Simplifying Design and Analysis of Complex Predicate Encryption Schemes

Shashank Agrawal[1]⋆ and Melissa Chase[2]

[1] Visa Research
shaagraw@visa.com
[2] Microsoft Research
melissac@microsoft.com

Abstract. Wee (TCC'14) and Attrapadung (Eurocrypt'14) introduced predicate and pair encodings, respectively, as a simple way to construct and analyze attribute-based encryption schemes, or more generally predicate encryption. However, many schemes do not satisfy the simple information theoretic property proposed in those works, and thus require much more complicated analysis. In this paper, we propose a new simple property for pair encodings called *symbolic* security. Proofs that pair encodings satisfy this property are concise and easy to verify. We show that this property is inherently tied to the security of predicate encryption schemes by arguing that any scheme which is not trivially broken must satisfy it. Then we use this property to discuss several ways to convert between pair encodings to obtain encryption schemes with different properties like small ciphertexts or keys. Finally, we show that any pair encoding satisfying our new property can be used to construct a fully secure predicate encryption scheme. The resulting schemes are secure under a new q-type assumption which we show follows from several of the assumptions used to construct such schemes in previous work.

1 Introduction

Traditional public key encryption allows an encryptor to use a public key to encrypt a message so that the owner of the corresponding secret key can decrypt. In 2005, Sahai and Waters [35] introduced the concept of attribute-based encryption, in which who can decrypt is determined by some more complex attributes of the decryptor and the message. Of course this is only meaningful if there is some party that can determine the attributes of the decryption, thus the basic model assumes a trusted party who publishes parameters used in encryption, and who issues decryption keys to users based on their attributes; given such a key, a user should be able to decrypt any ciphertext which is compatible with his attributes. The initial result considered a simple threshold functionality: every ciphertext was encrypted with a set of attributes, and a user could decrypt if

⋆ Part of this work was done when the author was a graduate student at the University of Illinois, Urbana-Champaign, supported by NSF CNS 12-28856 and Andrew & Shana Laursen fellowship, and then an intern at Microsoft Research.

they possessed sufficiently many of those attributes. This was then generalized to key-policy ABE [22], in which the user's key specifies a policy determining what attributes must be present in the ciphertext in order for that user to be able to decrypt, and ciphertext-policy ABE [10], which is the natural opposite in that the user's key corresponds to a list of attributes and ciphertexts are encrypted with a policy which determines which attributes the user must have to decrypt.

Since then the field of ABE has grown dramatically. There has been work which extends the type of policies that can be considered, for example to non-monotone formulas [32], or even regular languages [38]. There has also been work which improves the efficiency of ABE in various dimensions, for example considering schemes with very short (e.g. constant size) ciphertexts or keys [7,41], or schemes with very short parameters (again constant-size) which still support attributes from an unbounded space [29,31,33]. There has been work on distributing the job of the authority across multiple entities [14,28], on updating ciphertexts [34], or hiding the key and/or ciphertext attributes [12,25,36,11], and many other interesting directions.[3]

One weakness in much of the early work is that the schemes presented were only shown to satisfy a weak notion of security called *selective security*. Selective security essentially only guarantees security for an adversary who chooses which type of ciphertext to attack (i.e. the attributes/policy for the ciphertext) without seeing the system parameters, any ciphertexts, or any decryption keys. Thus it was a major breakthrough when Waters introduced the dual-system encryption technique [37], paving the way for schemes which satisfied the natural definition, in which the adversary may choose what type of ciphertext to attack adaptively based on any of the other information it sees while interacting with the system. Since then there has been a lot of work focused on obtaining the results above under this more natural security definition, which is usually referred to as *full security*.

One of the main downsides of this process, however, is that while most of the original constructions were simple and intuitive, many of these new constructions are significantly more complex. Also many of the first fully secure schemes relied on composite-order pairing groups, which while conceptually simpler are not really usable in practice [23]. The effort to move these results to be based on standard prime-order pairing groups has added even more complexity [18,27,24]. As a result, the intuition for the resulting constructions is often difficult to follow, and the security analysis for these schemes is much more involved, so much so that even *verifying* the security proof is often very time consuming.

Two recent works by Wee and Attrapadung [40,2] set out to simplify the process of designing and analyzing fully secure ABE schemes. They proposed a

[3] There has also been a very interesting line of work which uses indistinguishability obfuscation or multi-linear maps to construct ABE for circuits [19,20], and a lot of progress on building ABE schemes from lattices [13,21], although achieving the natural full security notion there still requires complexity leveraging. Here, we focus on pairing based constructions as to date they provide the best efficiency and security guarantees.

simple building block, called a predicate/pair encoding, which essentially considers what happens in the exponent of a single key and a single ciphertext. They proposed an information theoretic security property, which considers the distributions of these values, again only considering a single key and ciphertext, and showed that from any pair encoding scheme which satisfies this property one can construct a fully secure ABE scheme. The initial works proposed only composite-order group schemes; later works [15,1,4] have updated these results to prime-order groups.

These results led to very simple, intuitive, and easy to analyze constructions for several basic types of ABE schemes, that worked in efficient prime order groups, and were based on simple assumptions like DLIN or SXDH. However, there are many types of ABE schemes for which we do not know how to construct this type of pair encoding. And in fact there are many types of ABE which we do not know how to construct under simple assumptions using any approach, like ABE with short ciphertexts, or with large universe, or where an attribute can be used any number of times in a policy, etc.

To address this problem, Attrapadung [2] also proposed a different security notion for pair encodings, and showed that under this notion one could construct pair encodings for many more types of ABEs, and that this notion was sufficient to produce secure constructions under more complex q-type assumptions. However, proving that a pair encoding scheme satisfies the new security notion is again a challenging task. This property involves elements in bilinear groups rather than just the exponent, and it is no longer information-theoretic, so that it must be proved via reduction to a different q-type assumption for every encoding. These reductions are very complex, and again verifying the security becomes a matter of studying several pages of proof (9 pages for predicate encryption for regular languages, for instance), providing relatively little intuition for why the scheme is secure.

1.1 Our Contributions

Our goal in this work is to simplify the process of designing and analyzing ABE schemes for those types of ABEs which we only know how to construct from q-type assumptions. Towards this, we introduce a very different kind of security property for pair encodings that completely does away with any kind of distributions, and show that it is a very powerful and natural property through a series of results. We believe it provides a new perspective for looking at the security of predicate encryption schemes.

A pair encoding scheme, as defined by Attrapadung [2], gives a way to encode the two inputs x and y to a predicate into polynomials of a simple structure. These polynomials have three types of variables: common variables shared by the encodings of x and y, and variables specific to the encoding of x and to that of y.

A new property for pair encodings. We present a new security property for pair encodings that essentially requires one to describe a mapping from the variables in the encoding to matrices and vectors. Once a mapping is specified, verifying that the property holds is just a matter of checking if the polynomials in the encoding evaluate to 0 when the variables are substituted.[4] Thus verification is much easier compared to any property known before, since they all require checking whether certain distributions are (pefectly, statistically or computationally) indistinguishable. We call our new property the *symbolic property* (Sym-Prop) since verification only involves symbolic manipulation.

We show how to convert *any* pair encoding that satisfies Sym-Prop into a *fully* secure encryption scheme whose security is based on a *fixed* q-type assumption that we call q-ratio. We use the generic transformation from Agrawal and Chase [1], henceforth called Gen-Trans, for this purpose. Gen-Trans takes an encoding scheme satisfying a certain information-theoretic property and produces an encryption scheme in dual system groups [16], which can then be instantiated in composite-order groups under subgroup decision assumptions or prime-order groups under the k-linear assumption.

We show that the security of Gen-Trans can also be argued when the pair encoding satisfies a very different security property, the symbolic property. The main novelty in our proof, and the crucial difference from AC16, is in how the form of master secret key is changed: while AC16 uses an information-theoretic property, we use Sym-Prop in conjunction with a new assumption called $\mathsf{q\text{-}ratio}_{\mathsf{dsg}}$ on dual system groups. [5] At a very high level, the terms that cannot be generated from $\mathsf{q\text{-}ratio}_{\mathsf{dsg}}$ are exactly the ones that go to zero due to Sym-Prop. Thus we are able to embed $\mathsf{q\text{-}ratio}_{\mathsf{dsg}}$ successfully into the reduction. Interestingly, however, as we will discuss below, Sym-Prop is not just an artifact of our proof strategy but seems to be inherently linked to the fundamental security of the resulting predicate encryption schemes.

An added advantage of borrowing AC16's transformation is that when a pair encoding is *used* in a way that can be shown to be information-theoretically secure, then the encryption scheme obtained through Gen-Trans is fully secure under a standard assumption. We show a useful application of this feature below.

We also show that the q-ratio assumption is in fact implied by several other q-type assumptions used to construct ABE schemes, in particular those used in the Lewko-Waters ABE [30] and Attrapadung's fully secure predicate encryption for regular languages [2]. This assumption is also simpler to describe than either [30] or [2] and we believe that this approach better captures the intuition for why these schemes are secure.

Analysis of pair encodings. We show that Sym-Prop holds for several pair encoding schemes, both new and old: multi-use CP-ABE, short ciphertext CP-ABE, large universe KP-ABE, short ciphertext KP-ABE, and predicate encryption for regular languages.

[4] The trivial case is ruled out because we also require that the vectors corresponding to two special variables, in the encoding of x and y respectively, are not orthogonal.

[5] $\mathsf{q\text{-}ratio}_{\mathsf{dsg}}$ is very similar to q-ratio. We show that Chen and Wee's instantiations of dual system groups satisfy $\mathsf{q\text{-}ratio}_{\mathsf{dsg}}$ if the underlying bilinear maps satisfy q-ratio.

First, we present a new pair encoding $\Pi_{\mathsf{re\text{-}use}}$ for CP-ABE that allows an attribute to be used *any* number of times in a policy. An interesting feature of $\Pi_{\mathsf{re\text{-}use}}$ is that if no attribute is used more than once, then it collapses to the one-use scheme of [2], which is information-theoretically secure. So if we get an encryption scheme ES when Gen-Trans is applied on $\Pi_{\mathsf{re\text{-}use}}$, then ES is fully secure under a *standard* assumption as long as it is used to encrypt policies where attributes are not repeated. If a policy with multiple use of attributes needs to be encrypted, then ES still fully hides the payload but under a q-type assumption. As far as we know, no multi-use scheme with this feature was known before. For instance, the Lewko-Waters' scheme [30] uses an assumption whose size scales with that of the access policy in the challenge ciphertext. So even if no attribute is used more than once, security still relies on a q-type assumption. [6]

For short ciphertext CP-ABE, we show that the pair encoding of Agrawal and Chase [1] satisfies Sym-Prop. This means that the encryption scheme that comes out after applying Gen-Trans is fully secure, not just selectively secure as they proved it (since we use the same transformation as them), under a q-type assumption. Note that it was not known earlier whether there exists a fully-secure CP-ABE scheme with constant-size ciphertexts under any kind of assumption on bi-linear maps. In fact, we can *generically* build an encryption scheme with constant-size ciphertexts for any predicate P from *any* pair encoding for P that satisfies Sym-Prop as discussed in more detail below.

The last three encodings we analyze are borrowed from the work of Attrapadung [2] with slight simplification. Previously, we only knew how to analyze them using the much more complex computational security property in [2]. Our analysis of these schemes is considerably simpler: for comparison, the proof of computational security for the regular languages pair encoding required 9 full pages, while our proof of symbolic security only takes 2.5 llncs pages. Our proofs can be seen as extracting, abstracting and somewhat simplifying the key ideas behind Attrapadung's security analysis, so that they can be very easily verified, and more easily applied to future schemes.

Symbolic property inherent in a secure scheme. While there are several security properties for encoding schemes that allow one to check if they can be used to build some type of encryption scheme, is there a property that an encoding scheme should *not* satisfy? A natural one that comes to mind is that correctness holds for an x and y that make a predicate *false*. In other words, there exists a way to combine the polynomials in the encoding to recover the blinding factor for the message even when the predicate is false. We call a pair encoding scheme that satisfies this property *trivially broken*.

Building an encryption scheme from a pair encoding scheme seems to require at least that the pair encoding *not* be trivially broken, but there is no general

[6] There are other ABE schemes that get much more than attribute re-use, like large universe or short keys, based on q-type assumptions [2], but proving them secure under a standard assumption when re-use does not happen would be even more difficult.

result that shows some type of security for a scheme that only provides such a minimal guarantee. In Section 4, we give the first result of this kind: *Any pair encoding scheme that is not trivially broken satisfies our symbolic property.*

This result has several interesting broad implications. Suppose we have an encoding Π that we do not know to be secure. We apply Gen-Trans on it to get an encryption scheme ES. For this scheme to not be completely broken, there should not be a way to trivially combine some ciphertext and key to recover the message when the predicate is false. Now an interesting fact about our generic transformation Gen-Trans is that it preserves the structure of pair encodings, so that if there is way to combine the polynomials to recover the blinding factor, then the ciphertext and key coming out of Gen-Trans can be combined to recover the message. Therefore, if ES is not completely broken, Π is not broken either. This further implies that Π satisfies Sym-Prop and ES is fully secure under q-ratio. Thus we arrive at a very interesting conclusion: *Either* ES *is broken in an obvious way or it is fully secure under* q-ratio. Hence, Sym-Prop seems to be inherently linked to the fundamental security of encryption schemes, and is not just an artifact of our proof strategy.

