neq 0 \ \wedge \ \Big(\exists \beta \in \mathbb{Z}_p : \bigwedge_{i \in [n]} f_i(x) = \beta g_i(y) \Big)$$

It can be shown that the predicate $\mathsf{P}((x_1, x_2), y) = 1$ iff $(x_1 = y) \vee (x_2 = y)$ cannot be captured by $(1, 1, n)$-predicate encodings, while on the contrary, the predicate $\mathsf{P}((x_1, x_2), y) = 1$ iff $(x_1 = y) \wedge (x_2 = y)$ could be instantiated.

3.2 Optimizing predicate encodings

In this section, we show that the efficiency of predicate encodings can be improved by pre- and post-processing. Specifically, we show that an (s, r, w)-encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for a predicate P can be transformed into a (s', r', w')-encoding $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}', \mathsf{sD}', \mathsf{rD}')$ for the same predicate, by applying a linear transformation to the matrices induced by $\mathsf{sE}, \mathsf{rE}, \mathsf{kE}$.

More precisely, if we define $\mathsf{sE}'_x = A\mathsf{sE}_x$, $\mathsf{rE}'_y = B\mathsf{rE}_y$ and $\mathsf{kE}'_y = B\mathsf{kE}_y$ for two matrices A and B, the privacy of the encoding will be preserved, but reconstructability may be destroyed. On the contrary, when we consider the partial encoding $\mathsf{sE}'_x = \mathsf{sE}_x C$, $\mathsf{rE}'_y = \mathsf{rE}_y C$ and $\mathsf{kE}'_y = \mathsf{kE}_y$ for a matrix C, reconstructability is automatically guaranteed, but privacy could not hold (for the same predicate). Intuitively, this occurs because *reconstructability* depends on the *rowspan* of the matrices $\mathsf{sE}_x, \mathsf{rE}_y$, while *privacy* depends on their *colspan*. Our following theorem imposes conditions on these matrices A, B and C so that the resulting predicate encoding is equivalent to the original one.

Theorem 3. *Let $\mathcal{X}$ and $\mathcal{Y}$ be finite sets. Let $p \in \mathbb{N}$ be a prime, $s, r, w, s', r', w' \in \mathbb{N}$, and let $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ be a (s, r, w)-predicate encoding for $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$. Let A be a function that maps every element $x \in \mathcal{X}$ into a matrix $\mathsf{A}_x \in \mathbb{Z}_p^{s' \times s}$, B be a function that maps $y \in \mathcal{Y}$ into a matrix $\mathsf{B}_y \in \mathbb{Z}_p^{r' \times r}$ and let $\mathsf{C} \in \mathbb{Z}_p^{w \times w'}$ be a matrix. There exists a (s', r', w')-partial encoding $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}', \mathsf{sD}', \mathsf{rD}')$ for P, where*

$$\mathsf{sE}'_x = \mathsf{A}_x \mathsf{sE}_x \mathsf{C} \qquad \mathsf{rE}'_y = \mathsf{B}_y \mathsf{rE}_y \mathsf{C} \qquad \mathsf{kE}'_y = \mathsf{B}_y \mathsf{kE}_y$$

provided the following conditions hold:

- *For all $(x, y) \in \mathsf{P}$, $\mathsf{sD}_{x,y}^\top \in \underset{span}{row} \langle \mathsf{A}_x \rangle$ and $\mathsf{rD}_{x,y}^\top \in \underset{span}{row} \langle \mathsf{B}_y \rangle$;*
- *For all $(x, y) \notin \mathsf{P}$, there exists $\boldsymbol{w} \in \underset{span}{col} \langle \mathsf{C} \rangle$ s.t. $\mathsf{sE}_x \boldsymbol{w} = \mathbf{0}_s$ and $\mathsf{rE}_y \boldsymbol{w} = \mathsf{kE}_y$.*

This transformation is useful to make predicate encodings simpler and more efficient in different manners. For instance, it can be used to make the matrices corresponding to encoding and decoding functions become sparser. That is, if we consider A and B as functions that apply matrix *Gaussian elimination*[4] to the matrices associated to sE and rE, kE, many entries from these matrices will be zero. Hence, fewer group operations will be performed during encryption and key generation, improving the performance. Moreover, the transformation can be used to reduce the size of mpk, ct_x and sk_y. If $w' < w$, the number of elements in mpk will decrease. This will also improve the performance of encryption and key generation (both depend directly on mpk). Additionally, if $s' < s$ or $r' < r$, the number of elements in ct_x and sk_y will also decrease respectively.

Note that a simplification from the right (multiplying by C) changes the structure of the encoding and may open the possibility of left-simplifications that were not available before and vice versa. Example 4 illustrates this idea. We optimize a predicate encoding that corresponds to the result of applying our negation transformation (from next section, Theorem 6) to the predicate encoding from Example 1.

[4] note that if matrices A_x, B_y or C are invertible, they always satisfy their respective requirements

Example 4. Let $\mathcal{X} = \mathcal{Y} = \mathbb{Z}_p$ and consider the $(2,3,4)$-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for $\mathsf{P}(x,y) = 1$ iff $x \neq y$, defined as

$$\mathsf{sE}_x = \begin{pmatrix} x & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \end{pmatrix} \qquad \mathsf{rE}_y = \begin{pmatrix} 0 & 1 & 0 & y \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathsf{rE}_y = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$\mathsf{sD}_{x,y}^\top = \begin{pmatrix} \frac{-1}{x-y} & \frac{x}{x-y} \end{pmatrix} \qquad \mathsf{rD}_{x,y}^\top = \begin{pmatrix} \frac{1}{x-y} & \frac{-x}{x-y} & 1 \end{pmatrix}$$

Note that for every pair $(x,y) \notin \mathsf{P}$, i.e. $x = y$, the single solution of the system $\mathsf{sE}_x \boldsymbol{w} = \mathbf{0}_2 \wedge \mathsf{rE}_y \boldsymbol{w} = \mathsf{kE}_y$ is $\boldsymbol{w}^\top = \begin{pmatrix} -1 & -y & -1 & 1 \end{pmatrix}$, thus the matrix

$$\mathsf{C} = \begin{pmatrix} -1 & 0 & -1 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}^\top$$

satisfies the conditions of Theorem 3. Therefore, the $(2,3,2)$-partial encoding $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}')$, where

$$\mathsf{sE}'_x = \mathsf{sE}_x \mathsf{C} = \begin{pmatrix} -x & -1 \\ 0 & 0 \end{pmatrix} \quad \mathsf{rE}'_y = \mathsf{rE}_y \mathsf{C} = \begin{pmatrix} y & 1 \\ 0 & 0 \\ 1 & 0 \end{pmatrix} \quad \mathsf{kE}'_y = \mathsf{kE}_y = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

induces a predicate encoding for the same predicate. The previous simplification, opens the possibility of applying again the theorem, with matrices A_x and B_y, obtaining a $(1,2,2)$-predicate encoding for $\mathsf{P}(x,y) = 1$ iff $x \neq y$. Concretely,

$$\mathsf{A}_x = \begin{pmatrix} -1 & 0 \end{pmatrix} \qquad \mathsf{sE}''_x = \begin{pmatrix} x & 1 \end{pmatrix} \qquad \mathsf{rE}''_y = \begin{pmatrix} y & 1 \\ 1 & 0 \end{pmatrix} \qquad \mathsf{rE}''_y = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\mathsf{B}_y = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathsf{sD}''^\top_{x,y} = \begin{pmatrix} \frac{1}{x-y} \end{pmatrix} \quad \mathsf{rD}''^\top_{x,y} = \begin{pmatrix} \frac{1}{x-y} & 1 \end{pmatrix}$$

■

The above simplifications can be successfully applied to actual predicate encodings proposed in [12]. In Section 6.2 we propose improved predicate encodings for *monotonic boolean formulas* and *arithmetic span programs*.

3.3 Combining predicates

Using the new characterization of predicate encodings from the previous section, we define transformations to combine predicate encodings into new predicate encodings for more complex predicates. In particular, we define predicate encoding transformations for disjunction, conjunction, negation and the dual predicate. These combinations are useful to create new schemes that inherit different properties from the more basic building blocks. In Section 6, we propose several constructions that rely on these transformations.

Disjunction. We present a method to build a predicate encoding for the disjunction of P_1 and P_2 from predicate encodings for P_1 and P_2. Observe that the predicate encryption scheme obtained from the resulting predicate encoding is more efficient than the predicate encryption scheme obtained by compiling the predicate encodings of P_1 and P_2 separately, and then applying a generic transformation that builds predicate encryption schemes for a disjunction from predicate encryption schemes of its disjuncts.

Theorem 4 (Disjunction of predicate encodings). *For every (s_1, r_1, w_1)-predicate encoding $(\mathsf{sE}^1, \mathsf{rE}^1, \mathsf{kE}^1, \mathsf{sD}^1, \mathsf{rD}^1)$ for $\mathsf{P}_1 : \mathcal{X}_1 \times \mathcal{Y}_1 \to \{0,1\}$ and every (s_2, r_2, w_2)-predicate encoding $(\mathsf{sE}^2, \mathsf{rE}^2, \mathsf{kE}^2, \mathsf{sD}^2, \mathsf{rD}^2)$ for $\mathsf{P}_2 : \mathcal{X}_2 \times \mathcal{Y}_2 \to \{0,1\}$, there exists a $(s_1+s_2, r_1+r_2, w_1+w_2)$-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for the predicate $\mathsf{P} : (\mathcal{X}_1, \mathcal{X}_2) \times (\mathcal{Y}_1, \mathcal{Y}_2) \to \{0,1\}$ such that:*

$$\mathsf{P}((x_1, x_2), (y_1, y_2)) \Leftrightarrow \mathsf{P}_1(x_1, y_1) \vee \mathsf{P}_2(x_2, y_2)$$

Concretely,

$$\mathsf{sE}_{(x_1,x_2)} = \begin{pmatrix} \mathsf{sE}^1_{x_1} & \mathbf{0}_{s_1,w_2} \\ \mathbf{0}_{s_2,w_1} & \mathsf{sE}^2_{x_2} \end{pmatrix} \quad \mathsf{rE}_{(y_1,y_2)} = \begin{pmatrix} \mathsf{rE}^1_{y_1} & \mathbf{0}_{r_1,w_2} \\ \mathbf{0}_{r_2,w_1} & \mathsf{rE}^2_{y_2} \end{pmatrix} \quad \mathsf{kE}_{(y_1,y_2)} = \begin{pmatrix} \mathsf{kE}^1_{y_1} \\ \mathsf{kE}^2_{y_2} \end{pmatrix}$$

$$\begin{aligned} \mathsf{sD}^\top_{(x_1,x_2),(y_1,y_2)} &= \textit{ if } \mathsf{P}_1(x_1,y_1) \textit{ then } \begin{pmatrix} \mathsf{sD}^{1\top}_{x_1,y_1} & \mathbf{0}^\top_{s_2} \end{pmatrix} \textit{ else } \begin{pmatrix} \mathbf{0}^\top_{s_1} & \mathsf{sD}^{2\top}_{x_2,y_2} \end{pmatrix} \\ \mathsf{rD}^\top_{(x_1,x_2),(y_1,y_2)} &= \textit{ if } \mathsf{P}_1(x_1,y_1) \textit{ then } \begin{pmatrix} \mathsf{rD}^{1\top}_{x_1,y_1} & \mathbf{0}^\top_{r_2} \end{pmatrix} \textit{ else } \begin{pmatrix} \mathbf{0}^\top_{r_1} & \mathsf{rD}^{2\top}_{x_2,y_2} \end{pmatrix} \end{aligned}$$

Note that it is possible to obtain sharing between attributes, e.g., if $\mathcal{X}_1 = \mathcal{X}_2$ and the sender uses only the subset $\{(x,x) \mid x \in \mathcal{X}_1\}$ of $\mathcal{X}_1 \times \mathcal{X}_2$, the predicate becomes $\mathsf{P}(x, (y_1, y_2)) = 1$ iff $\mathsf{P}_1(x, y_1) \vee \mathsf{P}_2(x, y_2)$.

