Research Article

Mridul Nandi and Tapas Pandit*

Predicate signatures from pair encodings via dual system proof technique

https://doi.org/10.1515/jmc-2017-0007
Received February 27, 2017; revised April 16, 2019; accepted May 21, 2019

Abstract: Recently, Attrapadung (Eurocrypt 2014) proposed a generic framework for fully (adaptively) secure predicate encryption (PE) based on a new primitive, called *pair encodings*. The author shows that if the underlying pair encoding scheme is either perfectly secure or computationally (doubly-selectively) secure, then the PE scheme will be fully secure. Although the pair encodings were solely introduced for PE, we show that these can also be used to construct predicate signatures, a signature analogue of PE. More precisely, we propose a generic construction of predicate signature (PS) from pair encoding schemes. Our construction provides unconditional signer privacy, and unforgeability in the adaptive model. Thereafter, we instantiate many PS schemes with new results, e.g., the first practical PS schemes for regular languages, the first attribute-based signature (ABS) scheme with constant-size signatures in the adaptive model, unbounded ABS with large universes in key-policy flavor, etc.

Keywords: Pair encodings, predicate signatures, perfect privacy, adaptive unforgeability

MSC 2010: 11T71, 94A60, 94A62, 14G50

Communicated by: Carlo Blundo

1 Introduction

The dual system methodology of Waters [37] is a well-known tool for constructing adaptively secure predicate encryption schemes. But, for some predicates, e.g., regular languages, the adaptively secure predicate encryption was not known, even though their selectively secure version was available. Therefore, for those classes of predicates, the dual system technique of Waters [37] was unreachable. Recently, Attrapadung [1] introduced a new primitive, called pair encoding schemes, which are implicitly contained in many predicate encryption schemes. Using pair encodings, the author proposed a generic framework [1] for adaptively secure predicate encryption, which captures the core technique of the dual system methodology [37]. He showed that, by applying the generic approach on pair encodings, adaptively secure PE is possible. Their conversion assumes either the perfect security or computational (doubly-selective) security of the underlying pair encoding scheme. Using this framework, the author constructed the first fully secure predicate encryption schemes for which only selectively secure schemes were known. He instantiated some surprising results, e.g., PE for regular languages, unbounded ABE for large universes, ABE with constant-size ciphertexts, etc. Concurrently and independently, Wee [39] proposed the notion of predicate encodings, which is exactly identical to the perfectly secure pair encodings of [1]. Some of the instantiations in [39] are similar to [1], viz., ABE for small universes with improved efficiency and doubly spatial encryption. Later, Attrapadung and Yamada [5] showed a conversion for obtaining the dual of a computationally secure pairing encoding

***Corresponding author: Tapas Pandit,** Department of CSA, Indian Institute of Science, Bangalore, India, e-mail: tapasgmmath@gmail.com
Mridul Nandi, Applied Statistics Unit, Indian Statistical Institute, Kolkata, India, e-mail: mridul@isical.ac.in

scheme. The authors considered this conversion to construct the dual of a predicate encryption scheme based on a computational pair encoding scheme.

Predicate signature (PS) [3] is a signature analogue of predicate encryption (PE), where Alice signs a document under an associated data index (policy), provided Alice's key index $x \in \mathcal{X}$ is related to the associated data index $y \in \mathcal{Y}$. The term "related" is defined by a binary relation $\sim$, called predicate defined over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are respectively called key space and associated data space. Sometimes, we call the tuple $(\sim, \mathcal{X}, \mathcal{Y})$ predicate tuple. Attribute-based signature (ABS) [28] is a larger subclass of PS. Like ABS, predicate signature schemes are available in two forms, key-policy predicate signature (KP-PS) and signature-policy predicate signature (SP-PS). If the contents of $\mathcal{X}$ have more complex representations than the contents of $\mathcal{Y}$, then the predicate signature is called KP-PS; otherwise, it is SP-PS. Similar to ABS, we have two types of security, unforgeability and signer privacy. The former ensures that signatures are generated by a valid user, and the latter protects from revealing the key index of the signer.

Motivation. The available pair encoding schemes of [1, 5, 39] have been reached out to most of the practical predicate families. Therefore, it is interesting to see a framework of predicate signatures from pair encoding schemes which were solely introduced for predicate encryptions.

Question. *Is it possible to construct a generic predicate signature scheme from pair encoding scheme, and at the same time, it enjoys all the features analogous to those of* [1]*?*

Our Result. Affirmatively, we answer the above question. That is, we provide a generic construction of predicate signature schemes from pair encoding schemes. If the underlying pair encoding scheme has a least security[1] and fulfills some natural conditions, then the PS scheme will achieve unconditional signer privacy, and unforgeability in the adaptive model. The construction is given in the setting of composite order bilinear groups. The unforgeability of the proposed construction is proven under three subgroup assumptions DSG1, DSG2, DSG3, and extra hardness assumption(s) required for the CMH-security of the underlying pair encoding scheme. If the primitive pair encoding scheme has PMH-security, then we do not need any extra hardness assumption. In this case, we say that the corresponding PS scheme is *cost free*. Through this generic construction, what we achieved is summarized below.