We can take this line of argument even further. Suppose there is a generic transformation that preserves the structure of pair encodings in the sense described above. And suppose that when an encoding scheme satisfying a certain property X is given as input, it generates an encryption scheme that is not obviously broken, for example a *selectively* secure scheme. Then every encoding that satisfies X will also satisfy our symbolic property, and hence will lead to a *fully* secure encryption scheme through Gen-Trans! In this paper, we do not formalize the exact requirements a generic transformation should satisfy for such a general result to hold, leaving it as an interesting exercise for future work.

We conclude with an alternate way of proving symbolic security in case finding a mapping from an encoding's variables to matrices/vectors seems difficult: show that for all x and y for which the predicate is false, the blinding factor cannot be recovered from the encoding's polynomials.

New generic conversions. Thanks to the simplicity of our new symbolic property, we are able to show several useful transformations of pair encodings that preserve security. Specifically,

1. *Dual conversion.* Any secure pair encoding for a predicate can be transformed into a secure encoding scheme for the dual predicate (where the role of key and ciphertext are switched).
2. *Compact ciphertexts.* Any secure pair encoding can be converted into one that has a constant number of variables and polynomials in the ciphertext encoding. Thus, after applying Gen-Trans to the latter encoding, one gets encryption schemes with constant-size ciphertexts.

3. *Compact keys.* Analogous to above, any secure pair encoding can be converted into one that has a constant number of variables and polynomials in the key encoding, leading to encryption schemes with constant-size keys. [7]

This demonstrates the power and versatility of the new symbolic property. In contrast, only the first type of transformation is known for the security properties of Attrapadung [2,8], and none is known for Wee [40] or Chen et al. [15].

More new schemes. Apart from the new scheme for unbounded attribute-reuse and showing that the constant-size ciphertext CP-ABE of [1] is fully secure, our generic conversions for pair encodings help us arrive at schemes that were not known before:

– As mentioned before, we show that the regular language pair encoding from [2] satisfies our symbolic property. Here keys are associated with regular languages, expressed as deterministic finite automata (DFA), and ciphertexts are associated with strings of any length from an alphabet set. One can first apply the dual conversion transformation to get an encoding scheme where ciphertexts and keys are associated with DFAs and strings, respectively. Then applying our compact ciphertext transformation to this encoding, and using the resulting pair encoding in Gen-Trans, one gets an encryption scheme for regular languages with constant sized ciphertexts (but with an upper bound on the size of DFAs).
– Similarly, applying our compact ciphertext/key transformation to Attrapadung's pair encodings for doubly spatial encryption (DSE) yields new encoding schemes, that then lead to encryption schemes with constant size ciphertext and keys, respectively. The only previous work on short ciphertext DSE [5] relied on a more complex series of transformations in which one type of predicate family (e.g. CP-ABE) is embedded inside another (e.g. DSE), and resulted in more expensive encodings.

1.2 Overview of Symbolic Security

This section provides a high-level *informal* treatment of pair encodings and the symbolic property with the goal of building some intuition about these concepts. Please refer to Section 3 for a formal presentation.

Pair encodings. The pair encoding framework focuses on the exponent space of an encryption scheme. Suppose there is a predicate P that takes two inputs x and y. We want to encode x into a ciphertext and y into a key. An encryption scheme for P generally has terms like $g^{b_1}, g^{b_2}, \dots$ and a special one of the form $e(g,g)^{\alpha}$ in the public parameters (b_1, b_2, ... and α are chosen randomly). α plays the role of the master secret key. To encrypt a message m along with attribute x, some random numbers $s_0, s_1, s_2, \dots$ are chosen and new terms are created by

[7] This transformation and the one above requires some bound on the number of variables and polynomials in the respective encoding.

raising g, or some *common* term like g^{b_j}, to some s_i, and then taking a linear combination of these terms, where the terms and combination used depend on x. So, if we look at the exponent of any group element output by the encryption algorithm, it is usually a polynomial of the form $s_1 + \lambda_1 s_2 b_3 + \ldots$ where λ_1 is a constant that depends on x. Finally, m is hidden inside the ciphertext by blinding it with a re-randomization of $e(g,g)^{\alpha}$, say $e(g,g)^{\alpha s_0}$.

Similarly, the exponents of group elements in any key are of the form $r_1 + \mu r_2 b_1 + \ldots$, where $r_1, r_2, \ldots$ is fresh randomness chosen for this key. We could also have expressions that contain α because key generation involves the master secret key. Thus there are three different types of variables involved in a pair encoding: the common variables $b_1, b_2, \ldots$, the ciphertext encoding variables $s_0, s_1, s_2, \ldots$, and the key encoding variables $\alpha, r_1, r_2, \ldots$.

Overall, it can be seen that if we focus on the exponent space of an encryption scheme, we need to deal with polynomials of a special form only. If $P(x,y) = 1$, then it should be possible to combine the ciphertext and key polynomials so that αs_0 can be recovered, and then used to unblind the message. The pair encoding framework just abstracts out such similarities between predicate encryption schemes in a formal way.

Security properties and transformation. Many security properties have been proposed in the literature for pair encodings, and a more restricted structure called predicate encodings [40,2,15,1]. The main contribution of these papers is to give a *generic* transformation from *any* pair encoding that satisfies their respective property into a fully secure predicate encryption scheme in composite or prime order groups (or a higher level abstraction called dual-system groups [16]). Proving that a pair encoding scheme satisfies a certain property is *significantly* easier, especially if the property is information-theoretic, than directly proving security of an encryption scheme. This is not surprising because there are no bi-linear maps, hardness assumptions, or sophisticated dual-encryption techniques involved in this process. Furthermore, verifying security of any number of encryption schemes designed through the pair encoding framework reduces to checking that the respective pair encodings are secure—a much easier task—and that the generic transformation is correct—a one-time effort. Needless to say, this saves a huge amount of work.

A concrete example: Unbounded attribute re-use. Suppose we want to design an ABE scheme that puts *no* restriction on the number of times an attribute can be used in an access policy. We know that a linear secret sharing scheme is the standard way to present a policy. It consists of a matrix $\mathbf{A}$ of size $m \times k$ and a mapping π from its rows to the universe of attributes. A value γ can be secret-shared through $\mathbf{A}$ by creating m shares, one for each row. If a user has a set of attributes S, then she gets shares for all the rows that map to some attribute in S through π. If S satisfies $(\mathbf{A}, \pi)$, then those shares can be combined to recover γ; otherwise, γ is information-theoretically hidden. In nearly all fully secure ABE schemes, the mapping π is assumed to be injective or one-to-one (this is called the one-use restriction), but we want to build an

ABE scheme that supports any π whatsoever. In particular, the size of public parameters should not affect how many times an attribute can be used in a policy. (Any such scheme will likely rely on a q-type assumption [30].[8])

For a row i of $\mathbf{A}$, suppose $\rho(i)$ denotes which occurrence of $\pi(i)$ this is. (If an attribute y is attached to the second and fifth rows, then $\rho(2) = 1$ and $\rho(5) = 2$.) We now present a new pair encoding $\Pi_{\mathsf{re\text{-}use}}$ for unbounded re-use by adapting the one-use scheme of [2]. (Some minor elements of the encoding have been suppressed for simplicity; see the full version for a full description.)

$$\mathsf{EncCt}((\mathbf{A}, \pi)) \to s_0, s_1, \ldots, s_d, \quad \{\mathbf{a}_i(s_0 b', \hat{s}_2, \ldots, \hat{s}_k)^{\mathsf{T}} + s_{\rho(i)} b_{\pi(i)}\}_{i=1,\ldots,m}$$
$$\mathsf{EncKey}(S) \to r, \quad \alpha + rb', \quad \{rb_y\}_{y \in S}$$

Here $\mathbf{a}_i$ is the ith row of $\mathbf{A}$ and d is the maximum number of times any attribute appears in it. A nice feature of $\Pi_{\mathsf{re\text{-}use}}$ is that if no attribute is used more than once (i.e. $d = 1$), then the scheme collapses to that of [2], and one can show that α is information-theoretically hidden, or that $\Pi_{\mathsf{re\text{-}use}}$ is *perfectly* secure.

If attributes are used multiple times, so that the ciphertext encoding has several variables $s_1, \ldots, s_d$, then α might be revealed to an unbounded adversary. Thus we need to find out if $\Pi_{\mathsf{re\text{-}use}}$ satisfies a different type of property for which a generic transformation is known. One possibility is the computational *double selective master-key hiding* property due to Attrapadung, but then the advantages of an abstraction like pair encoding are more or less lost: we will have to work at the level of bi-linear maps instead of simple polynomials, and find a suitable q-type assumption(s) under which the property can be shown to hold.

The symbolic property. Our new symbolic property (Sym-Prop) can be very useful in such cases. It provides a new, clean way of reasoning about security of pair encodings: instead of arguing that one distribution is indistinguishable from another, whether information-theoretically or computationally, one needs to discover a mapping from the variables involved in an encoding to matrices and vectors, such that when the latter is substituted for the former in any ciphertext/key encoding polynomial, the zero vector is obtained. Indeed, one needs to invest some effort in order to find the right matrices and vectors that will make the polynomials go to zero, but once such a discovery is made, verifying the property is just a matter of doing some simple linear algebra.

Recall that a pair encoding scheme for a predicate P that takes two inputs x and y, consists of three different types of variables: common variables $b_1, b_2, \ldots$, ciphertext encoding variables $s_0, s_1, s_2, \ldots$, and key encoding variables

[8] In a recent work, Kowalczyk and Lewko [26] proposed a new technique to boost the entropy of a small set of (unpublished) semi-functional parameters. Using this idea, they propose a new KP-ABE scheme where the number of group elements in the public parameters grows only logarithmically in the bound on the number of attribute-uses in a policy, but note that the number of times an attribute can be reused is still affected. Furthermore, the size of ciphertexts scales with the maximum number of times an attribute can be re-used.

$\alpha, r_1, r_2, \ldots$. Sym-Prop is defined w.r.t. three (deterministic) algorithms, EncB, EncS and EncR. Among them, EncB generates matrices for the common variables; EncS and EncR generate vectors for ciphertext encoding and key encoding variables, respectively. The inputs to these three algorithms depend on what type of symbolic property we want to prove. For the selective version, the three algorithms get x as input, while EncR also gets y; and for the co-selective version, they all get y as input, while EncS also gets x. This is in line with the selective and co-selective security notions for encryption schemes. In the former, all key queries come after the challenge ciphertext, while in the latter, they come beforehand. A pair encoding scheme satisfies Sym-Prop if it satisfies both the selective and co-selective variants.

The trivial case where all the matrices and vectors output by the three algorithms are simply zero is ruled out because we also require that the vectors corresponding to two special variables, s_0 in the encoding of x and α in the encoding of y, are not orthogonal.

Proving the symbolic property for $\Pi_{\text{re-use}}$. To prove Sym-Prop for the multi-use encoding scheme $\Pi_{\text{re-use}}$ defined above, we need to define the outputs of the three algorithms EncB, EncS and EncR (in other words, a mapping from the variables in $\Pi_{\text{re-use}}$ to vectors and matrices) in both the selective and co-selective settings. Towards this, we make use of a simple combinatorial fact that is often used in arguing security of ABE schemes. If a set of attributes S does not satisfy an access policy $(\mathbf{A}, \pi)$, then there exists a vector $\mathbf{w} = (w_1, \ldots, w_k)$ s.t. $w_1 = 1$ and $\mathbf{a}_i$ is orthogonal to $\mathbf{w}$ for all i such that $\pi(i) \in S$. Note that $\mathbf{w}$ can be computed only by an algorithm that knows both $(\mathbf{A}, \pi)$ and S.