Conjunction. In contrast to disjunction, the naive solution that just concatenates secret keys fails. Given keys for attribute pairs (y_1, y_2) and (y_1', y_2'), it would be possible to recombine the components and obtain a key for (y_1, y_2') leading to collusion attacks. Our predicate encoding transformation deals with this problem by "tying" the two components together with additional randomness.

Theorem 5 (Conjunction of predicate encodings). *For every (s_1, r_1, w_1)-predicate encoding $(\mathsf{sE}^1, \mathsf{rE}^1, \mathsf{kE}^1, \mathsf{sD}^1, \mathsf{rD}^1)$ for $\mathsf{P}_1 : \mathcal{X}_1 \times \mathcal{Y}_1 \to \{0,1\}$ and every (s_2, r_2, w_2)-predicate encoding $(\mathsf{sE}^2, \mathsf{rE}^2, \mathsf{kE}^2, \mathsf{sD}^2, \mathsf{rD}^2)$ for $\mathsf{P}_2 : \mathcal{X}_2 \times \mathcal{Y}_2 \to \{0,1\}$, there exists a $(s_1+s_2,\ r_1+r_2,\ w_1+w_2+1)$-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for the predicate $\mathsf{P} : (\mathcal{X}_1, \mathcal{X}_2) \times (\mathcal{Y}_1, \mathcal{Y}_2) \to \{0,1\}$ such that:*

$$\mathsf{P}((x_1, x_2), (y_1, y_2)) \Leftrightarrow \mathsf{P}_1(x_1, y_1) \wedge \mathsf{P}_2(x_2, y_2)$$

Concretely,

$$\mathsf{sE}_{(x_1,x_2)} = \begin{pmatrix} \mathsf{sE}^1_{x_1} & \mathbf{0}_{s_1,w_2} & \mathbf{0}_{s_1} \\ \mathbf{0}_{s_2,w_1} & \mathsf{sE}^2_{x_2} & \mathbf{0}_{s_2} \end{pmatrix} \qquad \mathsf{sD}_{(x_1,x_2),(y_1,y_2)} = \frac{1}{2}\begin{pmatrix} \mathsf{sD}^1_{x_1,y_1} \\ \mathsf{sD}^2_{x_2,y_2} \end{pmatrix}$$

$$\mathsf{rE}_{(y_1,y_2)} = \begin{pmatrix} \mathsf{rE}^1_{y_1} & \mathbf{0}_{r_1,w_2} & \mathsf{kE}^1_{y_1} \\ \mathbf{0}_{r_2,w_1} & \mathsf{rE}^2_{y_2} & \text{-}\mathsf{kE}^2_{y_2} \end{pmatrix} \qquad \mathsf{rD}_{(x_1,x_2),(y_1,y_2)} = \frac{1}{2}\begin{pmatrix} \mathsf{rD}^1_{x_1,y_1} \\ \mathsf{rD}^2_{x_2,y_2} \end{pmatrix}$$

$$\mathsf{kE}_{(y_1,y_2)} = \begin{pmatrix} \mathsf{kE}^1_{y_1} \\ \mathsf{kE}^2_{y_2} \end{pmatrix}$$

Note that it is possible to combine Theorems 4 and 5 to create a predicate encoding for $\mathsf{P}_1 \bowtie \mathsf{P}_2$, where the placeholder $\bowtie \in \{\vee, \wedge\}$ can be part of keys or ciphertexts. See Appendix B for more details about this encoding.

Negation. To obtain a functionally complete set of boolean predicate encoding transformers, we now define a transformation for negation. This unifies negated predicates like Non-zero Inner Product Encryption (NIPE) and Zero Inner Product Encryption (ZIPE). In Section 6.2 we use this transformation to build optimized predicate encodings. The technique works for predicate encodings where the negation transformation yields a predicate encoding that can be further simplified (using our method from Section 3.2).

Theorem 6 (Negation of predicate encodings). *For every (s, r, w)-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ there exists a $(w, w+1, s+w+r)$-predicate encoding $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}', \mathsf{sD}', \mathsf{rD}')$ for the predicate $\mathsf{P}' : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ such that $\mathsf{P}'(x, y) \Leftrightarrow \neg\mathsf{P}(x, y)$. Concretely,*

$$\mathsf{sE}'_x = \begin{pmatrix} \mathsf{sE}_x^\top & \text{-}I_w & \mathbf{0}_{w,r} \end{pmatrix} \quad \mathsf{rE}'_y = \begin{pmatrix} \mathbf{0}_{w,s} & I_w & \mathsf{rE}_y^\top \\ \mathbf{0}_s^\top & \mathbf{0}_w^\top & \mathsf{kE}_y^\top \end{pmatrix} \quad \mathsf{kE}'_y = \begin{pmatrix} \mathbf{0}_w \\ 1 \end{pmatrix}$$

$$\mathsf{sD}'_{x,y} = \boldsymbol{w}_{x,y} \qquad \mathsf{rD}'_{x,y} = \begin{pmatrix} \text{-}\boldsymbol{w}_{x,y} \\ 1 \end{pmatrix}$$

where for a pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$ such that $\mathsf{P}(x, y) = 0$, $\boldsymbol{w}_{x,y}$ is defined as the witness for algebraic privacy, i.e., a vector such that

$$\mathsf{sE}_x \boldsymbol{w}_{x,y} = \mathbf{0}_s \quad \wedge \quad \mathsf{rE}_y \boldsymbol{w}_{x,y} = \mathsf{kE}_y$$

Note that such a vector always exists when $\mathsf{P}(x, y) = 0$. Moreover, sD and rD do not need to be defined when $\mathsf{P}'(x, y)$ is not 1, this is, when $\mathsf{P}(x, y)$ is not 0.

Dual. In the literature, the notions of KP-ABE and CP-ABE are considered separately. In fact, many works are only valid for one of the two versions of Attribute Based Encryption. Our transformation unifies the notion of KP-ABE and CP-ABE in the framework of predicate encodings. In this context they should not be considered separately, because our transformation provides a Ciphertext-Policy predicate encoding from any Key-Policy predicate encoding and vice versa.

Theorem 7 (Dual predicate encoding). *For every (s, r, w)-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ there exists a $(r, s+1, w+1)$-predicate encoding $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}', \mathsf{sD}', \mathsf{rD}')$ for the predicate $\mathsf{P}' : \mathcal{Y} \times \mathcal{X} \to \{0,1\}$ such that $\mathsf{P}'(y,x) \Leftrightarrow \mathsf{P}(x,y)$. Concretely,*

$$\mathsf{sE}'_y = \begin{pmatrix} \mathsf{rE}_y & \mathsf{kE}_y \end{pmatrix} \qquad \mathsf{rE}'_x = \begin{pmatrix} \mathsf{sE}_x & \mathbf{0}_s \\ \mathbf{0}_w^\top & 1 \end{pmatrix} \qquad \mathsf{kE}'_x = \begin{pmatrix} \mathbf{0}_s \\ 1 \end{pmatrix}$$

$$\mathsf{sD}'_{y,x} = \mathsf{rD}_{x,y} \qquad \mathsf{rD}'_{y,x} = \begin{pmatrix} \mathsf{sD}_{x,y} \\ 1 \end{pmatrix}$$

4 Tag-based Encodings

We show that our techniques for *predicate encodings* can be extended to the framework of *tag-based encodings*. In particular, we show a similar result to our Theorem 1, which establishes that $\boldsymbol{h}$-hiding and reconstructability are mutually exclusive and complementary.

Theorem 8. *Let $p \in \mathbb{N}$ be a prime, let $k, c, h \in \mathbb{N}$ and let $C \in \mathbb{Z}_p^{c \times h}$, $K \in \mathbb{Z}_p^{k \times h}$. The following are equivalent:*

- ***$\boldsymbol{h}$-hiding:*** $\boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^h$; *return* $(C\boldsymbol{h}, K\boldsymbol{h}) \equiv \boldsymbol{h}, \boldsymbol{h}' \xleftarrow{\$} \mathbb{Z}_p^h$; *return* $(C\boldsymbol{h}, K\boldsymbol{h}')$
- ***non-reconstructability*** *For every $\boldsymbol{m}_c \in \mathbb{Z}_p^c$ and very $\boldsymbol{m}_k \in \mathbb{Z}_p^k$, either $\boldsymbol{m}_c^\top C \neq \boldsymbol{m}_k^\top K$ or $\boldsymbol{m}_c^\top C = \mathbf{0}_h^\top$.*

where $\equiv$ denotes equality of distributions.

A consequence of Theorem 8 is that every valid tag-based encoding is perfectly hiding, or equivalently, there cannot exist a tag-based encoding where the two distributions from $\boldsymbol{h}$-hiding are negligibly close but not identical.

Thanks to the above theorem, it is possible to define *disjunction* and *conjunction* transformations for tag-based encodings along the lines of predicate encodings. We were not able to design a negation transformation for tag-based encodings and leave it for future work. On the other hand, the dual transformation is straightforward in this framework, as mentioned in [19], because the encoding primitives are completely symmetric.

Expressivity of tag-based encodings vs predicate encodings We try to improve our understanding of the differences between these two primitives by providing a transformation that produces valid predicate encodings from valid tag-based encodings for the same predicate.