All the pair encoding schemes of [1, 5, 39] possess the least security and satisfy the natural conditions (see Conditions 3.1 of Section 3.1). Therefore, the resultant predicate signature schemes are adaptively unforgeable and perfectly private. Our generic predicate signature can be used to derive the following new results (see Table 2 in Section 5).

- *PS for regular languages*. Predicate signature schemes for regular languages in both the forms, key policy and signature policy, are provided in this paper. Both the schemes support a large universe alphabet. To the best of our knowledge, these are the first practical constructions of predicate signature schemes beyond ABS.
- *Unbounded KP-ABS*. We present an unbounded KP-ABS scheme with large universes, where the size of the universe is super-polynomial and no restriction has been imposed on the access polices and sets of attributes. To the best of our knowledge, this is the first large universes KP-ABS construction with the feature *unbounded*.
- *Constant-size signatures and constant-size keys*. Till date, the only available ABS scheme [3] with constant-size signatures for general access structures is known to be unforgeable in the selective model. We propose the first KP-ABS with constant-size signature, where unforgeability is proven in the adaptive model. A dual version, SP-ABS with constant-size keys, is also provided in this paper.
- *Cost free signatures*. The following instantiations of predicate signature are cost free as the underlying pair encoding schemes are PMH secure.

1 We consider two notions of security [1] for pair encoding scheme, perfect and computational. The perfect security is called perfectly master-key hiding (PMH). The computational security is of two types, selectively master-key hiding (SMH) and co-selectively master-key hiding (CMH). By least security, we mean either PMH or CMH.

(1) ABS for large universes,
(2) predicate signature scheme for policy over doubly spatial predicate,
(3) predicate signature schemes with constant-size keys and constant-size signatures, respectively, for both zero inner product and non-zero inner product predicates,
(4) predicate signature schemes for doubly spatial predicate and negated spatial predicate,
(5) spatial signature schemes with constant-size keys and constant-size signatures, respectively.

Outline of our construction. Let $(N := p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e)$ denote composite order bilinear groups, where $e\colon \mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$ is a bilinear map and $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N. Note that $\mathbb{G} = \mathbb{G}_{p_1}\times\mathbb{G}_{p_2}\times\mathbb{G}_{p_3}$, where $\mathbb{G}_{p_i}$ is subgroup of $\mathbb{G}$ of order p_i. Let $g_T := e(g,g)$, where $g\in\mathbb{G}_{p_1}$. For $\boldsymbol{X},\boldsymbol{Y}\in\mathbb{G}^n$, the notation $\boldsymbol{X}\cdot\boldsymbol{Y}$ represents the pairwise group operations, and therefore $\boldsymbol{X}\cdot\boldsymbol{Y}\in\mathbb{G}^n$. The notation $e(\boldsymbol{X},\boldsymbol{Y})$ stands for $\prod_{i=1}^n e(X_i,Y_i)$.

In brief, a pair encoding scheme [1] consists of four deterministic algorithms, Param, Enc1, Enc2 and Pair. Let $N\in\mathbb{N}$.

- $\mathsf{Param}(\boldsymbol{j})\to n$, where $\boldsymbol{j}$ is the index for the system parameter and n describes the length of the common parameters $\boldsymbol{h}\in\mathbb{Z}_N^n$,
- $\mathsf{Enc1}(x)\to(\boldsymbol{k}_x, m_2)$, where $\boldsymbol{k}_x$ is a sequence of polynomials over $\mathbb{Z}_N$ with $|\boldsymbol{k}_x| = m_1$ and m_2 is the length of the random coin $\boldsymbol{r}\in\mathbb{Z}_N^{m_2}$,
- $\mathsf{Enc2}(y)\to(\boldsymbol{c}_y,\omega_2)$, where $\boldsymbol{c}_y$ is a sequence of polynomials over $\mathbb{Z}_N$ with $|\boldsymbol{c}_y| = \omega_1$ and ω_2+1 is the length of the random coin $\boldsymbol{s} = (s_0,\ldots,s_{\omega_2})\in\mathbb{Z}_N^{\omega_2+1}$,
- $\mathsf{Pair}(x,y)\to\boldsymbol{E}\in\mathbb{Z}_N^{m_1\times\omega_1}$.

The correctness says that, for $x\sim y$, $(\boldsymbol{k}_x,m_2)\leftarrow\mathsf{Enc1}(x)$, $(\boldsymbol{c}_y,\omega_2)\leftarrow\mathsf{Enc2}(y)$ and $\boldsymbol{E}\leftarrow\mathsf{Pair}(x,y)$, we have

$$\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}\boldsymbol{c}_y^\top(\boldsymbol{s},\boldsymbol{h}) = \alpha s_0.$$

Before describing the outline of our predicate signature, we state the following two facts:

Fact 1. A signature is nothing but a diluted key for a policy y computed from an actual (strong) key $\mathcal{SK}_x$ with $x\sim y$, where the message m and the policy y are to be committed.