We also need some simple notation to describe the mapping. Let $\mathbf{E}_{i,j}$ be an $k \times d$ matrix with 1 at the (i,j)-th position and 0 everywhere else. Also, let $\mathbf{e}_i$ be the ith d-length unit vector and $\overline{\mathbf{e}}_j$ be the jth k-length unit vector. Here is the mapping for the selective version:

$$b_y : -\sum_{\ell=1}^{d}\sum_{j=1}^{k} a_{\sigma(y,\ell),j}\mathbf{E}_{j,\ell}, \qquad b' : \mathbf{E}_{1,1},$$

$$s_0 : \mathbf{e}_1, \qquad s_\ell : \mathbf{e}_\ell, \qquad \hat{s}_j : \overline{\mathbf{e}}_j, \qquad \alpha : \mathbf{e}_1, \qquad r : -\sum_{j=1}^{k} w_j \overline{\mathbf{e}}_j,$$

where $\sigma(y,\ell)$ is the index of the row in $\mathbf{A}$ which has the ℓ-th occurrence of y. Further, if $\mathbf{E}_{i,j}$, $\mathbf{e}_i$ and $\overline{\mathbf{e}}_j$ carry the meaning as above, except that their dimensions are $1 \times T$, T and 1 respectively, then the mapping for the co-selective version is:

$$b_y : \mathbf{0} \text{ for } y \in S \text{ and } -\mathbf{E}_{1,y} \text{ otherwise}, \qquad b' : \mathbf{E}_{1,1},$$

$$s_0 : w_1\mathbf{e}_1, \qquad s_\ell : \sum_{i:\rho(i)=\ell} \mathbf{a}_i \mathbf{w}^{\mathsf{T}} \mathbf{e}_{\pi(i)}, \qquad \hat{s}_j : w_j \overline{\mathbf{e}}_1, \qquad \alpha : \mathbf{e}_1, \qquad r : -\overline{\mathbf{e}}_1.$$

We encourage the reader to verify that the polynomials in $\Pi_{\text{re-use}}$ (except the simples ones $s_0, s_1, \ldots, s_d, r$) go to zero when the two mappings described above are applied. (Vectors output by EncS (resp. EncR) are multiplied to the right (resp. left) of matrices output by EncB.) All it takes are simple observations like $\mathbf{E}_{i,j} \cdot \mathbf{e}_{j'}^{\mathsf{T}}$ gives a non-zero vector if and only if $j = j'$, and that $\mathbf{w}$ is orthogonal to every row in $\mathbf{A}$ that maps to an attribute in S. (See the full version for a formal proof.) One can consider the two mappings to be a short *certificate* of the security of $\Pi_{\mathsf{re\text{-}use}}$.

How to find a mapping? Indeed, as pointed out earlier, finding an appropriate mapping is not a trivial task. Nevertheless, Sym-Prop is still the *right* property for arguing security of pair encodings for the following reasons:

– If finding the right mapping is difficult for Sym-Prop, then finding a proof for the computational property of Attrapadung [2] is several times more difficult. A typical proof of the symbolic property is 1-2 pages while computational property proofs could go up to 10 pages (see the encoding for regular languages, for instance). A central issue with computational properties is finding an appropriate q-type assumption under which it holds, which may be very difficult for a complex predicate. Our approach can be seen as extracting out the *real* challenging part of designing Attrapadung's computational proofs.
– Verification of Sym-Prop involves doing simple linear algebra, arguably a much simpler task than checking indistinguishability of distributions, and certainly a much simpler task than verifying a long computational reduction.
– The *certificate* for the symbolic security of $\Pi_{\mathsf{re\text{-}use}}$ bears many similarities with those of other encodings that we will describe later in the paper. Thus proving Sym-Prop for a new encoding scheme is not as difficult as it might seem at first. Furthermore, modifying a short proof of the symbolic property is much easier than a long proof of a computational property.
– Recall our result that if an encoding scheme is not trivially broken then it satisfies Sym-Prop. This gives an alternate way of showing that Sym-Prop holds, by proving that the scheme is not broken.

1.3 Outline of The Paper

In Section 2 we define relevant notation and review the standard definition of predicate encryption. In Section 3 we define pair encoding schemes and our new symbolic property formally. Section 5 first reviews the notion of dual system groups, then shows how to build encryption schemes from any pair encoding by using them. This conversion is a two-step process: first we *augment* an encoding so that it satisfies a few extra properties (Section 5.1); next we apply the transformation from Agrawal and Chase [1] (Section 5.4). A proof of security of the resulting encryption scheme is provided in Section 7.

Section 6 gives generic transformations that can be used to reduce the number of variables and/or polynomials in an encoding, which can then be used to get encryption schemes with constant-size ciphertexts/keys. We also provide a transformation from any encoding for a predicate to an encoding for the dual predicate. However, due to space constraints, most of the details are available in the full version only. The full version also provides several examples to illustrate how symbolic property can substantially simplifying the analysis of encoding schemes.

2 Preliminaries

We use λ to denote the security parameter. A negligible function is denoted by negl. We use bold letters to denote matrices and vectors, with the former in uppercase and the latter in lowercase. The operator $\cdot$ applied to two vectors computes their entry-wise product and $\langle , \rangle$ gives the inner-product. For a vector $\mathbf{u}$, we use u_i to denote its ith element, and for a matrix $\mathbf{M}$, $M_{i,j}$ denotes the element in the ith row and jth column. When we write $g^{\mathbf{u}}$ for a vector $u = (u_1, \ldots, u_n)$, we mean the vector $(g^{u_1}, \ldots, g^{u_n})$. $g^{\mathbf{M}}$ for a matrix $\mathbf{M}$ should be interpreted in a similar way. The default interpretation of a vector should be as a row vector.

For two matrices $\mathbf{U}$ and $\mathbf{V}$ of dimension $n \times m_1$ and $n \times m_2$ respectively, let $\mathbf{U} \circ \mathbf{V}$ denote the column-wise *join* of $\mathbf{U}$ and $\mathbf{V}$ of dimension $n \times (m_1 + m_2)$, i.e., $\mathbf{U} \circ \mathbf{V}$ has the matrix $\mathbf{U}$ as the first m_1 columns and $\mathbf{V}$ as the remaining m_2 columns. We also refer to this operation as *appending* $\mathbf{V}$ to $\mathbf{U}$. (The notation easily extends to vectors because we represent them as row matrices.) If we want to join matrices row-wise instead, we could take their transpose, apply a column-wise join, and then take the transpose of the resultant matrix.

We use $x \leftarrow_R S$, for a set S, to denote that x has been drawn uniformly at random from it. The set of integers $a, a+1, \ldots, b$ is compactly represented as $[a, b]$. If $a = 1$, then we just use $[b]$, and if $a = 0$, then $[b]^+$.

Let $\mathbb{Z}_N$ denote the set of integers $\{0, 1, 2, \ldots, N\}$. Let $\mathcal{G}_N(m)$ denote the set of all vectors of length m with every element in $\mathbb{Z}_N$. Similarly, let $\mathcal{G}_N(m_1, m_2)$ denote the set of all matrices of size $m_1 \times m_2$ that have all the elements in $\mathbb{Z}_N$.

Bilinear Pairings: We use the standard definition of pairing friendly groups from literature. A mapping e from a pair of groups $(\mathcal{G}, \mathcal{H})$ to a target group $\mathcal{G}_T$ is bilinear if there is linearity in both the first and second inputs, i.e. $e(g^a, h^b) = e(g, h)^{ab}$ for every $g \in \mathcal{G}, h \in \mathcal{H}$ and $a, b \in \mathbb{Z}$. We require e to be non-degenerate and efficiently computable. The identity element of a group G is denoted by 1_G.

Let $\mathsf{GroupGen}$ be an algorithm that on input the security parameter λ outputs $(N, \mathcal{G}, \mathcal{H}, \mathcal{G}_T, g, h, e)$ where $N = \Theta(\lambda)$; $\mathcal{G}$, $\mathcal{H}$ and $\mathcal{G}_T$ are (multiplicative) cyclic groups of order N; g, h are generators of $\mathcal{G}$, $\mathcal{H}$, respectively; and $e : \mathcal{G} \times \mathcal{H} \to \mathcal{G}_T$ is a bilinear map. In this paper our focus will be on prime-order groups because they perform much better in practice.

Predicate family. We borrow the notation of predicate family from Attrapadung [2]. It is given by $P = \{P_\kappa\}_{\kappa \in \mathbb{N}^c}$ for some constant c, where P_κ maps an $x \in \mathcal{X}_\kappa$ and a $y \in \mathcal{Y}_\kappa$ to either 0 or 1. The first entry of κ is a number $N \in \mathbb{N}$ that is supposed to specify the size of a domain; rest of the entries are collectively referred to as par, i.e. $\kappa = (N, \mathsf{par})$.

2.1 Predicate Encryption

An encryption scheme for a predicate family $P = \{P_\kappa\}_{\kappa \in \mathbb{N}^c}$ over a message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$ consists of a tuple of four PPT algorithms (Setup, $\mathsf{Encrypt}$, KeyGen, $\mathsf{Decrypt}$) that satisfy a correctness condition. These algorithms behave as follows.

- $\mathsf{Setup}(1^\lambda, \mathsf{par})$. On input 1^λ and par, Setup outputs a master public key $\textsc{mpk}$ and a master secret key $\textsc{msk}$. The output of Setup is assumed to also define a natural number N, and κ is set to (N, par).
- $\mathsf{Encrypt}(\textsc{mpk}, x, m)$. On input $\textsc{mpk}$, $x \in \mathcal{X}_\kappa$ and $m \in \mathcal{M}_\lambda$, $\mathsf{Encrypt}$ outputs a ciphertext $\textsc{ct}$.
- $\mathsf{KeyGen}(\textsc{msk}, y)$. On input $\textsc{msk}$ and $y \in \mathcal{Y}_\kappa$, KeyGen outputs a secret key $\textsc{sk}$.
- $\mathsf{Decrypt}(\textsc{mpk}, \textsc{sk}, \textsc{ct})$. On input $\textsc{mpk}$, a secret key $\textsc{sk}$ and a ciphertext $\textsc{ct}$, $\mathsf{Decrypt}$ outputs a message $m' \in \mathcal{M}_\lambda$ or $\perp$.

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

$$\begin{aligned}&\Pr[(\textsc{mpk}, \textsc{msk}) \leftarrow \mathsf{Setup}(1^\lambda);\\ &\qquad \mathsf{Decrypt}(\textsc{mpk}, \mathsf{KeyGen}(\textsc{msk}, y), \mathsf{Encrypt}(\textsc{mpk}, x)) \neq P_\kappa(x, y)] \leq \mathsf{negl}(\lambda),\end{aligned}$$

where the probability is over the random coin tosses of Setup, $\mathsf{Encrypt}$ and KeyGen ($\mathsf{Decrypt}$ can be assumed to be deterministic without loss of generality).

Security: Consider the following game $\mathsf{IND\text{-}CPA}^b_{\mathcal{A}}(\lambda, \mathsf{par})$ between a challenger Chal and an adversary $\mathcal{A}$ for $b \in \{0, 1\}$ when both are given inputs 1^λ and par:

1. *Setup Phase*: Chal runs $\mathsf{Setup}(1^\lambda, \mathsf{par})$ to obtain $\textsc{mpk}$ and $\textsc{msk}$. It gives $\textsc{mpk}$ to $\mathcal{A}$.
2. *Query Phase*: $\mathcal{A}$ requests a key by sending $y \in \mathcal{Y}_\kappa$ to Chal, and obtains $\textsc{sk} \leftarrow \mathsf{KeyGen}(\textsc{msk}, y)$ in response. This step can be repeated any number of times.
3. *Challenge Phase*: $\mathcal{A}$ sends two messages $m_0, m_1 \in \mathcal{M}_\lambda$ and an $x^\star \in \mathcal{X}_\kappa$ to Chal, and gets $\textsc{ct} \leftarrow \mathsf{Encrypt}(\textsc{mpk}, x, m_b)$ as the challenge ciphertext.
4. *Query Phase*: This is identical to step 2.
5. *Output.* $\mathcal{A}$ outputs a bit.

The output of the experiment is the bit that $\mathcal{A}$ outputs at the end. It is required that for all y queried in steps 2 and 4, $P_\kappa(x^\star, y) = 0$.

Definition 2.1. An encryption scheme is *adaptively* or *fully* secure if for all par and PPT adversary $\mathcal{A}$,

$$|\Pr[\mathsf{IND\text{-}CPA}^0_{\mathcal{A}}(\lambda, \mathsf{par}) = 1] - \Pr[\mathsf{IND\text{-}CPA}^1_{\mathcal{A}}(\lambda, \mathsf{par}) = 1]| \leq \mathsf{negl}(\lambda), \qquad (1)$$

where the probabilities are taken over the coin tosses of $\mathcal{A}$ and Chal. It is *semi-adaptively* secure if (1) is satisfied with respect to a modified version of $\mathsf{IND\text{-}CPA}$ where the second step is omitted [17,39]. Further, it is *co-selectively* secure if (1) holds when the fourth step is removed from the $\mathsf{IND\text{-}CPA}$ game [6].

3 Pair encoding schemes

The notion of pair encoding schemes (PES) was introduced by Attrapadung [2], and later refined independently by Agrawal and Chase [1] and Attrapadung [4] himself in an identical way. As observed in the latter works, *all* pair encodings proposed originally in [2] satisfy the additional constraints in the refined versions.

We present here a more structured definition of pair encoding schemes so that the reader can easily see the different components involved. In the full version we describe the original formulation as well, and argue why our definition does not lose any generality.

3.1 Definition

A PES for a predicate family $P_\kappa : \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ indexed by $\kappa = (N, \mathsf{par})$, where par specifies some parameters, is given by four *deterministic* polynomial-time algorithms as described below.

– $\mathsf{Param}(\mathsf{par}) \to n$. When given par as input, Param outputs $n \in \mathbb{N}$ that specifies the number of *common* variables, which we denote by $\mathbf{b} := (b_1, \ldots, b_n)$.