Theorem 9. *Given a $(c, 1, h)$-tag-based encoding $(\mathsf{cE}, \mathsf{kE})$ for $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, the $(c, 1, h)$-partial predicate encoding $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}')$ defined as $\mathsf{sE}'_x = \mathsf{cE}_x$, $\mathsf{rE}'_y = \mathsf{kE}_y$, $\mathsf{kE}'_y = \begin{pmatrix} 1 \end{pmatrix}$, induces a predicate encoding for P.*

Note that because of the symmetry of tag-based encodings, Theorem 9 can be also applied to $(1, k, h)$-tag-based encodings. All the tag-based encodings proposed in [19] (except one) have either $c = 1$ or $k = 1$, so the above theorem can be applied to them.

5 Pair Encodings

In this section we provide an embedding that transforms every predicate encoding into an information-theoretic pair encoding. Consequently, we can see predicate encodings as a subclass of pair encodings. This opens the possibility of reusing the conjunction and dual transformation proposed by Attrapadung [7, 8] for pair encodings, to create combinations of predicate encodings via our embedding. We show that this alternative method is fundamentally different from our direct conjunction and dual transformations on predicate encodings, where our combinations produce more efficient encodings.

5.1 Embedding Predicate Encodings into Pair Encodings

In this section we provide an embedding that produces a valid *information-theoretic pair encoding* from every valid predicate encoding (see Definitions 1 and 3 for predicate encodings and pair encodings respectively).

Definition 5 (Embedding to Pair Encodings). *Given a (s, r, w)-predicate encoding $pe = (\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$, we define the embedding $\mathsf{Emb}(pe) = (\mathsf{Enc1}_{pe}, \mathsf{Enc2}_{pe}, \mathsf{Pair}_{pe})$ as follows:*

- $\mathsf{Enc1}_{pe}(x) = (c_0, \boldsymbol{c})$, *where* $c_0(s_0, \boldsymbol{h}) = s_0$, $\boldsymbol{c}(s_0, \boldsymbol{h}) = s_0 \cdot \mathsf{sE}_x \boldsymbol{h}$
- $\mathsf{Enc2}_{pe}(y) = (k_0, \boldsymbol{k})$, *where* $k_0(\alpha, r_1, \boldsymbol{h}) = r_1$, $\boldsymbol{k}(\alpha, r_1, \boldsymbol{h}) = \alpha \cdot \mathsf{kE}_y + r_1 \cdot \mathsf{rE}_y \boldsymbol{h}$
- $\mathsf{Pair}_{pe}(x, y) = \begin{pmatrix} 0 & \mathsf{rD}_{x,y}^{\top} \\ \text{-}\mathsf{sD}_{x,y} & \boldsymbol{0}_{s,r} \end{pmatrix}$

All variables $\boldsymbol{s} = (s_0)$ and $\boldsymbol{r} = (r_1)$ appear in all the $\mathsf{Enc1}$ and $\mathsf{Enc2}$ polynomials respectively. This simplifies the pair encoding's information-theoretical security notion into one equivalent to the privacy of the predicate encoding (see Proof A.9).

Theorem 10 (Correctness of the embedding). *If $pe = (\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ is a valid (s, r, w)-predicate encoding for P, then $\mathsf{Emb}(pe)$ is a valid information theoretic $(s + 1, r + 1, w)$-pair encoding for P.*

Our embedding shows that every predicate encoding can be transformed into a perfectly secure pair encoding. In fact, after applying the compiler from [1] to the embedding of a predicate encoding, we get the same predicate encryption scheme that the one provided by the compiler from [12].

We conclude that *predicate encodings* can be transformed into a very special class of *pair encodings*: encodings that allow decryption with 2 pairings and have only one element of randomness in both, ciphertexts and secret keys (what makes them very efficient).

5.2 Comparison between encoding transformations

Attrapadung proposed generic transformations of pair encodings [7, 8]. In particular, he proposed the conjunction and dual transformations. In this section

we compare these transformations with the ones proposed in this work. For this, we compare the conjunction of two pair encodings, (embedded from predicate encodings) with the embedding of the conjunction of a (s_1, r_1, w_1)-predicate encoding $pe^1 = (\mathsf{sD}^1, \mathsf{rE}^1, \mathsf{kE}^1, \mathsf{sD}^1, \mathsf{rD}^1)$ and a (s_2, r_2, w_2)-predicate encoding $pe^2 = (\mathsf{sD}^2, \mathsf{rE}^2, \mathsf{kE}^2, \mathsf{sD}^2, \mathsf{rD}^2)$, i.e.,

$$\mathsf{Emb}(pe^1 \wedge_{pred} pe^2) \quad vs \quad \mathsf{Emb}(pe^1) \wedge_{pair} \mathsf{Emb}(pe^2)$$

where $\wedge_{pred}$ and $\wedge_{pair}$ are the conjunction of predicate encodings and pair encodings respectively. Note that $\wedge_{pred}$ corresponds to the transformation from our Theorem 5. On the other hand, for $\wedge_{pair}$ we use the conjunction proposed in [8].

$$\mathsf{Emb}(pe^1 \wedge_{pred} pe^2) = \begin{cases} \mathsf{Enc1}((x_1, x_2)) = (c_0, \boldsymbol{c}_1, \boldsymbol{c}_2) \\ \mathsf{Enc2}((y_1, y_2)) = (k_0, \boldsymbol{k}_1, \boldsymbol{k}_2) \\ \mathsf{Pair}((x_1, x_2), (y_1, y_2)) = E_{(x_1,x_2),(y_1,y_2)} \end{cases}$$

where $\boldsymbol{h} = (h_0, \boldsymbol{h}_1, \boldsymbol{h}_2)$ and

$$\begin{array}{ll} c_0(s_0, \boldsymbol{h}) = s_0 & k_0(\alpha, r_1, \boldsymbol{h}) = r_1 \\ \boldsymbol{c}_1(s_0, \boldsymbol{h}) = s_0 \cdot \mathsf{sE}^1_{x_1} \boldsymbol{h}_1 & \boldsymbol{k}_1(\alpha, r_1, \boldsymbol{h}) = (\alpha + h_0) \cdot \mathsf{kE}^1_{y_1} + r_1 \cdot \mathsf{rE}^1_{y_1} \boldsymbol{h}_1 \\ \boldsymbol{c}_2(s_0, \boldsymbol{h}) = s_0 \cdot \mathsf{sE}^2_{x_2} \boldsymbol{h}_2 & \boldsymbol{k}_2(\alpha, r_1, \boldsymbol{h}) = (\alpha - h_0) \cdot \mathsf{kE}^2_{y_2} + r_1 \cdot \mathsf{rE}^2_{y_2} \boldsymbol{h}_2 \end{array}$$

$$E_{(x_1,x_2),(y_1,y_2)} = \frac{1}{2} \begin{pmatrix} 0 & \mathsf{rD}^{1\top}_{x_1,y_1} & \mathsf{rD}^{2\top}_{x_2,y_2} \\ \text{-}\mathsf{sD}^1_{x_1,y_1} & \boldsymbol{0}_{s_1,r_1} & \boldsymbol{0}_{s_1,r_2} \\ \text{-}\mathsf{sD}^2_{x_2,y_2} & \boldsymbol{0}_{s_2,r_1} & \boldsymbol{0}_{s_2,r_2} \end{pmatrix}$$

$$\mathsf{Emb}(pe^1) \wedge_{pair} \mathsf{Emb}(pe^2) = \begin{cases} \mathsf{Enc1}((x_1, x_2)) = (c_0, \boldsymbol{c}_1, \boldsymbol{c}_2) \\ \mathsf{Enc2}((y_1, y_2)) = (k_0, \boldsymbol{k}_1, k_2, \boldsymbol{k}_3) \\ \mathsf{Pair}((x_1, x_2), (y_1, y_2)) = E_{(x_1,x_2),(y_1,y_2)} \end{cases}$$

where $\boldsymbol{h} = (\boldsymbol{h}_1, \boldsymbol{h}_2)$ and

$$\begin{array}{ll} c_0(s_0, \boldsymbol{h}) = s_0 & k_0(\alpha, (r_1, r_2, r_3), \boldsymbol{h}) = r_1 \\ \boldsymbol{c}_1(s_0, \boldsymbol{h}) = s_0 \cdot \mathsf{sE}^1_{x_1} \boldsymbol{h}_1 & \boldsymbol{k}_1(\alpha, (r_1, r_2, r_3), \boldsymbol{h}) = r_3 \cdot \mathsf{kE}^1_{y_1} + r_1 \cdot \mathsf{rE}^1_{y_1} \boldsymbol{h}_1 \\ \boldsymbol{c}_2(s_0, \boldsymbol{h}) = s_0 \cdot \mathsf{sE}^2_{x_2} \boldsymbol{h}_2 & k_2(\alpha, (r_1, r_2, r_3), \boldsymbol{h}) = r_2 \\ & \boldsymbol{k}_3(\alpha, (r_1, r_2, r_3), \boldsymbol{h}) = (\alpha - r_3) \cdot \mathsf{kE}^2_{y_2} + r_2 \cdot \mathsf{rE}^2_{y_2} \boldsymbol{h}_2 \end{array}$$

$$E_{(x_1,x_2),(y_1,y_2)} = \begin{pmatrix} 0 & \mathsf{rD}^{1\top}_{x_1,y_1} & 0 & \mathsf{rD}^{2\top}_{x_2,y_2} \\ \text{-}\mathsf{sD}^1_{x_1,y_1} & \boldsymbol{0}_{s_1,r_1} & \boldsymbol{0}_{s_1} & \boldsymbol{0}_{s_1,r_2} \\ \boldsymbol{0}_{s_2} & \boldsymbol{0}_{s_2,r_1} & \text{-}\mathsf{sD}^2_{x_2,y_2} & \boldsymbol{0}_{s_2,r_2} \end{pmatrix}$$

The resulting pair encodings are different. The first one (result of our conjunction) does not introduce new random variables and does not increase the number of pairings for decryption. On the other hand, the second conjunction adds new random variables to key generation and increases the number of pairings needed during decryption. This overhead will be amplified if nested conjunctions are used.

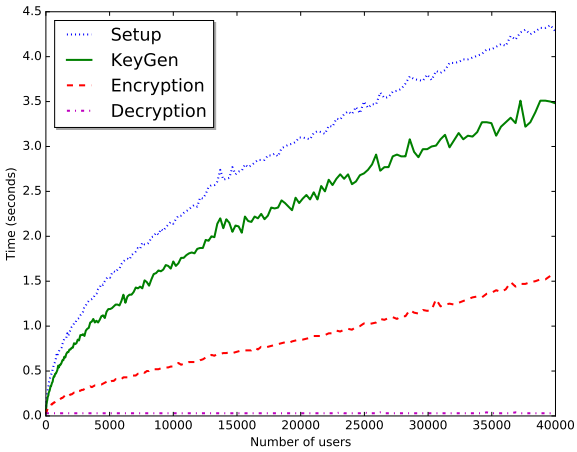

Fig. 1. Scalability of the predicate encryption for revocation

6 Constructions

We provide new instances of predicate encodings to achieve predicate encryption schemes with new properties or better performance.

6.1 Combining predicates

Dual-Policy ABE Dual-Policy Attribute Based Encryption [7, 8] has already been considered in the *pair encodings* framework. It combines KP-ABE and CP-ABE into a single construction that simultaneously allows two access control mechanisms. The main advantage is the possibility of considering policies over *objective* attributes (associated to data) and policies over *subjective* attributes (associated to user credentials) at the same time.

Our combinations of predicate encodings allow us to create predicate encryption constructions for Dual-Policy ABE in the framework of pair encodings and tag-based encodings. In particular, given an arbitrary predicate encoding for $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, applying Theorems 7 and 5 we get an encoding for DP-ABE, i.e., for the predicate $\mathsf{P}^\star : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{Y} \times \mathcal{X}) \to \{0,1\}$ defined as

$$\mathsf{P}^\star((x,y),(y',x')) = 1 \text{ iff } \mathsf{P}(x,y) \wedge \mathsf{P}(y',x')$$

Revocation Another application of our combinations is predicate encryption with revocation, by combining a *boolean formula predicate encoding* with a *broadcast encryption predicate encoding*. The former is used to encode the actual policy of the scheme, while the latter takes care of revocation.

Broadcast encryption has been considered in the literature to approach revocation [16, 20, 26]. In broadcast encryption, a broadcasting authority encrypts

P	$\bar{\mathsf{P}}$
attributes $= \{a, b, c, d\}$	attributes $= \{\bar{a}, \bar{b}, \bar{c}, \bar{d}\}$
$x = (a \wedge c) \vee d$	$x = (\bar{a} \vee \bar{c}) \wedge \bar{d}$
$y = \{a, c\}$	$y = \{\bar{c}, \bar{d}\}$
$\mathsf{P}(x, y) = 1$ iff $x(y)$	$\bar{\mathsf{P}}(x, y) = 1$ iff $\neg x(y)$

Fig. 2. Equivalent encodings of a policy using P (CP-ABE) on the left and $\bar{\mathsf{P}}$ (negated CP-ABE) on the right.

a message in such a way that only authorized users will be able to decrypt it. This can be expressed with the predicate $\mathsf{P}(\boldsymbol{x}, i) = 1$ iff $\boldsymbol{x}_i = 1$, where $\boldsymbol{x} \in \mathcal{X} = \{0,1\}^n, i \in \mathcal{Y} = [n]$. A drawback is that the number of users in the system, n, is polynomial size. Figure 1 shows the performance of predicate encryption built from a predicate encoding that combines boolean formulas with broadcast encryption. The system supports thousands of users in reasonable time.

6.2 Improved predicate encodings

In this section we propose new predicate encodings that are more efficient than some of the encodings proposed previously in [12]. Our encodings are built by applying Theorem 6 to obtain negated encodings and observing that, in some cases, Theorem 3 can be applied to simplify the negated version into a more efficient encoding than the original one. The predicate associated to this new encoding is negated, but if inputs are also negated, the predicate will be equivalent. Figure 2 illustrates this idea. On the left, there is a boolean formula CP-ABE for 4 attributes $\{a, b, c, d\}$. On the right side, secret keys and policies are modified so that the negated version is equivalent. The attribute universe is formed by the *negated attributes*, secret keys are formed by all *negated attributes* do not appear in the original key as normal attributes, policies are negated and expressed in NNF (Negation Normal Form).

Boolean formulas In [12], the authors propose two predicate encoding (KP and CP versions) for monotonic boolean formulas. The predicate they consider is a particular case of a Linear Secret Sharing scheme [21]. Let $\mathcal{X} = \{0,1\}^n, \mathcal{Y} = \mathbb{Z}_p^{n \times k}$ for some $n, k \in \mathbb{N}$,

$$\mathsf{P}(\boldsymbol{x}, M) = 1 \text{ iff } \begin{pmatrix} 1 & 0 & \overset{k\text{-}1}{\cdots} & 0 \end{pmatrix} \in \underset{span}{row} \langle M_{\boldsymbol{x}} \rangle$$

where $M_{\boldsymbol{x}}$ denotes the matrix M filtered by $\boldsymbol{x}$, i.e., $M_{\boldsymbol{x}}$ includes the i-th row of M iff $\boldsymbol{x}_i = 1$.

It has been shown [25] that for every[5] monotonic boolean formula f with attributes from $\mathcal{X}$ there exists a matrix $M \in \mathcal{Y}$ such that for every $\boldsymbol{x} \in \mathcal{X}$,

[5] where every attribute appears at most once and the number of *and-gates* is lower than k (one could overcome the one-use restriction by considering duplicated attributes)

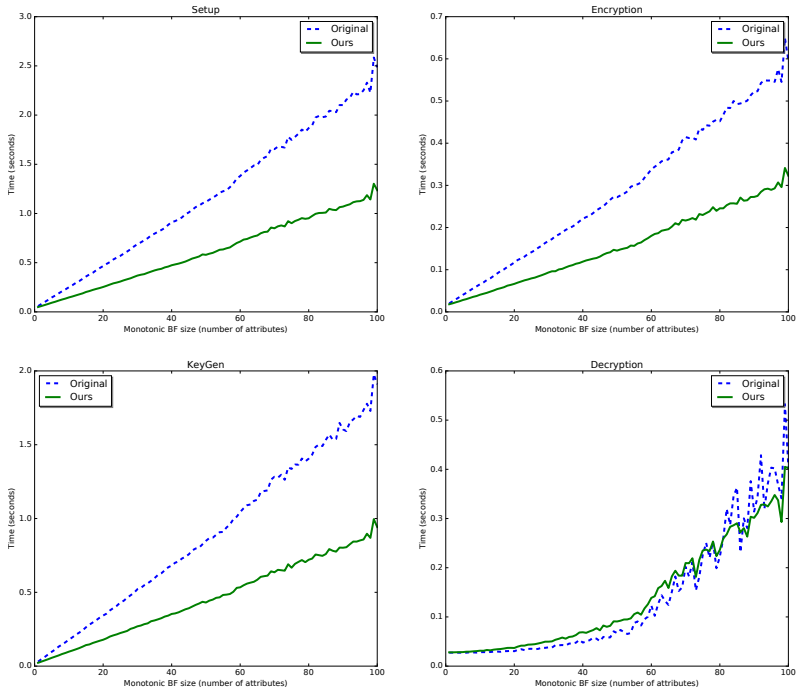

Fig. 3. Improved predicate encoding for boolean formulas vs original encoding

$f(\boldsymbol{x}) \Leftrightarrow \mathsf{P}(\boldsymbol{x}, M)$. The key-policy predicate encoding from [12] is the following,

$$\mathsf{sE}_{\boldsymbol{x}} = \begin{pmatrix} \mathsf{diag}(\boldsymbol{x}) & \mathbf{0}_{n,k-1} \end{pmatrix} \qquad \mathsf{rE}_M = \begin{pmatrix} I_n & M_{\{2,\ldots,k\}} \end{pmatrix} \qquad \mathsf{kE}_M = \begin{pmatrix} M_{\{1\}} \end{pmatrix}$$

where $M_{\{1\}}$ denotes the first column of matrix M, $M_{\{2,\ldots,k\}}$ represents the rest of the matrix. We do not include explicit decryption functions sD and rD, but they can be computed efficiently by *Gaussian elimination*.

In the above encoding, the number of elements in secret keys and ciphertexts is always maximal, it equals the number of (possibly duplicated) attributes, even for small policies. Furthermore, the maximum number of *and-gates* in a policy must be fixed a priori (because it is related the the number of columns in the matrix).

We propose the following improved predicate encoding for (negated) key-policy monotonic boolean formulas, which is an equivalent predicate if instantiated with negated inputs. Let $\mathcal{X} = \{0,1\}^n$ and $\mathcal{Y} = \mathbb{Z}_p^{n \times k}$,

$$\mathsf{sE}_{\boldsymbol{x}} = I_n - \mathsf{diag}(\boldsymbol{x}) \qquad \mathsf{rE}_M = M^\top \qquad \mathsf{kE}_M = \begin{pmatrix} 1 & 0 & \overset{k\text{-}1}{\cdots} & 0 \end{pmatrix}^\top$$

In our encoding, the number of columns has been reduced up to half[6]. Furthermore, the size of secret keys is proportional to the complexity of policies. In

[6] being half when the bound on the number of *and-gates* is maximal

particular, it is equal to the number of *and-gates* in the policy (or equivalently, the number of *or-gates* in the non-negated version). Note that our improvement also works in the ciphertext-policy case.

In Figure 3 we present a comparison between our improved encoding for *key-policy monotonic boolean formulas* and the original one. To this end, we generate random boolean formulas for different sizes, starting from a random set of leaf nodes and combining them with boolean operators $\vee$ and $\wedge$. Our tables report on the average time for each algorithm. Our encoding needs 50% less time than the original algorithms for setup, encryption and key generation. For decryption the performance is similar. We note that all the analyzed predicate encryption schemes were instantiated with the same compiler and therefore all of them are secure (under SXDH assumption) with the same level of security, but in terms of secret key size, our encoding is smaller in general (in the worst case, when all the gates in the policy are *or-gates*, key sizes are equal).