– $\mathsf{EncCt}(x, N) \to (w_1, w_2, \mathbf{c}(\mathbf{s}, \hat{\mathbf{s}}, \mathbf{b}))$. On input $N \in \mathbb{N}$ and $x \in \mathcal{X}_{(N,\mathsf{par})}$, EncCt outputs a vector of polynomials $\mathbf{c} = (c_1, \ldots, c_{w_3})$ in *non-lone* variables $\mathbf{s} = (s_0, s_1, \ldots, s_{w_1})$ and *lone* variables $\hat{\mathbf{s}} = (\hat{s}_1, \ldots, \hat{s}_{w_2})$. (The variables $\hat{s}_1, \ldots, \hat{s}_{w_2}$ never appear in the form $\hat{s}_z b_j$, and are hence called lone.) For $\ell \in [w_3]$, where $\eta_{\ell,z}, \eta_{\ell,i,j} \in \mathbb{Z}_N$, the ℓth polynomial is given by

$$\sum_{z \in [w_2]} \eta_{\ell,z} \hat{s}_z \quad + \sum_{\substack{i \in [w_1]^+, \\ j \in [n]}} \eta_{\ell,i,j} s_i b_j .$$

– $\mathsf{EncKey}(y, N) \to (m_1, m_2, \mathbf{k}(\mathbf{r}, \hat{\mathbf{r}}, \mathbf{b}))$. On input $N \in \mathbb{N}$ and $y \in \mathcal{Y}_{(N,\mathsf{par})}$, EncKey outputs a vector of polynomials $\mathbf{k} = (k_1, \ldots, k_{m_3})$ in non-lone variables $\mathbf{r} = (r_1, \ldots, r_{m_1})$ and lone variables $\hat{\mathbf{r}} = (\alpha, \hat{r}_1, \ldots, \hat{r}_{m_2})$. For $t \in [m_3]$, where $\phi_t, \phi_{t,z'}, \phi_{t,i',j} \in \mathbb{Z}_N$ the tth polynomial is given by

$$\phi_t \alpha \quad + \sum_{z' \in [m_2]} \phi_{t,z'} \hat{r}_{z'} \quad + \sum_{\substack{i' \in [m_1], \\ j \in [n]}} \phi_{t,i',j} r_{i'} b_j .$$

– $\mathsf{Pair}(x, y, N) \to (\mathbf{E}, \overline{\mathbf{E}})$. On input N, and both x and y, Pair outputs two matrices $\mathbf{E}$ and $\overline{\mathbf{E}}$ of size $(w_1 + 1) \times m_3$ and $w_3 \times m_1$, respectively.

Observe that the output of EncKey is analogous to that of EncCt, except in how the special variables α and s_0 are treated in the respective case. While α is lone variable, i.e. it never appears in conjunction with a common variable, s_0 is not. See the full version for several concrete examples of pair encodings and the different types of variables involved.

Correctness. A PES is correct if for every $\kappa = (N, \mathsf{par})$, $x \in \mathcal{X}_\kappa$ and $y \in \mathcal{Y}_\kappa$ such that $P_\kappa(x, y) = 1$, the following holds symbolically

$$\mathbf{s}\mathbf{E}\mathbf{k}^\mathsf{T} + \mathbf{c}\overline{\mathbf{E}}\mathbf{r}^\mathsf{T} = \sum_{\substack{i \in [w_1]^+, \\ t \in [m_3]}} s_i E_{i,t} k_t + \sum_{\substack{\ell \in [w_3], \\ i' \in [m_1]}} c_\ell \overline{E}_{\ell,i'} r_{i'} = \alpha s_0.$$

The matrix $\mathbf{E}$ takes a linear combination of the products of non-lone variables output by EncCt and polynomials output by EncKey. (Its rows are numbered from 0 to w_1.) Analogously, $\overline{\mathbf{E}}$ takes a linear combination of the products of polynomials output by EncCt and non-lone variables output by EncKey. Below we use $\mathsf{ct\text{-}enc}$ and $\mathsf{key\text{-}enc}$ as a shorthand for polynomials and variables output by EncCt (ciphertext-encoding) and EncKey (key-encoding), respectively.

3.2 Symbolic Property

We introduce a new symbolic property for pair encoding schemes that significantly simplifies their analysis for even complex predicates. We get the best of two worlds: not only is our symbolic property very clean to describe (like information-theoretic properties), it can also capture all the predicates that have been previously captured by any computational property. Further, the property does not involve dealing with any kind of distribution.

We now formally define the property. We use $a : b$ below to denote that a variable a is substituted by a matrix/vector b.

Definition 3.1 (Symbolic property). *A pair encoding scheme* $\Gamma = (\mathsf{Param}, \mathsf{EncCt}, \mathsf{EncKey}, \mathsf{Pair})$ *for a predicate family* $P_\kappa : \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0, 1\}$ *satisfies* (d_1, d_2)-selective symbolic property[9] *for positive integers* d_1 *and* d_2 *if there exist three deterministic polynomial-time algorithms* EncB*,* EncS*,* EncR *such that for all* $\kappa = (N, \mathsf{par})$*,* $x \in \mathcal{X}_\kappa$*,* $y \in \mathcal{Y}_\kappa$ *with* $P_\kappa(x, y) = 0$*,*

- $\mathsf{EncB}(x) \to \mathbf{B}_1, \ldots, \mathbf{B}_n \in \mathcal{G}_N(d_1, d_2)$*;*
- $\mathsf{EncS}(x) \to \mathbf{s}_0, \ldots, \mathbf{s}_{w_1} \in \mathcal{G}_N(d_2)$*,* $\quad \hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_{w_2} \in \mathcal{G}_N(d_1)$*;*
- $\mathsf{EncR}(x, y) \to \mathbf{r}_1, \ldots, \mathbf{r}_{m_1} \in \mathcal{G}_N(d_1)$*,* $\quad \mathbf{a}, \hat{\mathbf{r}}_1, \ldots, \hat{\mathbf{r}}_{m_2} \in \mathcal{G}_N(d_2)$*;*

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

$$\hat{s}_z : \hat{\mathbf{s}}_z^\mathsf{T} \qquad s_i b_j : \mathbf{B}_j \mathbf{s}_i^\mathsf{T} \qquad \alpha : \mathbf{a} \qquad \hat{r}_{z'} : \hat{\mathbf{r}}_{z'} \qquad r_{i'} b_j : \mathbf{r}_{i'} \mathbf{B}_j$$

for $z \in [w_2]$*,* $i \in [w_1]^+$*,* $j \in [n]$*,* $z' \in [m_2]$ *and* $i' \in [m_1]$ *in all the polynomials output by* EncCt *and* EncKey *on input* x *and* y*, respectively, they evaluate to* $\mathbf{0}$*.*

Similarly we say a pair encoding scheme satisfies (d_1, d_2)-co-selective symbolic security property *if there exist* $\mathsf{EncB}, \mathsf{EncR}, \mathsf{EncS}$ *that satisfy the above properties but where* EncB *and* EncR *depend only on* y*, and* EncS *depends on both* x *and* y*. Finally, a scheme satisfies* (d_1, d_2)-symbolic property *if it satisfies both* (d_1', d_2')*-selective and* (d_1'', d_2'')*-co-selective properties for some* $d_1', d_1'' \leq d_1$ *and* $d_2', d_2'' \leq d_2$*.*

[9] d_1, d_2 could depend on κ but we leave this implicit for simplicity of presentation.

We use Sym-Prop as a shorthand for symbolic property. It is easy to see that if a scheme satisfies (d_1, d_2)-selective Sym-Prop then it also satisfies (d'_1, d'_2) for any $d'_1 \geq d_1$ and $d'_2 \geq d_2$. Just append $d'_1 - d_1$ rows of zeroes and $d'_2 - d_2$ columns of zeroes to the $\mathbf{B}_j$ matrices, $d'_2 - d_2$ zeroes to the $\mathbf{s}_i$ vectors, $d'_1 - d_1$ zeroes to the $\hat{\mathbf{s}}_z$ vectors, $d'_1 - d_1$ zeroes to the $\mathbf{r}_{i'}$ vectors, and $d'_2 - d_2$ zeroes to the $\hat{\mathbf{r}}_{z'}$ vectors. A similar claim can also be made about co-selective Sym-Prop. Thus if a PES satisfies (d_1, d_2)-Sym-Prop then it also satisfies selective and co-selective properties with the same parameters, as well as (d'_1, d'_2)-Sym-Prop for any $d'_1 \geq d_1$ and $d'_2 \geq d_2$.

Lastly, if a PES Γ satisfies Sym-Prop for a predicate family P_κ, we say that Γ is *symbolically secure* for P_κ, or simply that Γ is symbolically secure if the predicate family is clear from context.

4 Obtaining Symbolic Security Generically

In this section, we prove an interesting and useful result. If a pair encoding scheme in not *trivially* broken in the sense that for any x, y that do not satisfy the predicate, there does not exist a way to directly recover αs_0 from the encoding polynomials (note that for correctness we require exactly this, but when the predicate is true), then the scheme satisfies the symbolic property.

Definition 4.1 (Trivially broken scheme). *A pair encoding scheme $\Gamma = (\mathsf{Param}, \mathsf{EncCt}, \mathsf{EncKey}, \mathsf{Pair})$ for a predicate family $P_\kappa : \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ is* trivially broken *if for a $\kappa = (N, \mathsf{par})$, $x \in \mathcal{X}_\kappa$, $y \in \mathcal{Y}_\kappa$ that satisfy $P_\kappa(x, y) = 0$, there exists a matrix $\mathbf{E}$ such that $(\mathbf{s}, \mathbf{c})\mathbf{E}(\mathbf{r}, \mathbf{k})^{\mathsf{T}} = \alpha s_0$, where $\mathbf{c}$ is the vector of polynomials output by $\mathsf{EncCt}(x, N)$ in variables $\mathbf{s} = (s_0, \ldots)$, $\hat{\mathbf{s}}$, $\mathbf{b}$, and $\mathbf{k}$ is the vector of polynomials output by $\mathsf{EncKey}(y, N)$ in variables $\mathbf{r}$, $\hat{\mathbf{r}} = (\alpha, \ldots)$, $\mathbf{b}$.*

Theorem 4.2. *If a pair encoding scheme is not trivially broken then it satisfies the symbolic property.*

Proof. If a scheme Γ is not trivially broken, then for all x and y for which the predicate evaluates to false, the ct-enc non-lone variables $\mathbf{s} = (s_0, \ldots, s_{w_1})$ and polynomials $\mathbf{c} = (c_1, \ldots, c_{w_3})$ cannot be paired with the key-enc non-lone variables $\mathbf{r} = (r_1, \ldots, r_{m_1})$ and polynomials $\mathbf{k} = (k_1, \ldots, k_{m_3})$ to recover αs_0. We know that the former have monomials of the form $s_0, \ldots, s_{w_1}$, $\hat{s}_1, \ldots, \hat{s}_{w_2}$, $s_0 b_1, \ldots, s_0 b_n$, …, $s_{w_1} b_1, \ldots, s_{w_1} b_n$, so a total of $w_2 + (n+1)(w_1+1)$. Similarly, the total number of distinct monomials in the latter is $m_2 + 1 + (n+1)m_1$ (because α is a lone variable as opposed to s_0). Let us denote the two quantities above by var_c and var_k respectively.

Define a matrix $\mathbf{\Delta}$ over $\mathbb{Z}_N$ with $(w_1 + w_3 + 1)(m_1 + m_3)$ rows and $\mathsf{var}_c \mathsf{var}_k$ columns. A row is associated with the product of a ct-enc non-lone variable or polynomial with a key-enc non-lone variable or polynomial. Each column represents a unique monomial that can be obtained by multiplying a ct-enc monomial with a key-enc monomial, with the first column representing αs_0. The (i, j)th entry in this matrix is the coefficient of the monomial associated with the jth column in the product polynomial attached with the ith row. Since Γ is not broken, we know that the rows in $\mathbf{\Delta}$ cannot be linearly combined to get the vector $(1, 0, \ldots, 0)$.

Note that it is enough to work with any subset of rows because they cannot be combined to get $(1, 0, \ldots, 0)$ either. Thus, for the rest of the proof, we consider only those rows of $\mathbf{\Delta}$ that multiply a $\mathsf{ct\text{-}enc}$ non-lone variable with a $\mathsf{key\text{-}enc}$ polynomial and vice versa (and only those columns which have monomials that can be obtained from multiplying such polynomials). Let n_1 denote the number of rows now.

Since rows in $\mathbf{\Delta}$ cannot be linearly combined to get $(1, 0, \ldots, 0)$, the first column of $\mathbf{\Delta}$, say col, can be written as a linear combination of the other columns. Because if not, one can show that there exists a vector $\mathbf{v} = (v_1, \ldots, v_{n_1})$ that is orthogonal to all the columns except the first one[10]. We can then combine the rows of $\mathbf{\Delta}$ using $v_1 / \langle \mathsf{col}, \mathbf{v} \rangle, \ldots, v_{n_1} / \langle \mathsf{col}, \mathbf{v} \rangle$ to get $(1, 0, \ldots, 0)$—a contradiction.