Arithmetic span programs. Chen et al. proposed in [12] a predicate encoding for *Arithmetic Span Programs (ASP)*. This is, an encoding for the predicate P defined as follows. Let $\mathcal{X} = \mathbb{Z}_p^n$, $\mathcal{Y} = \mathbb{Z}_p^{n\times k} \times \mathbb{Z}_p^{n\times k}$, for some $n, k \in \mathbb{N}$; for every $\boldsymbol{x} \in \mathcal{X}$ and every $(Y, Z) \in \mathcal{Y}$,

$$\mathsf{P}(\boldsymbol{x}, (Y, Z)) = 1 \text{ iff } \begin{pmatrix} 1 & 0 & \overset{k-1}{\cdots} & 0 \end{pmatrix} \in \underset{span}{row} \langle \mathsf{diag}(\boldsymbol{x})Y + Z \rangle$$

In [18], Ishai and Wee show how to relate *Arithmetic Span Programs* computations of polynomial functions over a finite field $\mathbb{F}$, i.e., functions $f : \mathbb{F}^n \to \mathbb{F}$ that only use addition and multiplication over the field. Therefore, the above predicate can be seen as $f(\boldsymbol{x}) = 0$, where f is the polynomial function induced by (Y, Z). Let $\mathcal{X} = \mathbb{Z}_p^n$, $\mathcal{Y} = \mathbb{Z}_p^{n\times k} \times \mathbb{Z}_p^{n\times k}$, the original predicate encoding for arithmetic span programs proposed in [12] is the following:

$$\mathsf{sE}_x = \begin{pmatrix} \mathsf{diag}(\boldsymbol{x}) & I_n & \mathbf{0}_{n,k\text{-}1} \end{pmatrix} \quad \mathsf{rE}_{(Y,Z)} = \begin{pmatrix} I_n & \mathbf{0}_{n,n} & Y_{\{2,\ldots,l\}} \\ \mathbf{0}_{n,n} & I_n & Z_{\{2,\ldots,l\}} \end{pmatrix} \quad \mathsf{kE}_{(Y,Z)} = \begin{pmatrix} Y_{\{1\}} \\ Z_{\{1\}} \end{pmatrix}$$

We present a more efficient encoding for (negated[7]) arithmetic span programs:

$$\mathsf{sE}_x = \begin{pmatrix} \mathsf{diag}(\boldsymbol{x}) & \text{-}I_n \end{pmatrix} \quad \mathsf{rE}_{(Y,Z)} = \begin{pmatrix} Z^\top & Y^\top \end{pmatrix} \quad \mathsf{kE}_{(Y,Z)} = \begin{pmatrix} 1 & 0 & \overset{k-1}{\cdots} & 0 \end{pmatrix}^\top$$

Figure 4 shows the performance of our new encoding for KP-ABE for Arithmetic Span Programs compared to the original encoding from [12]. As we expected, our encoding needs 66% of the time required for the original encoding for setup, encryption and key generation. Additionally, secret key size is halved with our encoding.

6.3 Extra features

In this section we consider new theoretical results that can be proved thanks to our algebraic characterization of α-privacy and can be used to produce new predicate encodings enhanced with extra properties.

[7] in [18] there is a modification of their algorithm that produces matrices (Y, Z) such that the predicate associated is $f(\boldsymbol{x}) \neq 0$ (the double negation will cancel out)

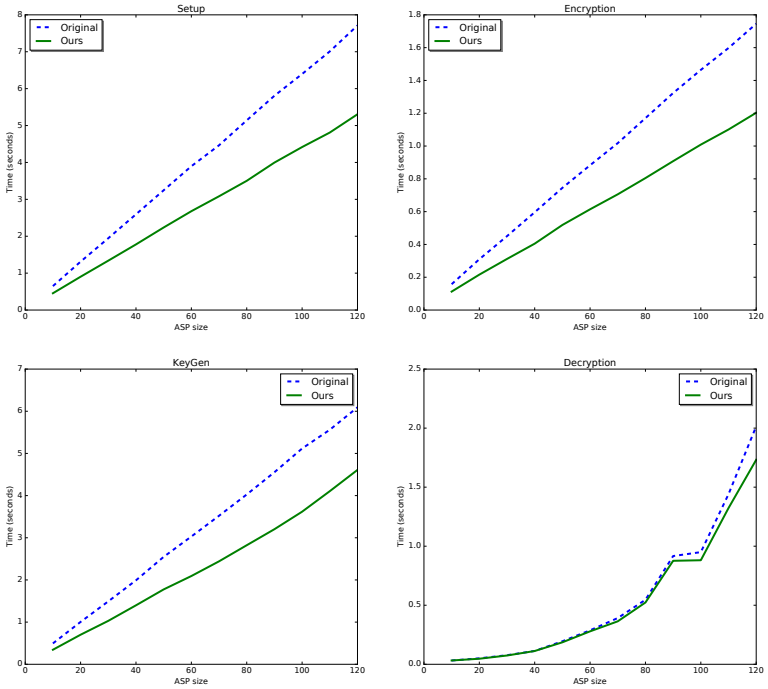

Fig. 4. Improved predicate encoding for ASP vs original encoding

Attribute-hiding for boolean formulas Chen et al. proposed an extension of the compiler in [12] to build weakly attribute-hiding predicate encryption schemes [11, 22]. In a weakly attribute-hiding scheme, the ciphertext attribute x remains secret for unauthorized users, that only learn the fact that their secret keys are not valid. This additional compiler needs to be instantiated with *predicate encodings* satisfying additional properties. The following is a definition from [12].

Definition 6 (Attribute-Hiding Encodings). *A (s, r, w)-predicate encoding, $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ is* attribute-hiding *if it verifies the additional requirements:*

x-oblivious reconstruction: $\mathsf{sD}_{x,y}$ *and* $\mathsf{rD}_{x,y}$ *are independent of x.*

attribute-hiding: *for all $(x, y) \notin \mathsf{P}$,*

$$\boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathit{return}\ (\mathsf{sE}_x\boldsymbol{w},\ \mathsf{rE}_y\boldsymbol{w}) \quad \equiv \quad \boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_p^s; \boldsymbol{r} \xleftarrow{\$} \mathbb{Z}_p^r;\ \mathit{return}\ (\boldsymbol{s},\ \boldsymbol{r})$$

where $\equiv$ denotes equality of distributions.

The following theorem relates these two properties with our alternative definition of predicate encodings:

Theorem 11 (Algebraic characterization of attribute-hiding). *Let $p \in \mathbb{N}$ be a prime, let $s, r, w \in \mathbb{N}$ and let $S \in \mathbb{Z}_p^{s \times w}$, $R \in \mathbb{Z}_p^{r \times w}$, $\boldsymbol{k} \in \mathbb{Z}_p^r$. The following are equivalent:*

- $\boldsymbol{w} \xleftarrow{\$} \mathbb{Z}_p^w$; *return* $(S\boldsymbol{w}, R\boldsymbol{w}) \equiv \boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_p^s$; $\boldsymbol{r} \xleftarrow{\$} \mathbb{Z}_p^r$; *return* $(\boldsymbol{s}, \boldsymbol{r})$
- $\operatorname{rank}\begin{pmatrix} S \\ R \end{pmatrix} = \operatorname{rank}(S) + \operatorname{rank}(R)$

Note that for every (s, r, w)-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ that is attribute-hiding, there exists an equivalent $(s, 1, w)$-predicate encoding. This is because rD is independent from x and thus, we can apply our optimization Theorem 3 with matrices $\mathsf{B}_y = \mathsf{rD}_{x,y}^\top \in \mathbb{Z}_p^{1 \times w}$, $\mathsf{A}_x = I_s$, $\mathsf{C} = I_w$. Therefore, the class of predicates that can be built from attribute-hiding encodings is included in the class of predicates achieved from $(s, 1, w)$-predicate encodings.

Further, note that our *disjunction* and *conjunction* combinations for predicate encodings (Theorems 4 and 5 respectively) preserve the notion of attribute-hiding[8]. Exploiting this fact, we propose a Policy-Hiding ABE scheme for non-monotonic boolean formulas expressed in DNF (Disjunctive Normal Form). The inner product can be used to encode conjunctions [22]. More concretely, let $\boldsymbol{y} \in \{0,1\}^n \subseteq \mathbb{Z}_p^n$. We establish that the i-th attribute a_i appears in a secret key for $\boldsymbol{y}$ iff $\boldsymbol{y}_i = 1$. Let $S, \bar{S} \subseteq \{a_i\}_{i=1}^n$ be sets such that $S \cap \bar{S} = \emptyset$,

$$\bigwedge_{a \in S} a \ \wedge \ \bigwedge_{a \in \bar{S}} \bar{a} \ \Leftrightarrow^9 \ \boldsymbol{x}^\top \boldsymbol{y} = |S| \qquad \text{where } \forall i \in [n], \ \boldsymbol{x}_i = \begin{cases} 1 & \text{if } a_i \in S \\ -1 & \text{if } a_i \in \bar{S} \\ 0 & \text{otherwise} \end{cases}$$

Note that the ZIPE predicate encoding from [12, Appendix A.1] can be modified into an attribute-hiding encoding for the predicate $\mathsf{P}((\boldsymbol{x}, \gamma), \boldsymbol{y}) = 1$ iff $\boldsymbol{x}^\top \boldsymbol{y} = \gamma$ (see Appendix C.1).

Therefore, with a disjunction of k predicate encodings like the former we can encode boolean formulas that have at most k disjuncts. Note that the resulting encoding is *attribute-hiding* but it is not *x-oblivious*. However, without the knowledge of the policy x, one can guess for the disjunct his secret key satisfies (if any). In this way, a valid key will be enough to decrypt after at most k decryption tries (one for every disjunct).

Delegation Delegation of keys is a desirable property for every predicate encryption scheme. Roughly, it allows the owner of a secret key to weaken his key creating a new one that is less powerful than the original one. This property can be used to achieve *forward secrecy*, where past sessions are protected from the compromise of future secret keys. To achieve this goal, users can periodically weaken their secret keys and destroy the more powerful ones. In this way, past ciphertexts cannot be decrypted any more and thus, they are protected against the

[8] conjunction also preserves x-oblivious reconstruction, while disjunction does not

[9] this equivalence holds when $|S| < p$, but in practice p is a large prime

Let $\mathcal{U} = \{a, b, c\}$ be the set of attributes. We consider the predicate encoding for monotonic boolean formulas from [12]. Let $\mathcal{X} = \{0,1\}^3, \mathcal{Y} = \mathbb{Z}_p^{3\times 2}$,

$$\mathsf{sE}_{\boldsymbol{x}} = \begin{pmatrix} \mathsf{diag}(\boldsymbol{x}) & \mathbf{0}_{3,2} \end{pmatrix} \qquad \mathsf{rE}_M = \begin{pmatrix} I_3 & M_{\{2\}} \end{pmatrix} \qquad \mathsf{kE}_M = \begin{pmatrix} M_{\{1\}} \end{pmatrix}$$

The following is the encoding of a key for the formula $(a \vee c) \wedge b$, enhanced for delegation according to Theorem 12 (with $k = 1$),

$$\mathsf{rE}_M = \begin{pmatrix} 1&0&0&1&0 \\ 0&1&0&1&0 \\ 0&0&1&1&0 \\ 0&0&0&0&1 \end{pmatrix} \qquad \mathsf{kE}_M = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Let's assume we want to weaken this key to one for the formula $a \wedge b \wedge c$. Note that in this case we want to make an update of the matrix M:

$$M = \begin{pmatrix} 1&1 \\ 0&1 \\ 1&1 \end{pmatrix} \text{ encodes } (a \vee c) \wedge b \qquad M' = \begin{pmatrix} 1&1&1 \\ 0&1&1 \\ 1&1&0 \end{pmatrix} \text{ encodes } a \wedge b \wedge c$$

It can be done by multiplying rE_M from the left by A

$$\mathsf{rE}'_M = \underbrace{\begin{pmatrix} 1&0&0&1 \\ 0&1&0&1 \\ 0&0&1&0 \\ 0&0&0&0 \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} 1&0&0&1&0 \\ 0&1&0&1&0 \\ 0&0&1&1&0 \\ 0&0&0&0&1 \end{pmatrix}}_{\mathsf{rE}_M} = \begin{pmatrix} 1&0&0&1&1 \\ 0&1&0&1&1 \\ 0&0&1&1&0 \\ 0&0&0&0&0 \end{pmatrix} \qquad \mathsf{kE}'_M = A \cdot \mathsf{kE}_M = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Fig. 5. Example of delegation of keys for monotonic boolean formulas. Since A is a linear function, it can be computed in the exponent from the given key.

theft of future keys. Delegation is also required for Hierarchical Identity Based Encryption (HIBE). More formally, we say that a predicate $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ supports delegation if there is a partial ordering $\preceq$ on $\mathcal{Y}$ such that for every $x \in \mathcal{X}$, if $\mathsf{P}(x, y) = 1$ and $(y \preceq y')$, then $\mathsf{P}(x, y') = 1$.

Delegation has been considered in [12] as the property of some predicate encodings. We propose a generic method to convert any predicate encoding into one supporting delegation.

Theorem 12 (Delegation). *For every (s, r, w)-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for $\mathsf{P} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, for every $k \in \mathbb{N}$, $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}', \mathsf{sD}', \mathsf{rD}')$ defined below is a valid $(s, r + k, w + k)$-predicate encoding for P.*

$$\mathsf{sE}'_x = \begin{pmatrix} \mathsf{sE}_x & \mathbf{0}_{s,k} \end{pmatrix} \qquad \mathsf{rE}'_y = \begin{pmatrix} \mathsf{rE}_y & \mathbf{0}_{r,k} \\ \mathbf{0}_{k,w} & I_k \end{pmatrix} \qquad \mathsf{kE}'_y = \begin{pmatrix} \mathsf{kE}_y \\ \mathbf{0}_k \end{pmatrix}$$

$$\mathsf{sD}'_{x,y} = \mathsf{sD}_{x,y} \qquad \mathsf{rD}'_{x,y} = \begin{pmatrix} \mathsf{rD}_{x,y} \\ \mathbf{0}_k \end{pmatrix}$$

The additional set of not-null rows in rE'_y can be used to weaken the linear span of rE_y, what directly modifies the predicate. In particular, this method works really well for monotonic boolean formulas. (See Figure 5 for an example).

Acknowledgements The work presented here was supported by projects S2013/ICE-2731 N-GREENS Software-CM, ONR Grants N000141210914 and N000141512750, as well as FP7 Marie Curie Actions-COFUND 291803.

References

1. S. Agrawal and M. Chase. *A Study of Pair Encodings: Predicate Encryption in Prime Order Groups*, pages 259–288. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.
2. J. A. Akinyele, C. U. Lehmann, M. D. Green, M. W. Pagano, Z. N. J. Peterson, and A. D. Rubin. Self-protecting electronic medical records using attribute-based encryption. Cryptology ePrint Archive, Report 2010/565, 2010. `http://eprint.iacr.org/2010/565`.
3. D. F. Aranha and C. P. L. Gouvêa. RELIC is an Efficient LIbrary for Cryptography. `https://github.com/relic-toolkit/relic`.
4. N. Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 557–577. Springer, May 2014.
5. N. Attrapadung. Dual system encryption framework in prime-order groups. Cryptology ePrint Archive, Report 2015/390, 2015. `http://eprint.iacr.org/2015/390`.
6. N. Attrapadung and H. Imai. *Conjunctive Broadcast and Attribute-Based Encryption*, pages 248–265. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
7. N. Attrapadung and H. Imai. *Dual-Policy Attribute Based Encryption*, pages 168–185. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
8. N. Attrapadung and S. Yamada. Duality in abe: Converting attribute based encryption for dual predicate and dual policy via computational encodings. Cryptology ePrint Archive, Report 2015/157, 2015. `http://eprint.iacr.org/2015/157`.
9. P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. Cryptology ePrint Archive, Report 2005/133, 2005. `http://eprint.iacr.org/2005/133`.
10. A. Beimel. *Secret-Sharing Schemes: A Survey*, pages 11–46. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
11. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In S. P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 535–554. Springer, Feb. 2007.
12. J. Chen, R. Gay, and H. Wee. Improved dual system ABE in prime-order groups via predicate encodings. In *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 595–624. Springer, 2015.
13. J. Chen and H. Wee. Fully, (almost) tightly secure IBE and dual system groups. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 435–460. Springer, Aug. 2013.
14. J. Chen and H. Wee. Dual system groups and its applications — compact hibe and more. Cryptology ePrint Archive, Report 2014/265, 2014. `http://eprint.iacr.org/2014/265`.
15. T. T. A. Dinh and A. Datta. Streamforce: outsourcing access control enforcement for stream data to the clouds. In *Fourth ACM Conference on Data and Application Security and Privacy, CODASPY'14, San Antonio, TX, USA - March 03 - 05, 2014*, pages 13–24, 2014.

16. S. Garg, A. Kumarasubramanian, A. Sahai, and B. Waters. Building efficient fully collusion-resilient traitor tracing and revocation schemes. In E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, editors, *ACM CCS 10*, pages 121–130. ACM Press, Oct. 2010.
17. R. Gay, I. Kerenidis, and H. Wee. Communication complexity of conditional disclosure of secrets and attribute-based encryption. In *CRYPTO 2015, Part II*, LNCS, pages 485–502. Springer, Aug. 2015.
18. Y. Ishai and H. Wee. *Partial Garbling Schemes and Their Applications*, pages 650–662. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.
19. F. G. Jongkil Kim, Willy Susilo and M. H. Au. A tag based encoding: An efficient encoding for predicate encoding in prime order groups. Cryptology ePrint Archive, Report 2016/655, 2016. `http://eprint.iacr.org/2016/655`.
20. P. Junod and A. Karlov. An efficient public-key attribute-based broadcast encryption scheme allowing arbitrary access policies. In *Proceedings of the Tenth Annual ACM Workshop on Digital Rights Management*, DRM '10, pages 13–24, New York, NY, USA, 2010. ACM.
21. M. Karchmer and A. Wigderson. On span programs. In *In Proc. of the 8th IEEE Structure in Complexity Theory*, pages 102–111. IEEE Computer Society Press, 1993.
22. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, Apr. 2008.
23. A. B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 318–335. Springer, Apr. 2012.
24. A. B. Lewko and B. Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 455–479. Springer, Feb. 2010.
25. A. B. Lewko and B. Waters. Decentralizing attribute-based encryption. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 568–588. Springer, May 2011.
26. Z. Liu and D. S. Wong. *Practical Ciphertext-Policy Attribute-Based Encryption: Traitor Tracing, Revocation, and Large Universe*, pages 127–146. Springer International Publishing, Cham, 2015.
27. D. Lubicz and T. Sirvent. Attribute-based broadcast encryption scheme made efficient. In S. Vaudenay, editor, *AFRICACRYPT 08*, volume 5023 of *LNCS*, pages 325–342. Springer, June 2008.
28. A. Sahai, H. Seyalioglu, and B. Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 199–217. Springer, Aug. 2012.
29. A. Sahai and B. R. Waters. Fuzzy identity-based encryption. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, May 2005.
30. A. Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and D. Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, Aug. 1984.
31. F. Wang, J. Mickens, N. Zeldovich, and V. Vaikuntanathan. Sieve: Cryptographically enforced access control for user data in untrusted clouds. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pages 611–626, Santa Clara, CA, Mar. 2016. USENIX Association.

32. B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, Aug. 2009.
33. H. Wee. Dual system encryption via predicate encodings. In Y. Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 616–637. Springer, Feb. 2014.

A Proofs from main body

A.1 Proof of Theorem 2

Proof. The proof follows from Theorem 1 and the observation that reconstructability of predicate encodings is equivalent to Pred, while privacy of predicate encodings is equivalent to $\neg\,\mathsf{Pred}$.

A.2 Proof of Theorem 3

Proof. To see correctness of the new encoding, note that for all $(x, y) \in \mathsf{P}$, since

$$\mathsf{sD}_{x,y}^{\top} \in \underset{span}{row}\langle \mathsf{A}_x \rangle \ \wedge\ \mathsf{rD}_{x,y}^{\top} \in \underset{span}{row}\langle \mathsf{B}_y \rangle$$

there exist $\mathsf{sD'}_{x,y}^{\top}$ and $\mathsf{rD'}_{x,y}^{\top}$ such that

$$\mathsf{sD}_{x,y}^{\top} = \mathsf{sD'}_{x,y}^{\top}\mathsf{A}_x \ \wedge\ \mathsf{rD}_{x,y}^{\top} = \mathsf{rD'}_{x,y}^{\top}\mathsf{B}_y$$

Therefore,

$$\mathsf{sD'}_{x,y}^{\top}(\mathsf{A}_x\mathsf{sE}_x\mathsf{C}) = (\mathsf{sD}_{x,y}^{\top}\mathsf{sE}_x)\mathsf{C} = (\mathsf{rD}_{x,y}^{\top}\mathsf{rE}_y)\mathsf{C} = \mathsf{rD'}_{x,y}^{\top}(\mathsf{B}_y\mathsf{rE}_y\mathsf{C})$$
$$\mathsf{rD'}_{x,y}^{\top}(\mathsf{B}_y\mathsf{kE}_y) = \mathsf{rD}_{x,y}^{\top}\mathsf{kE}_y = 1$$

To see privacy, note that for every $(x, y) \notin \mathsf{P}$, there exists $\boldsymbol{w} \in \underset{span}{col}\langle \mathsf{C} \rangle$ such that $\mathsf{sE}_x\boldsymbol{w} = \mathbf{0}_s \ \wedge\ \mathsf{rE}_y\boldsymbol{w} = \mathsf{kE}_y$. Therefore, there also exists $\boldsymbol{w'} \in \mathbb{Z}_p^{w'}$ such that $\boldsymbol{w} = \mathsf{C}\boldsymbol{w'}$. Note that,

$$\mathsf{sE}'_x\boldsymbol{w'} = (\mathsf{A}_x\mathsf{sE}_x\mathsf{C})\boldsymbol{w'} = \mathsf{A}_x\mathsf{sE}_x\boldsymbol{w} = \mathsf{A}_x\mathbf{0}_s = \mathbf{0}_{s'}$$
$$\mathsf{rE}'_y\boldsymbol{w'} = (\mathsf{B}_y\mathsf{rE}_y\mathsf{C})\boldsymbol{w'} = \mathsf{B}_y\mathsf{rE}_y\boldsymbol{w} = \mathsf{B}_y\mathsf{kE}_y = \mathsf{kE}'_y$$

so algebraic privacy is satisfied.