Let $\mathcal{Q}$ denote the set of monomials associated with the columns of $\mathbf{\Delta}$. These columns can be linearly combined to get the zero vector, without zeroing out col, which corresponds to αs_0. Let λ_q be the factor that multiplies the column associated with the monomial $q \in \mathcal{Q}$ in one such linear combination. Note that $\lambda_{\alpha s_0} \neq 0$.

Our first goal is to show that Γ satisfies the selective symbolic property. So we need to define matrices and vectors for various variables in the encoding such that all the polynomials evaluate to the zero vector. Towards this, pick any non-lone $\mathsf{key\text{-}enc}$ variable $r_{i'}$ for $i' \in [m_1]$ and consider the sub-matrix $\mathbf{\Delta'}$ of $\mathbf{\Delta}$ that consists of rows which are attached with the product of $r_{i'}$ with a $\mathsf{ct\text{-}enc}$ polynomial and columns which are associated with the product of $r_{i'}$ and a $\mathsf{ct\text{-}enc}$ monomial. (Note that it does not matter which non-lone $\mathsf{key\text{-}enc}$ variable we consider; the sub-matrix obtained in each case will be exactly the same.) Recall that a $\mathsf{ct\text{-}enc}$ polynomial c_ℓ is given by

$$\sum_{z \in [w_2]} \eta_{\ell,z} \hat{s}_z \quad + \sum_{i \in [w_1]^+, j \in [n]} \eta_{\ell,i,j} s_i b_j$$

for $\ell \in [w_3]$. So more formally, rows in $\mathbf{\Delta'}$ are associated with $(c_\ell, r_{i'})$, and columns are associated with monomials $\hat{s}_z r_{i'}$, $s_i b_j r_{i'}$, where the range of i, j, z is as described above. For simplicity in the following, assume that the columns are ordered as $\hat{s}_1, \ldots, \hat{s}_{w_2}$, $s_0 b_1, \ldots, s_0, b_n$, $\ldots$, $s_{w_1} b_1, \ldots, s_{w_1} b_n$ and the rows are ordered as $(c_1, r_{i'}), \ldots, (c_{w_3}, r_{i'})$, so that the lth row of $\mathbf{\Delta'}$ is $(\eta_{\ell,1}, \ldots, \eta_{\ell,w_2}, \eta_{\ell,0,1}, \ldots, \eta_{\ell,0,n}, \ldots, \eta_{\ell,w_1,1}, \ldots, \eta_{\ell,w_1,n})$.

Let $\mathcal{T}$ be the kernel of $\mathbf{\Delta'}$, i.e. the set of all vectors $\mathbf{v}$ such that $\mathbf{\Delta' v} = \mathbf{0}$. Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{d_1}$ be a basis of $\mathcal{T}$ and write $\mathbf{v}_p$ as $(v_{p,1}, \ldots, v_{p,w_2}, v_{p,0,1}, \ldots, v_{p,0,n},$

[10] The claim is similar to one made in the case of linear secret sharing schemes where we say that if a set of attributes does not satisfy a policy, i.e. the associated set of rows cannot be linearly combined to get a certain vector $\mathbf{v}$, then one can find a vector orthogonal to all those rows but not to $\mathbf{v}$. See, for instance, [9, Claim 2] for a formal proof.

$\ldots, v_{p,w_1,1}, \ldots, v_{p,w_1,n})$ for $p \in [d_1]$. (We discuss the special case of $\mathbf{\Delta}'$'s kernel being empty later on.) Therefore, we have that for any $\ell \in [w_3]$ and $p \in [d_1]$,

$$\sum_z \eta_{\ell,z} v_{p,z} \quad + \quad \sum_{i,j} \eta_{\ell,i,j} v_{p,i,j} \tag{2}$$

is equal to 0. Let $\mathbf{u}_z = (v_{1,z}, \ldots, v_{d_1,z})$ and $\mathbf{u}_{i,j} = (v_{1,i,j}, \ldots, v_{d_1,i,j})$ for $z \in [w_2]$, $i \in [w_1]^+$, $j \in [n]$.

We now define matrices $\mathbf{B}_1, \ldots, \mathbf{B}_n$ and vectors $\mathbf{s}_0, \ldots, \mathbf{s}_{w_1}, \hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_{w_2}$ as follows. $\mathbf{B}_j$ has d_1 rows and $d_2 = w_1 + 1$ columns with the $(i+1)$th column being $\mathbf{u}_{i,j}^{\mathsf{T}}$ for $i = [w_1]^+$. Vector $\mathbf{s}_i$ is set to $\mathbf{e}_{i+1}$ for $i = [w_1]^+$, where $\mathbf{e}_i$ denotes the ith unit vector of size d_2, and $\hat{\mathbf{s}}_z$ is set to $\mathbf{u}_z$ for $z \in [w_2]$. These matrices and vectors depend only on $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{d_1}$, which in turn depends on $\mathbf{\Delta}'$ only. The entries in $\mathbf{\Delta}'$ are the coefficients of the monomials obtained by multiplying $r_{i'}$ with various ct-enc polynomials. Hence, they only depend on x and, in particular, not on y. Further, it is easy to observe that all the operations involved in computing $\mathbf{B}_j$, $\mathbf{s}_i$, $\hat{\mathbf{s}}_z$ are efficient. Thus, one can define two deterministic polynomial time algorithms EncB and EncS that on input x only, output $\mathbf{B}_1, \ldots, \mathbf{B}_n$ and $\mathbf{s}_0, \ldots, \mathbf{s}_{w_1}$, $\hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_{w_2}$ respectively.

We need to verify that if we substitute $\hat{s}_z$ with $\hat{\mathbf{s}}_z^{\mathsf{T}}$ and $s_i b_j$ with $\mathbf{B}_j \mathbf{s}_i^{\mathsf{T}}$ in any ct-enc polynomial c_ℓ, then we get an all zeroes vector. On performing such a substitution, we have

$$\sum_z \eta_{\ell,z} \mathbf{u}_z^{\mathsf{T}} \quad + \quad \sum_{i,j} \eta_{\ell,i,j} (\mathbf{u}_{0,j}^{\mathsf{T}}, \ldots, \mathbf{u}_{w_1,j}^{\mathsf{T}}) \mathbf{e}_{i+1}^{\mathsf{T}} \quad = \quad \sum_z \eta_{\ell,z} \mathbf{u}_z^{\mathsf{T}} \quad + \quad \sum_{i,j} \eta_{\ell,i,j} \mathbf{u}_{i,j}^{\mathsf{T}}$$

The pth element in the column vector above is given by (2), which is equal to 0 for any p.

In the special case where $\mathbf{\Delta}'$'s kernel is empty, $\mathbf{B}_1, \ldots, \mathbf{B}_n$ are all set to $d_1 \times d_2$ matrices with zero entries; $\hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_{w_2}$ are set to the zero vector of size d_1; $\mathbf{s}_1, \ldots, \mathbf{s}_{w_1}$ are set to the zero vector of size d_2; and $\mathbf{s}_0$ is set to $(1, 0, \ldots, 0)$. It is easy to see that all ct-enc polynomials still evaluate to zero upon substitution.

We also need to make sure that with the appropriate choice of vectors for the key-enc variables, all the key-enc polynomials also evaluate to the zero vector. Recall that such polynomials are given by

$$k_t \quad = \quad \phi_t \alpha \quad + \quad \sum_{z' \in [m_2]} \phi_{t,z'} \hat{r}_{z'} \quad + \quad \sum_{\substack{i' \in [m_1], \\ j \in [n]}} \phi_{t,i',j} r_{i'} b_j$$

for $t \in [m_3]$. When they are multiplied with a non-lone ct-enc variable s_i, we get the monomials αs_i, $s_i \hat{r}_{z'}$, $s_i r_{i'} b_j$ for $i \in [w_1]^+$ and i', j, z' as above.

Recall that the columns of $\mathit{\Delta}$ can be linearly combined using $\{\lambda_q\}_{q \in Q}$ to get the zero vector. Going back to the product of $r_{i'}$ with c_ℓ, we can say that

$$\sum_z \eta_{\ell,z} \lambda_{\hat{s}_z r_{i'}} \quad + \quad \sum_{i,j} \eta_{\ell,i,j} \lambda_{s_i b_j r_{i'}} \quad = \quad 0$$

irrespective of what ℓ and i' are because only the entries in the columns associated with monomials $\hat{s}_z r_{i'}$, $s_i b_j r_{i'}$ are non-zero. Hence, the vector $\mathbf{w}_{i'}$ given by $(\lambda_{\hat{s}_1 r_{i'}}, \ldots, \lambda_{\hat{s}_{w_2} r_{i'}}, \lambda_{s_0 b_1 r_{i'}}, \ldots, \lambda_{s_0 b_n r_{i'}}, \ldots, \lambda_{s_{w_1} b_1 r_{i'}}, \ldots, \lambda_{s_{w_1} b_n r_{i'}})$ lies in the kernel of $\mathbf{\Delta'}$. (Recall that no matter what key-enc non-lone variable is chosen, one always gets the same $\mathbf{\Delta'}$.) In other words, there exists a vector $\mathbf{r}_{i'}$ of size d_1 such that $[\mathbf{v}_1^\mathsf{T}, \ldots, \mathbf{v}_{d_1}^\mathsf{T}]\mathbf{r}_{i'}^\mathsf{T} = \mathbf{w}_{i'}$. Now the transpose of $\mathbf{r}_{i'}\mathbf{B}_j$ is given by

$$\begin{bmatrix} \mathbf{u}_{0,j} \\ \vdots \\ \mathbf{u}_{w_1,j} \end{bmatrix} \mathbf{r}_{i'}^\mathsf{T} \quad = \quad \begin{bmatrix} v_{1,0,j} & \ldots & v_{d_1,0,j} \\ \vdots & \vdots & \vdots \\ v_{1,w_1,j} & \ldots & v_{d_1,w_1,j} \end{bmatrix} \mathbf{r}_{i'}^\mathsf{T} \quad = \quad \begin{bmatrix} \lambda_{s_0 b_j r_{i'}} \\ \vdots \\ \lambda_{s_{w_1} b_j r_{i'}} \end{bmatrix}$$

for every $j \in [n]$. In the special case where Δ''s kernel is empty, set $\mathbf{r}_{i'}$ to be the zero vector of size d_1. The relation $\mathbf{r}_{i'}\mathbf{B}_j = (\lambda_{s_0 b_j r_{i'}}, \ldots, \lambda_{s_{w_1} b_j r_{i'}})$ for all j still holds because $\mathbf{w}_{i'}$ must be zero.

Define the remaining vectors as follows: $\mathbf{a}$ is set to be $[\lambda_{\alpha s_0}, \ldots, \lambda_{\alpha s_{w_1}}]$ and $\hat{\mathbf{r}}_{z'}$ to be $[\lambda_{s_0 \hat{r}_{z'}}, \ldots, \lambda_{s_{w_1} \hat{r}_{z'}}]$ for $z' \in [m_2]$. (Note that the first element of $\mathbf{a}$ is not zero.) When we substitute α with $\mathbf{a}$, $\hat{r}_{z'}$ with $\hat{\mathbf{r}}_{z'}$ and $r_{i'} b_j$ with $\mathbf{r}_{i'}\mathbf{B}_j$ in k_t for $t \in [m_3]$, we get

$$\phi_t[\lambda_{\alpha s_0}, \ldots, \lambda_{\alpha s_{w_1}}] \quad + \quad \sum_{z'} \phi_{t,z'}[\lambda_{s_0 \hat{r}_{z'}}, \ldots, \lambda_{s_{w_1} \hat{r}_{z'}}] \\ + \quad \sum_{i',j} \phi_{t,i',j}[\lambda_{s_0 b_j r_{i'}}, \ldots, \lambda_{s_{w_1} b_j r_{i'}}].$$

The ith element of this sum is given by

$$\phi_t \lambda_{\alpha s_i} \quad + \quad \sum_{z'} \phi_{t,z'} \lambda_{s_i \hat{r}_{z'}} \quad + \quad \sum_{i',j} \phi_{t,i',j} \lambda_{s_i r_{i'} b_j}$$

for $i \in [w_1]^+$. It is easy to see that the above quantity is zero when we consider the row in $\mathbf{\Delta}$ attached with the product $s_i k_t$.

One can define a deterministic polynomial time algorithm EncR that on input x and y, computes how the columns of $\mathbf{\Delta}$ can be combined to get the zero vector, and then uses this information to define $\mathbf{a}$, $\hat{\mathbf{r}}_{z'}$, $\mathbf{r}_{i'}$ as shown above.

The proof for the co-selective symbolic property is analogous to the proof above, so we skip the details. $\square$

5 Predicate encryption from Pair Encodings

In this section, we describe how any pair encoding scheme for a predicate can be transformed into an encryption scheme for the same predicate in dual system groups (DSG), introduced by Chen and Wee [16], and later used and improved by several works [15,1,4]. This transformation is a two-step process: first we *augment* an encoding so that it satisfies a few extra properties (Section 5.1)[11]; next we apply the transformation from Agrawal and Chase [1] (Section 5.4).