A.3 Proof of Theorem 4

Proof. Reconstructability can be seen by a simple check based on the reconstructability of the original encodings.
To see privacy, note that $\mathsf{P}_1(x_1, y_1) \vee \mathsf{P}_2(x_2, y_2) = 0$ implies $\mathsf{P}_1(x_1, y_1) = 0$ and $\mathsf{P}_2(x_2, y_2) = 0$ implies. Let $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ be witnesses of privacy of predicate encodings 1 and 2 respectively. It is easy to check that $\boldsymbol{w}^{\top} = \begin{pmatrix} \boldsymbol{w}_1^{\top} & \boldsymbol{w}_2^{\top} \end{pmatrix}$ is a witness of privacy of the transformed encoding.

A.4 Proof of Theorem 5

Proof. A simple check shows reconstructability. To see privacy, $\mathsf{P}_1(x_1, y_1) \wedge \mathsf{P}_2(x_2, y_2) = 0$ implies $\mathsf{P}_1(x_1, y_1) = 0$ or $\mathsf{P}_2(x_2, y_2) = 0$. If the first holds, let $\boldsymbol{w}_1$ be a witness of privacy of the first encoding. Then, $\boldsymbol{w}^\top = \begin{pmatrix} 2\boldsymbol{w}_1^\top & \mathbf{0}_{w_2}^\top & -1 \end{pmatrix}$ is a witness of the algebraic privacy of the transformed encoding. If the second holds, let $\boldsymbol{w}_2$ be a witness of privacy of the second encoding. A valid witness now is $\boldsymbol{w}^\top = \begin{pmatrix} \mathbf{0}_{w_2}^\top & 2\boldsymbol{w}_2^\top & 1 \end{pmatrix}$.

A.5 Proof of Theorem 6

Proof. It is not difficult to check reconstructability. Privacy holds because when $\mathsf{P}(x, y) = 1$, we can define $\boldsymbol{w}^\top = \begin{pmatrix} \text{-}\mathsf{sD}_{x,y}^\top & \text{-}\mathsf{sD}_{x,y}^\top \mathsf{sE}_x & \mathsf{rD}_{x,y}^\top \end{pmatrix}$ which can be checked to be a witness of the algebraic privacy of the transformed predicate encoding.

A.6 Proof of Theorem 7

Proof. A simple check is enough to verify reconstructability. For privacy, note that when $\mathsf{P}'(y, x) = 0$, we have $\mathsf{P}(x, y) = 0$. Let $\boldsymbol{w}$ be a witness of the algebraic privacy of the original encoding. Now, ${\boldsymbol{w}'}^\top = \begin{pmatrix} \text{-}\boldsymbol{w}^\top & 1 \end{pmatrix}$ is a witness of the dual predicate encoding.

A.7 Proof of Theorem 8

Proof. The proof follows directly from the following lemma and the observation that *i)* is equivalent to $\boldsymbol{h}$-hiding, while *iii)* is non-reconstructability (take $A = C$ and $B = K$).

Lemma 1. *Let $A \in \mathbb{Z}_p^{m \times n}$ and $B \in \mathbb{Z}_p^{l \times n}$ be matrices. Let $C \in \mathbb{Z}_p^{(m+l) \times n}$ be the concatenation of A and B by rows. The following three statements are equivalent:*

i) $\forall \boldsymbol{a} \in \mathbb{Z}_p^m, \forall \boldsymbol{b} \in \mathbb{Z}_p^l, \Pr_{\boldsymbol{x} \xleftarrow{\$} \mathbb{Z}_p^n} [A\boldsymbol{x} = \boldsymbol{a} \,|\, B\boldsymbol{x} = \boldsymbol{b}] = \Pr_{\boldsymbol{x} \xleftarrow{\$} \mathbb{Z}_p^n} [A\boldsymbol{x} = \boldsymbol{a}]$

ii) $\operatorname{rank}(C) = \operatorname{rank}(A) + \operatorname{rank}(B)$

iii) $\forall \boldsymbol{a} \in \mathbb{Z}_p^m, \forall \boldsymbol{b} \in \mathbb{Z}_p^l, \quad \boldsymbol{a}^\top A \neq \boldsymbol{b}^\top B \ \vee \ \boldsymbol{a}^\top A = \mathbf{0}_n^\top$

Proof (Of the Lemma).
Note that *i)* holds for every $\boldsymbol{a} \in \mathbb{Z}_p^m, \boldsymbol{b} \in \mathbb{Z}_p^l$ such that $A\boldsymbol{x} = \boldsymbol{a}$ or $B\boldsymbol{x} = \boldsymbol{b}$ have no solution. Let $\boldsymbol{a} \in \mathbb{Z}_p^m, \boldsymbol{b} \in \mathbb{Z}_p^l$ be such that the systems $A\boldsymbol{x} = \boldsymbol{a}$ and $B\boldsymbol{x} = \boldsymbol{b}$ have individually at least one solution (note that such $\boldsymbol{a}$ and $\boldsymbol{b}$ always exist). We define the sets $\Omega_A = \{\boldsymbol{x} \in \mathbb{Z}_p^n : A\boldsymbol{x} = \boldsymbol{a}\}$, $\Omega_B = \{\boldsymbol{x} \in \mathbb{Z}_p^n : B\boldsymbol{x} = \boldsymbol{b}\}$, $\Omega_{AB} = \{\boldsymbol{x} \in \mathbb{Z}_p^n : A\boldsymbol{x} = \boldsymbol{a} \ \wedge \ B\boldsymbol{x} = \boldsymbol{b}\}$. By the Rouché-Capelli Theorem, we know that

$$|\Omega_A| = p^{n - \operatorname{rank}(A)} \qquad |\Omega_B| = p^{n - \operatorname{rank}(B)} \qquad |\Omega_{AB}| = p^{n - \operatorname{rank}(C)}$$

Note that *i)* can be expressed as $\frac{|\Omega_{AB}|}{p^n} = \frac{|\Omega_A|}{p^n} \cdot \frac{|\Omega_B|}{p^n}$ which is equivalent to the equation $p^n \cdot |\Omega_{AB}| = |\Omega_A| \cdot |\Omega_B|$, and therefore,

$$p^n \cdot p^{n-\mathrm{rank}(C)} = p^{n-\mathrm{rank}(A)} \cdot p^{n-\mathrm{rank}(B)}$$

if and only if $\mathrm{rank}(C) = \mathrm{rank}(A) + \mathrm{rank}(B)$ which is *ii)*.

Now, note that $\mathrm{rank}(C) = \mathrm{rank}(A) + \mathrm{rank}(B)$ if and only if there is not a non-zero linear combination of rows of A that can be expressed as a linear combination of rows of B, which is equivalent to statement *iii)*.

A.8 Proof of Theorem 9

Proof. According to our Theorem 2, the partial encoding $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}')$ induces a predicate encoding for the predicate $\mathsf{Pred}(x, y) = 1$ iff $\exists \boldsymbol{s} \in \mathbb{Z}_p^c, r \in \mathbb{Z}_p^1$ s.t. $\boldsymbol{s}^\top \mathsf{sE}'_x = r \cdot \mathsf{rE}'_y$ and $r \cdot \mathsf{kE}'_y = 1$, or equivalently, $\exists \boldsymbol{s} \in \mathbb{Z}_p^c$ s.t. $\boldsymbol{s}^\top \mathsf{cE}_x = \mathsf{kE}_y$, which, in this case, is equivalent to the reconstructability of the tag-based encoding $(\mathsf{cE}, \mathsf{kE})$. According to Theorem 8 it is also equivalent to the predicate P.

A.9 Proof of Theorem 10

Proof. Verifying correctness of the pair encoding is a simple check. For perfect security we need to check that, when $(x, y) \notin \mathsf{P}$, the following two distributions are identical:

$$\begin{aligned} &\alpha, s_0 \xleftarrow{\$} \mathbb{Z}_p;\ r_1 \xleftarrow{\$} \mathbb{Z}_p^*;\ \boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (s_0,\ s_0 \cdot \mathsf{sE}_x \boldsymbol{h},\ r_1,\ r_1 \cdot \mathsf{rE}_y \boldsymbol{h}) \quad \equiv \\ &\quad s_0 \xleftarrow{\$} \mathbb{Z}_p;\ r_1 \xleftarrow{\$} \mathbb{Z}_p^*;\ \boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (s_0,\ s_0 \cdot \mathsf{sE}_x \boldsymbol{h},\ r_1,\ r_1 \cdot \mathsf{rE}_y \boldsymbol{h} + \alpha \cdot \mathsf{kE}_y) \end{aligned}$$

Since both distributions provide s_0 and r_1 in the clear, the above checking is equivalent to the following:

$$\begin{aligned} \boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (\mathsf{sE}_x \boldsymbol{h},\ \mathsf{rE}_y \boldsymbol{h}) \quad \equiv \\ \alpha \xleftarrow{\$} \mathbb{Z}_p;\ r_1 \xleftarrow{\$} \mathbb{Z}_p^*;\ \boldsymbol{h} \xleftarrow{\$} \mathbb{Z}_p^w;\ \mathsf{return}\ (\mathsf{sE}_x \boldsymbol{h},\ \mathsf{rE}_y \boldsymbol{h} + \alpha / r_1 \cdot \mathsf{kE}_y) \end{aligned}$$

but those distributions are identical due to the α-privacy of the predicate encoding[10].

A.10 Proof of Theorem 11

Proof. Given $(\boldsymbol{s}, \boldsymbol{r}) \in \mathbb{Z}_p^s \times \mathbb{Z}_p^r$, we define $\Omega_{\boldsymbol{s},\boldsymbol{r}} = \{\boldsymbol{w} \in \mathbb{Z}_p^w : S\boldsymbol{w} = \boldsymbol{s} \ \wedge\ R\boldsymbol{w} = \boldsymbol{r}\}$. The condition on the second bullet holds iff the cardinality of $\Omega_{\boldsymbol{s},\boldsymbol{r}}$ is p^{w-s-r}. Additionally, $|\Omega_{\boldsymbol{s},\boldsymbol{r}}|$ is independent from $\boldsymbol{r}$ and $\boldsymbol{s}$ iff the two distributions from the first bullet are identical.

[10] note that $\alpha \xleftarrow{\$} \mathbb{Z}_p$, $r_1 \xleftarrow{\$} \mathbb{Z}_p^*$ and therefore, α / r_1 distributes uniformly over $\mathbb{Z}_p$, so we can apply the α-privacy property from the predicate encoding

A.11 Proof of Theorem 12

Proof. Correctness can be easily checked. For privacy, let $(x, y) \notin \mathsf{P}$ and let $\boldsymbol{w} \in \mathbb{Z}_p^w$ be such that $\mathsf{sE}_x \boldsymbol{w} = \mathbf{0}_s$ and $\mathsf{rE}_y \boldsymbol{w} = \mathsf{kE}_y$. It is enough to check that $\boldsymbol{w}'^\top = \begin{pmatrix} \boldsymbol{w}^\top & \mathbf{0}_k^\top \end{pmatrix}$ is a witness of privacy for $(\mathsf{sE}', \mathsf{rE}', \mathsf{kE}', \mathsf{sD}', \mathsf{rD}')$.