[11] This step need not be applied if the properties are already satisfied.

5.1 Augmenting pair encodings

We need the matrices and vectors involved in the symbolic property to have some extra features, so that we can prove the security of the derived predicate encryption scheme from our q-ratio assumption. Towards this, we show how any pair encoding scheme that satisfies Sym-Prop can be transformed into another scheme that satisfies a more constrained version of this property, with only a few additional variables and polynomials.

We note that, although they are presented monolithically, many of the pair encodings introduced by Attrapadung [2] can be viewed as the result of applying a very similar augmentation to simpler underlying encodings. Thus, our results also help explain the structure of those previous encodings.

Recall that the algorithms of symbolic security output $\mathbf{a}$ for α, $\mathbf{B}_1, \ldots, \mathbf{B}_n$ for common variables, $\mathbf{s}_0, \ldots, \mathbf{s}_{w_1}$ for non-lone ct-enc variables, and $\mathbf{r}_1, \ldots, \mathbf{r}_{m_1}$ for key-enc non-lone variables. Let $\mathbf{b}_j$ denote the first column of $\mathbf{B}_j$ and $s_{i,1}$ the first element of $\mathbf{s}_i$.

Definition 5.1 (Enhanced symbolic property). *A pair encoding scheme satisfies (d_1, d_2)-*Sym-Prop* *for a predicate P_κ if it satisfies selective and co-selective (d_1, d_2)-*Sym-Prop *for P_κ but under the following constraints for both*

1. *$\mathbf{a}$ is set to $(1, 0, \ldots, 0)$.*
2. *In every* ct-enc *polynomial, if $s_i b_j$ is replaced by*
   - *$\mathbf{s}_i^\top \mathbf{b}_j$ then we get a matrix with non-zero elements in the first row only;*
   - *$s_{i,1} \mathbf{B}_j$ then we get a matrix with non-zero elements in the first column only.*

   (The lone variables are replaced by the zero vector.)
3. *In every* key-enc *polynomial, if we replace $r_{i'} b_j$ with $\mathbf{b}_j^\top \mathbf{r}_{i'}$, then we get a diagonal matrix. (The lone variables, once again, are replaced by the zero vector.)*
4. *The set of vectors $\{\mathbf{s}_0, \ldots, \mathbf{s}_{w_1}\}$ is linearly independent, and so is the set $\{\mathbf{r}_1, \ldots, \mathbf{r}_{m_1}\}$.*

We convert any pair encoding that satisfies Sym-Prop into one that satisfies Sym-Prop* in three steps. First we show that with only one additional key-enc non-lone variable, an additional common variable, and an extra ct-enc polynomial, we can get an encoding scheme for which the vector $\mathbf{a}$ corresponding to α can be set to $(1, 0, \ldots, 0)$ (in proving that Sym-Prop holds). Next, with two extra common variables, and an additional variable and a polynomial each in the ciphertext and key encoding, one can satisfy the second and third properties from above. Finally, a simple observation can be used to satisfy the fourth property as well. More formally, we prove the following theorem in the full version.

Theorem 5.2 (Augmentation). *Suppose a* PES *for a predicate family $P_\kappa : \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ outputs n on input* par*, $(w_1, w_2, \mathbf{c})$ on input $x \in \mathcal{X}_\kappa$, $(m_1, m_2, \mathbf{k})$ on input $y \in \mathcal{Y}_\kappa$ and satisfies (d_1, d_2)-*Sym-Prop*, then there exists another* PES *for P_κ that outputs $n + 3$ on input* par*, $(w_1 + 1, w_2, \overline{\mathbf{c}})$ on input x and $(m_1 + 2, m_2, \overline{\mathbf{k}})$ on input y, where $|\overline{\mathbf{c}}| = |\mathbf{c}| + 2$ and $|\overline{\mathbf{k}}| = |\mathbf{k}| + 1$, and satisfies* $(\max(d_1, d_2 -$

$1)+M_1+1, d_2+W_1+2)$-$\mathsf{Sym\text{-}Prop}^\star$*, where* M_1 *and* W_1 *are bounds on the number of* $\mathsf{key\text{-}enc}$ *and* $\mathsf{ct\text{-}enc}$ *non-lone variables, respectively.*[12]

The extra constraints of $\mathsf{Sym\text{-}Prop}^\star$ give rise to some nice combinatorial facts. Please refer to the full version for details.

5.2 Dual System Groups

Dual system groups (DSG) were introduced by Chen and Wee [16] and generalized by Agrawal and Chase [1]. The latter work also shows that the two instantiations of DSG – in composite-order groups under the subgroup decision assumption and in prime-order groups under the decisional linear assumption – given by Chen and Wee satisfy the generalized definition as well. Here we give a brief informal description of dual system groups. See the full version or existing work [1] for a formal definition.

Dual system groups are parameterized by a security parameter λ and a number n. They have a SampP algorithm that on input 1^λ and 1^n, outputs public parameters PP and secret parameters SP. The parameter PP contains a triple of groups $(\mathbb{G}, \mathbb{H}, \mathbb{G}_T)$ and a non-degenerate bilinear map $e : \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$, a homomorphism μ from $\mathbb{H}$ to $\mathbb{G}_T$, along with some additional parameters used by SampG, SampH. Given PP, we know the exponent of group $\mathbb{H}$ and how to sample uniformly from it; let $N = \exp(\mathbb{H})$. It is required that N is a product of distinct primes of $\Theta(\lambda)$ bits. The secret parameters SP contain $\tilde{h} \in \mathbb{H}$ (where $\tilde{h} \neq 1_{\mathbb{H}}$) along with additional parameters used by $\overline{\mathsf{SampG}}$ and $\overline{\mathsf{SampH}}$.

A dual system group has several sampling algorithms: SampGT algorithm takes an element in the image of μ and outputs another element from $\mathbb{G}_T$. SampG and SampH take PP as input and output a vector of $n+1$ elements from $\mathbb{G}$ and $\mathbb{H}$ respectively. $\overline{\mathsf{SampG}}$ and $\overline{\mathsf{SampH}}$ take both PP and SP as inputs and output a vector of $n+1$ elements from $\mathbb{G}$ and $\mathbb{H}$ respectively. These two algorithms are used in security proofs only. $\overline{\mathsf{SampG}}_0$ and $\overline{\mathsf{SampH}}_0$ denote the first element of $\overline{\mathsf{SampG}}$ and $\overline{\mathsf{SampH}}$ respectively.

A dual system group is *correct* if it satisfies the following two properties for all PP.

– *Projective*: For all $h \in \mathbb{H}$ and coin tosses σ, $\mathsf{SampGT}(\mu(h); \sigma) = e(\mathsf{SampG}_0(\mathrm{PP}; \sigma), h)$, where SampG_0 is an algorithm that outputs only the first element of SampG.
– *Associative*: If $(g_0, g_1, \ldots, g_n)$ and $(h_0, h_1, \ldots, h_n)$ are samples from $\mathsf{SampG}(\mathrm{PP})$ and $\mathsf{SampH}(\mathrm{PP})$ respectively, then for all $i \in [1, n]$, $e(g_0, h_i) = e(g_i, h_0)$.

Dual system groups have a number of interesting security properties as well that makes them very useful for building encryption schemes, see the full version

[12] As we will see later, when a pair encoding scheme is transformed into a predicate encryption scheme, the parameters of $\mathsf{Sym\text{-}Prop}^\star$ have no effect on the construction. They only affect the size of assumption on which the security of encryption scheme is based.

for details. We additionally require that there exists a way to sample the setup parameters so that one not only gets PP and SP, but also some trapdoor information td that can be used to generate samples from $\overline{\mathsf{SampG}}$ and $\overline{\mathsf{SampH}}$ given only the first element. We formalize this property and show that both instantiations of Chen and Wee [16] satisfy them in the full version. The new sampling algorithm will be denoted by SampP^* below.

5.3 New computational assumption

We introduce a new assumption, called $\mathsf{q\text{-}ratio_{dsg}}$, on dual system groups parameterized by positive integers d_1 and d_2.

Definition 5.3 **((d_1, d_2)-$\mathsf{q\text{-}ratio_{dsg}}$ assumption).** *Consider the following distribution on a dual system group's elements:*

$$\mathsf{dsg\text{-}par} := (\text{PP}, \text{SP}, \mathsf{td}) \leftarrow \mathsf{SampP}^*(1^\lambda, 1^n);$$

$$\hat{g} \leftarrow \overline{\mathsf{SampG}}_0(\text{PP}, \text{SP}); \quad \hat{h} \leftarrow \overline{\mathsf{SampH}}_0(\text{PP}, \text{SP})$$

$$u_0, u_1, \ldots, u_{d_2}, v_1, \ldots, v_{d_1} \leftarrow_R \mathbb{Z}_N^*;$$

$$D_{\mathbb{G}} \quad := \quad \{\hat{g}^{u_i}\}_{i\in[d_2]^+} \quad \cup \quad \left\{\hat{g}^{\frac{u_i}{u_j v_k}}\right\}_{i,j\in[d_2], i\neq j, k\in[d_1]};$$

$$D_{\mathbb{H}} \quad := \quad \{\hat{h}^{v_i}\}_{i\in[d_1]} \quad \cup \quad \left\{\hat{h}^{\frac{v_i}{v_j u_k}}\right\}_{i,j\in[d_1], i\neq j, k\in[d_2]};$$

$$T_0 := \hat{h}^{1/u_0}; \quad T_1 \leftarrow_R \mathbb{H}.$$

We say that the (d_1, d_2)-$\mathsf{q\text{-}ratio_{dsg}}$ assumption holds if for any PPT *algorithm $\mathcal{A}$,*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{qr_{dsg}}}(\lambda) := \big|\Pr[\mathcal{A}(1^\lambda, \mathsf{dsg\text{-}par}, D_{\mathbb{G}}, D_{\mathbb{H}}, T_0) = 1] \\ - \Pr[\mathcal{A}(1^\lambda, \mathsf{dsg\text{-}par}, D_{\mathbb{G}}, D_{\mathbb{H}}, T_1) = 1]\big|$$

is negligible in λ.

Note that u_0 is present in exactly one of the terms in $D_{\mathbb{G}}$ and not at all in $D_{\mathbb{H}}$.

We also define a similar assumption on bilinear maps.

Definition 5.4 **((d_1, d_2)-$\mathsf{q\text{-}ratio}$ assumption).** *Consider the following distribution:*

$$\mathsf{par} := (N, \mathcal{G}, \mathcal{H}, \mathcal{G}_T, g, h, e) \leftarrow \mathsf{GroupGen}(1^\lambda)$$

$$\hat{g} \leftarrow_R \mathcal{G}; \quad \hat{h} \leftarrow_R \mathcal{H}; \qquad u_0, u_1, \ldots, u_{d_2}, v_1, \ldots, v_{d_1} \leftarrow_R \mathbb{Z}_N^*;$$

$$D_{\mathcal{G}} \quad := \quad \{\hat{g}^{u_i}\}_{i\in[d_2]^+} \quad \cup \quad \left\{\hat{g}^{\frac{u_i}{u_j v_k}}\right\}_{i,j\in[d_2], i\neq j, k\in[d_1]};$$

$$D_{\mathcal{H}} \quad := \quad \{\hat{h}^{v_i}\}_{i\in[d_1]} \quad \cup \quad \left\{\hat{h}^{\frac{v_i}{v_j u_k}}\right\}_{i,j\in[d_1], i\neq j, k\in[d_2]};$$

$$T_0 := \hat{h}^{1/u_0}; \quad T_1 \leftarrow_R \mathcal{H}.$$

We say that the (d_1, d_2)-$\mathsf{q\text{-}ratio}$ *assumption holds if for any* PPT *algorithm* $\mathcal{A}$*,*

$$\mathsf{Adv}^{\mathsf{qr}}_{\mathcal{A}}(\lambda) := \big|\mathsf{Pr}[\mathcal{A}(1^\lambda, \mathsf{par}, D_{\mathcal{G}}, D_{\mathcal{H}}, T_0) = 1] \\ - \mathsf{Pr}[\mathcal{A}(1^\lambda, \mathsf{par}, D_{\mathcal{G}}, D_{\mathcal{H}}, T_1) = 1]\big|$$

is negligible in λ*.*

In this paper our focus is on constructions in prime-order groups because they are much more practical, so we will consider the $\mathsf{q\text{-}ratio}$ assumption on prime-order bilinear maps only. We show that this assumption is implied by the assumptions proposed by Lewko, Waters [30] and Attrapadung [2] in the full version. We also show that Chen and Wee's prime order DSG construction [16] (along with the new sampling algorithms we introduce) satisfies the $\mathsf{q\text{-}ratio}_{\mathsf{dsg}}$ assumption if the underlying group satisfies the $\mathsf{q\text{-}ratio}$ assumption. Thus we have,

Lemma 5.5. *A dual system group with a bilinear map* $e : \mathbb{G} \times \mathbb{H} \rightarrow \mathbb{G}_T$ *that satisfies the* (d_1, d_2)-$\mathsf{q\text{-}ratio}_{\mathsf{dsg}}$ *assumption can be instantiated in a prime-order bilinear map* $e' : \mathcal{G} \times \mathcal{H} \rightarrow \mathcal{G}_T$ *that satisfies the* (d_1, d_2)-$\mathsf{q\text{-}ratio}$ *and* k*-linear assumptions. Further, an element of* $\mathbb{G}$ *and* $\mathbb{H}$ *is represented using* $k+1$ *elements of* $\mathcal{G}$ *and* $\mathcal{H}$*, respectively. (An element of* $\mathbb{G}_T$ *is represented by just one from* $\mathcal{G}_T$*).*

5.4 Encryption Scheme

In this section, we show how to obtain an encryption scheme from a pair encoding using the sampling algorithms of dual system groups. Our transformation is based on the one given by Agrawal and Chase [1], and is referred to as $\mathsf{Gen\text{-}Trans}$. If a PES $\mathit{\Gamma}_P$ is defined by the tuple of algorithms $(\mathsf{Param}, \mathsf{EncCt}, \mathsf{EncKey}, \mathsf{Pair})$ for a predicate family $P = \{P_\kappa\}_{\kappa \in \mathbb{N}^c}$, then the algorithms for $\mathit{\Pi}_P := \mathsf{Gen\text{-}Trans}(\mathit{\Gamma}_P)$ are given as follows.

- $\mathsf{Setup}(1^\lambda, \mathsf{par})$: First the pair encoding algorithm $\mathsf{Param}(\mathsf{par})$ is run to obtain n, and then the dual system group algorithm $\mathsf{SampP}(1^\lambda, 1^n)$ is run to get $\textsc{pp}$, $\textsc{sp}$. A randomly chosen element from $\mathbb{H}$ is designated to be the master secret key $\textsc{msk}$. Master public key $\textsc{mpk}$ is set to be $(\textsc{pp}, \mu(\textsc{msk}))$. Further, N and κ are set to $\exp(\mathbb{H})$ and (N, par), respectively (where the exponent of $\mathbb{H}$ is a part of $\textsc{pp}$).
- $\mathsf{Encrypt}(\textsc{mpk}, x, \mathsf{msg})$: On input $x \in \mathcal{X}_\kappa$ and $\mathsf{msg} \in \mathbb{G}_T$, $\mathsf{EncCt}(x, N)$ is run to obtain w_1, w_2 and polynomials $(c_1, \ldots, c_{w_3})$. For $i' \in [w_1 + w_2]^+$, draw a sample $(g_{i',0}, \ldots, g_{i',n})$ from SampG using $\textsc{pp}$. Recall that the ℓth polynomial is given by
$$\sum_{z \in [w_2]} \eta_{\ell,z} \hat{s}_z \quad + \sum_{i \in [w_1]^+, j \in [n]} \eta_{\ell,i,j} s_i b_j.$$

Set CT_i to be $g_{i,0}$ for $i \in [w_1]^+$ and $\widetilde{\text{CT}}_\ell$ to be

$$\prod_{z\in[w_2]} g_{w_1+z,0}^{\eta_{\ell,z}} \quad \cdot \quad \prod_{i\in[w_1]^+, j\in[n]} g_{i,j}^{\eta_{\ell,i,j}}$$

for $\ell \in [w_3]$. Also, let $\text{CT}^\star = \mathsf{msg} \cdot \mathsf{SampGT}(\mu(\text{MSK}); \sigma)$ where σ denotes the coin tosses used in drawing the first sample from SampG. Output $\text{CT} := (\text{CT}_0, \ldots, \text{CT}_{w_1}, \widetilde{\text{CT}}_1, \ldots, \widetilde{\text{CT}}_{w_3}, \text{CT}^\star)$.

– $\mathsf{KeyGen}(\text{MPK}, \text{MSK}, y)$: On input $y \in \mathcal{Y}_\kappa$, $\mathsf{EncKey}(y, N)$ is run to obtain m_1, m_2 and polynomials $(k_1, k_2, \ldots, k_{m_3})$. For $i \in [m_1 + m_2]$, draw a sample $(h_{i,0}, \ldots, h_{i,n})$ from SampH using PP. Recall the tth polynomial is given by

$$\phi_t\alpha \quad + \sum_{z'\in[m_2]} \phi_{t,z'}\hat{r}_{z'} \quad + \sum_{i'\in[m_1], j\in[n]} \phi_{t,i',j} r_{i'} b_j .$$

Set $\text{SK}_{i'}$ to be $h_{i',0}$ for $i' \in [m_1]$ and $\widetilde{\text{SK}}_t$ to be

$$\text{MSK}^{\phi_t} \quad \cdot \quad \prod_{z'\in[m_2]} h_{m_1+z',0}^{\phi_{t,z'}} \quad \cdot \quad \prod_{i'\in[m_1], j\in[n]} h_{i',j}^{\phi_{t,i',j}}$$

for $t \in [m_3]$. Output $\text{SK} := (\text{SK}_1, \ldots, \text{SK}_{m_1}, \widetilde{\text{SK}}_1, \ldots, \widetilde{\text{SK}}_{m_3})$.

– $\mathsf{Decrypt}(\text{MPK}, \text{SK}_y, \text{CT}_x)$: On input SK_y and CT_x, $\mathsf{Pair}(x, y, N)$ is run to obtain matrices $\mathbf{E}$ and $\overline{\mathbf{E}}$. Output

$$\text{CT}^\star \quad \cdot \quad \left(\prod_{i\in[w_1]^+, t\in[m_3]} e(\text{CT}_i, \widetilde{\text{SK}}_t)^{E_{i,t}} \quad \cdot \quad \prod_{\ell\in[w_3], i'\in[m_1]} e(\widetilde{\text{CT}}_\ell, \text{SK}_{i'})^{\overline{E}_{\ell,i'}} \right)^{-1} .$$

One can use the projective and associative property of DSG to show that the predicate encryption scheme defined above is correct (see [1] for details). We defer a proof of security for Π_P to Section 7, and conclude with the following remark.

Remark 5.6 (Size of ciphertexts and keys). Ciphertexts have w_1+w_3+1 elements from $\mathbb{G}$ and an element from $\mathbb{G}_T$; keys have m_1+m_3 elements from $\mathbb{H}$. So the size of these objects depends only on the number of non-lone variables and polynomials. Moreover, there is a one-to-one mapping between variables/polynomials and ciphertext/key elements. Thus if we can reduce the size of an encoding, we will immediately get an equivalent reduction in the size of ciphertexts or keys.

6 Transformations on Pair Encodings

In this section we present several useful transformations on pair encodings that preserve symbolic property. The first class of transformations help in reducing the size of ciphertexts and keys, and the second one provides a way to develop schemes for *dual* predicates (where the role of the two inputs to a predicate is reversed).

Compact encoding schemes. We show how pair encoding schemes can be made compact by reducing the number of ct-enc and/or key-enc polynomials and/or variables to a constant in a *generic* way. Importantly, we show that if the encoding scheme we start with satisfies the symbolic property, then so does the transformed scheme. As a result, building encryption schemes with constant-size ciphertexts or keys, for instance, becomes a very simple process.

Our first transformation converts any encoding scheme Γ' to another scheme Γ where the number of ct-enc variables is *just one*. Naturally, we need to assume a bound on the total number of ct-enc variables for this transformation to work. If W_1+1 and W_2 are bounds on the number of non-lone and lone ct-enc variables, respectively, and the number of common variables in Γ' is n, then Γ has $(W_1+1)n+W_2$ common variables, 1 ct-enc non-lone variable and 0 lone variables. The number of lone key-enc variables and polynomials increases by a multiplicative factor of W_1+1.

Our second transformation brings down the number of ct-enc polynomials to *just one*. Once again the transformation is fully generic, as long as there is a bound W_3 on the number of polynomials. In this case, the number of common variables increases by a multiplicative factor of W_3+1, the number of non-lone key-enc variables by a multiplicative factor of W_3, and the number of key-enc polynomials by an additive factor of $m_1W_3^2n$.

When the two transformations above are applied one after the other, we obtain an encoding scheme with just one non-lone variable and one polynomial in the ciphertext encoding. After augmenting the scheme as per Theorem 5.2 which adds a non-lone variable and two polynomials, we can convert the resulting encoding scheme into a predicate encryption scheme by using the generic mechanism of Section 5.4. This encryption scheme will have exactly 5 dual system's source group elements in any ciphertext, a number which would only double if the instantiation from Lemma 5.5 is used under the SXDH (1-linear) assumption.

One can also reduce the number of key-enc variables and polynomials in a manner analogous to how the corresponding quantities are reduced in the ciphertext encoding, at the cost of increasing the number of common variables and ct-enc variables and polynomials. If there is a bound on both the number of variables and polynomials in the key encoding, then one can obtain an encoding scheme with just one of each. This will result in encryption schemes with constant-size key.

Finally, we remark that one can also mix-and-match. For instance, first the number of ct-enc variables can be reduced to one, and then we can do the same for key-enc variables, resulting in a scheme with just one variable each in the ciphertext and key encodings at the cost of more polynomials in both. (This might be interesting, for example, because it produces a pair encoding of the form used in [15].) Note that when the ciphertext variable reduction transformation is applied, no lone variables are left in the ciphertext encoding (the only remaining variable is a non-lone variable). Hence, the key variable reduction transformation does not affect the number of ct-enc variables.

See the full version for a formal treatment of the two transformations described above.

Dual predicates. The dual predicate for a family $P'_\kappa : \mathcal{Y}_\kappa \times \mathcal{X}_\kappa \to \{0,1\}$ is given by $P_\kappa : \mathcal{X}_\kappa \times \mathcal{Y}_\kappa \to \{0,1\}$ where $P_\kappa(x,y) = P'_\kappa(y,x)$ for all κ, $x \in \mathcal{X}_\kappa$, $y \in \mathcal{Y}_\kappa$. For example, CP-ABE and KP-ABE are duals of each other. In the full version we show that Attrapadung's dual scheme conversion [3, Section 8.1] mechanism preserves symbolic property too.

7 Security of Predicate Encryption Scheme

In this section we show that the transformation Gen-Trans leads to a secure encryption scheme if the underlying encoding satisfies the (enhanced) symbolic property. More formally, we have:

Theorem 7.1. *If a pair encoding scheme Γ_P satisfies (d_1, d_2)-$\mathsf{Sym\text{-}Prop}^\star$ for a predicate family P_κ, then the scheme $\mathsf{Gen\text{-}Trans}(\Gamma_P)$ defined in Section 5.4 is a fully secure predicate encryption scheme for P_κ in dual system groups under the $(d_1, d_2 - 1)$-$\mathsf{q\text{-}ratio_{dsg}}$ assumption.*

When the above theorem is combined with Theorem 5.2 and Lemma 5.5, we get the following corollary:

Corollary 7.2. *If a pair encoding scheme satisfies (d_1, d_2)-$\mathsf{Sym\text{-}Prop}$ for a predicate family then there exists a fully secure predicate encryption scheme for that family in prime-order bilinear maps under the $(\mathsf{max}(d_1, d_2 - 1) + M_1 + 1, d_2 + W_1 + 1)$-$\mathsf{q\text{-}ratio}$ and k-linear assumptions, where M_1 and W_1 are bounds on the number of* key-enc *and* ct-enc *non-lone variables, respectively, in the encoding.*

The rest of this section is devoted to the proof of Theorem 7.1. We follow the same general outline as in other papers that use dual system groups [16,1,15]. The design of hybrids in our proof is closer to [16] and [15] rather than [1]. In particular, our hybrid structure is simpler because, unlike [1], we don't add noise to individual samples in every key. However, since we have adopted the generic transformation from [1], the indistinguishability between several hybrids follows from that of corresponding hybrids in [1]. (We briefly review these hybrids and the properties they follow from below—for full proofs see [1].) The main novelty in our proof, and the crucial difference from [1], is how the form of master secret key is changed: in [1] relaxed perfect security is used for this purpose, but we use the symbolic property in conjunction with the $\mathsf{q\text{-}ratio_{dsg}}$ assumption.

We first define auxiliary algorithms for encryption and key generation. Below we use $g_{i,0}$ (resp. $h_{i,0}$) to denote the first element of $\mathbf{g}_i$ (resp. $\mathbf{h}_i$). Also w and m denote $w_1 + w_2$ and $m_1 + m_2$, respectively.

– $\overline{\mathsf{Encrypt}}(\text{PP}, x, \mathsf{msg}; (\mathbf{g}'_0, \mathbf{g}'_1, \ldots, \mathbf{g}'_w), \text{MSK})$: This algorithm is same as $\mathsf{Encrypt}$ except that it uses $\mathbf{g}'_i \in \mathbb{G}^{n+1}$ instead of the samples $\mathbf{g}_i$ from SampG, and sets $\text{CT}^\star$ to $\mathsf{msg} \cdot e(g'_{0,0}, \text{MSK})$.

– $\overline{\mathsf{KeyGen}}(\mathrm{PP}, \mathrm{MSK}, y; (\mathbf{h}'_1, \ldots, \mathbf{h}'_m))$: This algorithm is same as KeyGen except that it uses $\mathbf{h}'_i \in \mathbb{H}^{n+1}$ instead of the samples $\mathbf{h}_i$ from SampH.