B Flexible boolean combinations

Boolean combinations of predicate encodings can be applied dynamically. This is, we can combine them by leaving placeholders $\mathsf{P} \bowtie \mathsf{P}'$ that will be chosen during encryption or key generation. The following theorem shows how to dynamically combine two predicates making the combinator part of the secret key sk_y. Thanks to this dynamic combination, we can create predicate encodings for boolean formulas where the leaves are predicates instead of attributes. The only drawback is that the structure of the boolean formula has to be fixed.

Theorem 13 (Flexible combination of predicate encodings). *For every (s_1, r_1, w_1)-predicate encoding $(\mathsf{sE}^1, \mathsf{rE}^1, \mathsf{kE}^1, \mathsf{sD}^1, \mathsf{rD}^1)$ for $\mathsf{P}_1 : \mathcal{X}_1 \times \mathcal{Y}_1 \to \{0,1\}$ and every (s_2, r_2, w_2)-predicate encoding $(\mathsf{sE}^2, \mathsf{rE}^2, \mathsf{kE}^2, \mathsf{sD}^2, \mathsf{rD}^2)$ for $\mathsf{P}_2 : \mathcal{X}_2 \times \mathcal{Y}_2 \to \{0,1\}$, there exists a $(s_1 + s_2,\ r_1 + r_2,\ w_1 + w_2 + 1)$-predicate encoding $(\mathsf{sE}, \mathsf{rE}, \mathsf{kE}, \mathsf{sD}, \mathsf{rD})$ for the predicate $\mathsf{P} : (\mathcal{X}_1, \mathcal{X}_2) \times (\mathcal{Y}_1, \mathcal{Y}_2, \{\vee, \wedge\}) \to \{0,1\}$ such that:*

$$\mathsf{P}((x_1, x_2), (y_1, y_2, \bowtie)) \Leftrightarrow \mathsf{P}_1(x_1, y_1) \bowtie \mathsf{P}_2(x_2, y_2)$$

Concretely,

$$\mathsf{sE}_{(x_1,x_2)} = \begin{pmatrix} \mathsf{sE}^1_{x_1} & \mathbf{0}_{s_1,w_2} & \mathbf{0}_{s_1} \\ \mathbf{0}_{s_2,w_1} & \mathsf{sE}^2_{x_2} & \mathbf{0}_{s_2} \end{pmatrix} \qquad \mathsf{sD}_{(x_1,x_2),(y_1,y_2,\bowtie)} = \frac{1}{2} \begin{pmatrix} \mathsf{sD}^1_{x_1,y_1} \\ \mathsf{sD}^2_{x_2,y_2} \end{pmatrix}$$

$$\mathsf{rE}_{(y_1,y_2,\bowtie)} = \begin{pmatrix} \mathsf{rE}^1_{y_1} & \mathbf{0}_{r_1,w_2} & f_\bowtie \cdot \mathsf{kE}^1_{y_1} \\ \mathbf{0}_{r_2,w_1} & \mathsf{rE}^2_{y_2} & -f_\bowtie \cdot \mathsf{kE}^2_{y_2} \end{pmatrix} \qquad \mathsf{rD}_{(x_1,x_2),(y_1,y_2,\bowtie)} = \frac{1}{2} \begin{pmatrix} \mathsf{rD}^1_{x_1,y_1} \\ \mathsf{rD}^2_{x_2,y_2} \end{pmatrix}$$

$$\mathsf{kE}_{(y_1,y_2,\bowtie)} = \begin{pmatrix} \mathsf{kE}^1_{y_1} \\ \mathsf{kE}^2_{y_2} \end{pmatrix}$$

where $f_\bowtie$ is defined as 1 if $\bowtie = \wedge$ and 0 if $\bowtie = \vee$.

Proof. This Theorem follows straightforwardly from Theorems 4 and 5.

Note that our Theorem 7 gives us an equivalent version of the above theorem, where the placeholder is part of the ciphertext ct_x. Figure 6 presents a possible application of a flexible fixed-structure combination of boolean operators. It encodes the predicate $\mathsf{P}(x, y) = 1$ iff $x \geq y$, where $\mathcal{X} = \mathcal{Y} = \{0,1\}^4$ (4-bit strings). Note that the leaf nodes are IBE predicate encodings (one of the simplest predicate encodings).

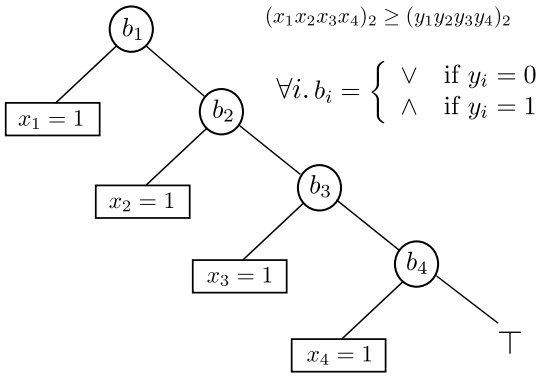

Fig. 6. Example of fixed structure for inequalities

C Concrete encodings

C.1 Predicate encoding for γ-inner product (modified from [12])

Let $n \in \mathbb{N}$ and let $\mathcal{X} = \mathbb{Z}_p^n \times \mathbb{Z}_p$, $\mathcal{Y} \in \mathbb{Z}_p^n$. The following is a $(n+1, 1, n+2)$-valid predicate encoding for the predicate $\mathsf{P}((\boldsymbol{x}, \gamma), \boldsymbol{y}) = 1$ iff $\boldsymbol{x}^\top \boldsymbol{y} = \gamma$,

$$\mathsf{sE}_{\boldsymbol{x}} = \begin{pmatrix} \boldsymbol{x} & I_n & \mathbf{0}_n \\ \gamma & \mathbf{0}_n^\top & 1 \end{pmatrix} \qquad \mathsf{rE}_{\boldsymbol{y}} = \begin{pmatrix} 0 & \boldsymbol{y}^\top & -1 \end{pmatrix} \qquad \mathsf{kE}_{\boldsymbol{y}} = \begin{pmatrix} 1 \end{pmatrix}$$

$$\mathsf{sD}_{\boldsymbol{x},\boldsymbol{y}}^\top = \begin{pmatrix} \boldsymbol{y}^\top & -1 \end{pmatrix} \qquad \mathsf{rD}_{\boldsymbol{x},\boldsymbol{y}} = \begin{pmatrix} 1 \end{pmatrix}$$

It is an *attribute-hiding predicate encoding*.

C.2 Generalized predicate encoding for broadcast encryption

We generalize the predicate encoding for broadcast encryption from [12]. Let $\mathcal{X} = \mathbb{Z}_p^n$, $\mathcal{Y} = [n] \times \mathbb{Z}_p$, we consider the predicate $\mathsf{P}(\boldsymbol{x}, (i, \gamma)) = 1$ iff $\boldsymbol{x}_i = \gamma$. As in [12], it is convenient to express the above predicate as follows:
$\mathcal{X} = (\mathbb{Z}_p^{t_2})^{t_1}$, $\mathcal{Y} = [t_1] \times [t_2] \times \mathbb{Z}_p$ and

$$\mathsf{P}((\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t_1}), (i_1, i_2, \gamma)) = 1 \text{ iff } \boldsymbol{x}_{i_1}^\top \boldsymbol{e}_{i_2} = \gamma$$

where $n = t_1 t_2$ and (i_1, i_2) is the unique pair of integers satisfying $i = (i_1\text{-}1) \cdot t_2 + i_2$ and $0 < i_2 \leq t_2$. Also, $(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{t_2})$ is the standard basis of $\mathbb{Z}_p^{t_2}$. The following is a valid $(t_1, t_2, t_1 + t_2)$-predicate encoding for the above predicate:

$$\mathsf{sE}_{\boldsymbol{x}} = \begin{pmatrix} & \boldsymbol{x}_1^\top \\ I_{t_1} & \vdots \\ & \boldsymbol{x}_{t_1}^\top \end{pmatrix} \qquad \mathsf{rE}_{(i_1,i_2,\gamma)} = \begin{pmatrix} \mathbf{0}_{t_2,(i_1\text{-}1)} & \boldsymbol{e}_{i_2} & \mathbf{0}_{t_2,(t_1-i_1)} & \gamma \cdot I_{t_2} \end{pmatrix}$$

$$\mathsf{kE}_{(i_1,i_2,\gamma)} = \boldsymbol{e}_{i_2} \qquad \mathsf{sD}_{\boldsymbol{x},(i_1,i_2,\gamma)} = \boldsymbol{e}_{i_1} \qquad \mathsf{rD}_{\boldsymbol{x},(i_1,i_2,\gamma)} = \gamma^{-1} \cdot \boldsymbol{x}_{i_1}$$

This encoding can be used to perform 2-dimensional broadcast encryption. This is, users are divided in n groups and every user i has a unique identifier γ_i. Encryption can be done in such a way that at most one user from every group will be able to decrypt.

C.3 Tag-based encoding for root sharing of polynomials

Let $m, n \in \mathbb{N}$ and let $\mathcal{X}, \mathcal{Y} \subset \mathbb{Z}_p[T]$ be the sets of polynomials of degree m and n respectively with coefficients over $\mathbb{Z}_p$. For $f(t) \in \mathcal{X}$ and $g(t) \in \mathcal{Y}$, let P be the predicate defined as $\mathsf{P}(f, g) = 1$ iff $\exists t_0 \in \mathbb{Z}_p : f(t_0) = g(t_0) = 0$. The following is a valid $(n, m, m+n)$-tag-based encoding for P,

$$\mathsf{cE}_f = \begin{pmatrix} a_m & \dots & a_0 & 0 & & \overset{n-1}{\vdots} & 0 \\ 0 & a_m & \dots & a_0 & 0 & \overset{n-2}{\vdots} & 0 \\ & & \ddots & & \ddots & & \\ 0 & \overset{n-1}{\vdots} & 0 & & a_m & \dots & a_0 \end{pmatrix} \quad \mathsf{kE}_g = \begin{pmatrix} b_n & \dots & b_0 & 0 & & \overset{m-1}{\vdots} & 0 \\ 0 & b_n & \dots & b_0 & 0 & \overset{m-2}{\vdots} & 0 \\ & & \ddots & & \ddots & & \\ 0 & \overset{m-1}{\vdots} & 0 & & b_n & \dots & b_0 \end{pmatrix}$$

where $f(t) = a_m t^m + \cdots + a_1 t + a_0$ and $g(t) = b_n t^n + \cdots + b_1 t + b_0$.