Using the algorithms described above, we define alternate forms for the ciphertext, master secret key, and secret keys.

– *Semi-functional master secret key* is defined to be $\overline{\mathrm{MSK}} := \mathrm{MSK} \cdot \tilde{h}^{\mu}$ where $\mu \leftarrow_R \mathbb{Z}_N$.
– *Semi-functional ciphertext* is given by $\overline{\mathsf{Encrypt}}(\mathrm{PP}, x, m; \mathbf{G} \cdot \hat{\mathbf{G}}, \mathrm{MSK})$, where $\mathbf{G} \cdot \hat{\mathbf{G}}$ is defined as follows: sample $\mathbf{g}_1, \ldots, \mathbf{g}_w$ from SampG and $\hat{\mathbf{g}}_1, \ldots, \hat{\mathbf{g}}_w$ from $\overline{\mathsf{SampG}}$ (which also requires SP); set $\mathbf{G}$ and $\mathbf{G}'$ to be the vector of vectors $(\mathbf{g}_1, \ldots, \mathbf{g}_w)$ and $(\hat{\mathbf{g}}_1, \ldots, \hat{\mathbf{g}}_w)$, respectively; and denote $(\mathbf{g}_1 \cdot \hat{\mathbf{g}}_1, \ldots, \mathbf{g}_w \cdot \hat{\mathbf{g}}_w)$ by $\mathbf{G} \cdot \hat{\mathbf{G}}$.
– *Ext-semi-functional ciphertext* is given by $\overline{\mathsf{Encrypt}}(\mathrm{PP}, x, m; \mathbf{G} \cdot \hat{\mathbf{G}} \cdot \hat{\mathbf{G}}', \mathrm{MSK})$, where $\mathbf{G}$, $\hat{\mathbf{G}}$ are as above, and $\hat{\mathbf{G}}'$ is defined to be $(\hat{\mathbf{g}}'_1, \ldots, \hat{\mathbf{g}}'_w)$, where $\hat{\mathbf{g}}'_i = (1, \hat{g}_{i,0}^{\gamma_1}, \ldots, \hat{g}_{i,0}^{\gamma_n})$ for $i \in [w]$ and $\gamma_1, \ldots, \gamma_n \leftarrow_R \mathbb{Z}_N$. (Here these $\gamma_1, \ldots, \gamma_n$ will be chosen once and used in both ciphertext and key components.)
– Table 1 lists the different types of keys we need and the inputs that should to be passed to $\overline{\mathsf{KeyGen}}$ (besides PP and y) in order to generate them. In the table, $\mathbf{h}_1, \ldots, \mathbf{h}_m$ are samples from SampH; $\hat{\mathbf{h}}_1, \ldots, \hat{\mathbf{h}}_m$ are samples from $\overline{\mathsf{SampH}}$ (which also requires SP); and $\hat{\mathbf{h}}'_i = (1, \hat{h}_{i,0}^{\gamma_1}, \ldots, \hat{h}_{i,0}^{\gamma_n})$ for $i \in [m]$, where $\gamma_1, \ldots, \gamma_n$ are the values described above for the ext-semi-functional ciphertext.

Type of key	*Inputs to* $\overline{\mathsf{KeyGen}}$ *(besides* PP *and* y*)*
Normal	$\mathrm{MSK}; (\mathbf{h}_1, \ldots, \mathbf{h}_m)$
Pseudo-normal	$\mathrm{MSK}; (\mathbf{h}_1 \cdot \hat{\mathbf{h}}_1, \ldots, \mathbf{h}_m \cdot \hat{\mathbf{h}}_m)$
Ext-pseudo-normal	$\mathrm{MSK}; (\mathbf{h}_1 \cdot \hat{\mathbf{h}}_1 \cdot \hat{\mathbf{h}}'_1, \ldots, \mathbf{h}_m \cdot \hat{\mathbf{h}}_m \cdot \hat{\mathbf{h}}'_m)$
Ext-pseudo-semi-functional	$\overline{\mathrm{MSK}}; (\mathbf{h}_1 \cdot \hat{\mathbf{h}}_1 \cdot \hat{\mathbf{h}}'_1, \ldots, \mathbf{h}_m \cdot \hat{\mathbf{h}}_m \cdot \hat{\mathbf{h}}'_m)$
Pseudo-semi-functional	$\overline{\mathrm{MSK}}; (\mathbf{h}_1 \cdot \hat{\mathbf{h}}_1, \ldots, \mathbf{h}_m \cdot \hat{\mathbf{h}}_m)$
Semi-functional	$\overline{\mathrm{MSK}}; (\mathbf{h}_1, \ldots, \mathbf{h}_m)$

Table 1. Six types of keys.

Let ξ denote the number of key queries made by the adversary. In Table 2, we give an outline of the proof-structure with the first column stating the various hybrids we have ($\varphi \in [\xi]$), second column describes the way in which a hybrid differs from the one in the previous row, and the third column lists the properties we need to show indistinguishability from the previous one. To prevent the table from overflowing, we use some shorthands like ct for ciphertext, func for functional, norm for normal, msg for message, and ind for indistinguishability. Also, Hyb_0 is the game $\mathsf{IND\text{-}CPA}^b_{\mathcal{A}}(\lambda, \mathsf{par})$ which is formally defined in Section 2.1. See the full version for a more formal description of the hybrids.

Hybrid	*Difference from previous*	*Properties required*
Hyb_0	-	-
Hyb_1	ct semi-func	left subgroup ind
$\vdots$	$\vdots$	$\vdots$
$\mathsf{Hyb}_{2,\varphi-1,5}$	$\varphi - 1$ keys semi-func	-
$\mathsf{Hyb}_{2,\varphi,1}$	φth key pseudo-norm	right subgroup ind
$\mathsf{Hyb}_{2,\varphi,2}$	ct ext-semi-func, φth key ext-pseudo-norm	parameter hiding
$\mathsf{Hyb}_{2,\varphi,3}$	φth key ext-pseudo-semi-func	non-degeneracy, $\mathsf{Sym\text{-}Prop}^\star$, $\mathsf{q\text{-}ratio_{dsg}}$ assumption
$\mathsf{Hyb}_{2,\varphi,4}$	ct semi-func, φth key pseudo-semi-func	parameter-hiding
$\mathsf{Hyb}_{2,\varphi,5}$	φth key semi-func	right subgroup ind
$\vdots$	$\vdots$	$\vdots$
$\mathsf{Hyb}_{2,\xi,5}$	All keys semi-func	-
Hyb_3	ct semi-func encryption of random msg	projective, orthogonality, non-degeneracy

Table 2. An outline of the proof structure.

Our main concern here is the indistinguishability of hybrids $\mathsf{Hyb}_{2,\varphi,2}$ and $\mathsf{Hyb}_{2,\varphi,3}$ when the φth key changes from ext-pseudo-normal to ext-pseudo semi-functional, while the ciphertext stays ext-semi-functional. (Indistinguishability of the rest of the hybrids follows from [1] as noted earlier.) We prove the following lemma in the full version.

Lemma 7.3. *For any* PPT *adversary* $\mathcal{A}$*, there exists a* PPT *adversary* $\mathcal{B}$ *such that the advantage of* $\mathcal{A}$ *in distinguishing* $\mathsf{Hyb}_{2,\varphi,2}$ *and* $\mathsf{Hyb}_{2,\varphi,3}$ *is at most the advantage of* $\mathcal{B}$ *in the* $\mathsf{q\text{-}ratio_{dsg}}$ *assumption plus some negligible quantity in the security parameter.*

References

1. Agrawal, S., Chase, M.: A study of pair encodings: Predicate encryption in prime order groups. In: TCC 2016-A. pp. 259–288 (2016)
2. Attrapadung, N.: Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In: EUROCRYPT. pp. 557–577 (2014)
3. Attrapadung, N.: Dual system encryption via doubly selective security: Framework, fully-secure functional encryption for regular languages, and more. Cryptology ePrint Archive, Report 2014/428 (2014), http://eprint.iacr.org/2014/428
4. Attrapadung, N.: Dual system encryption framework in prime-order groups via computational pair encodings. In: ASIACRYPT 2016. pp. 591–623 (2016)
5. Attrapadung, N., Hanaoka, G., Yamada, S.: Conversions among several classes of predicate encryption and applications to ABE with various compactness tradeoffs. In: ASIACRYPT 2015. pp. 575–601 (2015)
6. Attrapadung, N., Libert, B.: Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation. In: PKC 2010. pp. 384–402 (2010)

7. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: PKC 2011. pp. 90–108 (2011)
8. Attrapadung, N., Yamada, S.: Duality in ABE: Converting attribute based encryption for dual predicate and dual policy via computational encodings. In: CT-RSA. pp. 87–105 (2015)
9. Beimel, A.: Secret-sharing schemes: A survey. In: Chee, Y., Guo, Z., Ling, S., Shao, F., Tang, Y., Wang, H., Xing, C. (eds.) Coding and Cryptology, Lecture Notes in Computer Science, vol. 6639, pp. 11–46. Springer Berlin Heidelberg (2011)
10. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy. pp. 321–334 (2007)
11. Boneh, D., Raghunathan, A., Segev, G.: Function-private identity-based encryption: Hiding the function in functional encryption. In: CRYPTO. pp. 461–478 (2013)
12. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: TCC. pp. 535–554 (2007)
13. Boyen, X.: Attribute-based functional encryption on lattices. In: TCC. pp. 122–142 (2013)
14. Chase, M.: Multi-authority attribute based encryption. In: TCC. pp. 515–534 (2007)
15. Chen, J., Gay, R., Wee, H.: Improved dual system ABE in prime-order groups via predicate encodings. In: EUROCRYPT 2015. pp. 595–624 (2015)
16. Chen, J., Wee, H.: Dual system groups and its applications — compact HIBE and more. Cryptology ePrint Archive, Report 2014/265 (2014), http://eprint.iacr.org/2014/265
17. Chen, J., Wee, H.: Semi-adaptive attribute-based encryption and improved delegation for Boolean formula. In: SCN. pp. 277–297 (2014)
18. Freeman, D.M.: Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In: EUROCRYPT. pp. 44–61 (2010)
19. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS. pp. 40–49 (2013)
20. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Functional encryption without obfuscation. In: TCC 2016-A. pp. 480–511 (2016)
21. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: ACM STOC. pp. 545–554 (2013)
22. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM CCS. pp. 89–98 (2006), available as Cryptology ePrint Archive Report 2006/309
23. Guillevic, A.: Comparing the pairing efficiency over composite-order and prime-order elliptic curves. In: ACNS. pp. 357–372 (2013)
24. Herold, G., Hesse, J., Hofheinz, D., Ràfols, C., Rupp, A.: Polynomial spaces: A new framework for composite-to-prime-order transformations. In: CRYPTO. pp. 261–279 (2014)
25. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: EUROCRYPT. pp. 146–162 (2008)
26. Kowalczyk, L., Lewko, A.B.: Bilinear entropy expansion from the decisional linear assumption. In: CRYPTO. pp. 524–541 (2015)
27. Lewko, A.B.: Tools for simulating features of composite order bilinear groups in the prime order setting. In: EUROCRYPT. pp. 318–335 (2012)
28. Lewko, A.B., Waters, B.: Decentralizing attribute-based encryption. In: EUROCRYPT. pp. 568–588 (2011)

29. Lewko, A.B., Waters, B.: Unbounded HIBE and attribute-based encryption. In: EUROCRYPT. pp. 547–567 (2011)
30. Lewko, A.B., Waters, B.: New proof methods for attribute-based encryption: Achieving full security through selective techniques. In: CRYPTO. pp. 180–198 (2012)
31. Okamoto, T., Takashima, K.: Fully secure unbounded inner-product and attribute-based encryption. In: ASIACRYPT. pp. 349–366 (2012)
32. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: ACM CCS. pp. 195–203 (2007)
33. Rouselakis, Y., Waters, B.: Practical constructions and new proof methods for large universe attribute-based encryption. In: ACM CCS. pp. 463–474 (2013)
34. Sahai, A., Seyalioglu, H., Waters, B.: Dynamic credentials and ciphertext delegation for attribute-based encryption. In: CRYPTO. pp. 199–217 (2012)
35. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: EUROCRYPT. pp. 457–473 (2005)
36. Shen, E., Shi, E., Waters, B.: Predicate privacy in encryption systems. In: TCC. pp. 457–473 (2009)
37. Waters, B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In: CRYPTO. pp. 619–636 (2009)
38. Waters, B.: Functional encryption for regular languages. In: CRYPTO. pp. 218–235 (2012)
39. Waters, B.: A punctured programming approach to adaptively secure functional encryption. In: CRYPTO. pp. 678–697 (2015)
40. Wee, H.: Dual system encryption via predicate encodings. In: TCC. pp. 616–637 (2014)
41. Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.: A framework and compact constructions for non-monotonic attribute-based encryption. In: PKC 2014. pp. 275–292 (2014)