Fact 2. To maintain signer privacy, the signature is to be labeled with policy y, at least not labeled with the key index x.

The above facts are implicitly used in many predicate signatures and also provide insight to predicate signature. In the following, we first give an outline of the initial structure of our predicate signature using the structure of predicate encryption of [1] based on pair encodings. Recall that $\mathcal{SK}_x = g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})}\cdot\boldsymbol{R}_3\in\mathbb{G}^{m_1}$ with $\boldsymbol{R}_3\in\mathbb{G}_{p_3}^{m_1}$ is the key structure of [1] for the key index x.

- *Signature generation.* Suppose Alice is playing the role of a sender. Let $\mathcal{SK}_x = g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})}\cdot\boldsymbol{R}_3\in\mathbb{G}^{m_1}$ be the key of Alice. To sign a message m under a policy y with $x\sim y$, Alice first runs $\boldsymbol{E}\leftarrow\mathsf{Pair}(x,y)$. Then it generates the signature as $\boldsymbol{\delta}_y := \mathcal{SK}_x^{\boldsymbol{E}}\cdot\boldsymbol{R}_3'\in\mathbb{G}^{\omega_1}$, where $\boldsymbol{R}_3'\xleftarrow{\cup}\mathbb{G}_{p_3}^{\omega_1}$. On simplification, we have $\boldsymbol{\delta}_y = g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}}\cdot\widetilde{\boldsymbol{R}_3}$, where $\widetilde{\boldsymbol{R}_3} := \boldsymbol{R}_3^{\boldsymbol{E}}\cdot\boldsymbol{R}_3'$. The signature $\boldsymbol{\delta}_y$ plays the role of a diluted key, derived from the actual key $\mathcal{SK}_x$.
- *Signature verification.* The verification process considered here is a probabilistic one as it is performed by running some routines which are similar to the encryption and decryption of the predicate encryption of [1]. Since a signature is a poor or diluted key, verifying a signature is nothing but checking its capability to extract out some information from the part of a ciphertext. Therefore, to verify a signature $\boldsymbol{\delta}_y$, we first prepare a verification text (that is same as the ciphertext, but without the message m) $\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := g_T^{\alpha s_0}, \mathcal{V}_y := g^{\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h})})$. The signature is accepted if $e(\boldsymbol{\delta}_y,\mathcal{V}_y) = \mathcal{V}_{\mathrm{INT}}$, else rejected. We note that the $\mathbb{G}_{p_3}$ part of $\boldsymbol{\delta}_y$ gets canceled in the verification due to the orthogonal property of composite order bilinear groups.
- *Correctness.* For $x\sim y$, we have $e(\boldsymbol{\delta}_y,\mathcal{V}_y) = e(g,g)^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}\boldsymbol{c}_y^\top(\boldsymbol{s},\boldsymbol{h})} = g_T^{\alpha s_0}$, where the last equality is obtained from the correctness of the pair encoding scheme.

Limitations of the initial structure of the signature. The above initial structure of the signature only shows that Alice is capable to generate such a signature. We note that neither the message nor the policy is committed to the above signature, and this is very crucial to guarantee unforgeability. Although the above signature

is not labeled with the key index x, it misses a very important property of predicate signature, perfect privacy of the signer.

To overcome the limitations of the above signature, we have to modify the initial structure. The modifications are explained briefly in the following two steps.

Step 1. The initial structure of the signature is $\boldsymbol{\delta}_y = g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}} \cdot \widetilde{\boldsymbol{R}_3} \in \mathbb{G}^{\omega_1}$. To ensure unforgeability, the message m and the policy y are to be committed to $\boldsymbol{\delta}_y$. The binding is to be done in such a way that the binding part of the signature cannot be updated once the signature has been generated. The binding is made in the following way. A collision resistance hash function $H\colon \{0,1\}^* \to \mathbb{Z}_N$ and two others parameters $g^{\theta_1}, g^{\theta_2}$ are added to the public parameters $\mathcal{PP}$. A group element $g^{\tau(\theta_1\hbar+\theta_2)}$ is composed with the first component of $g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}}$, where $\tau \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and $\hbar = H(m, y)$. Additionally, $g^{-\tau}$ is given as a part of the signature. In other words, the modified signature becomes $\boldsymbol{\delta}_y = g^{\boldsymbol{v}} \cdot \widetilde{\boldsymbol{R}_3} \in \mathbb{G}^{\omega_1+1}$, where $\boldsymbol{v}$ is implicitly set to $\boldsymbol{v} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}) \in \mathbb{Z}_N^{\omega_1+1}$ and $\boldsymbol{\psi} := (\tau(\theta_1\hbar + \theta_2), 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$.

To verify this signature, the verification text is to be changed to $\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := g_T^{\alpha s_0}, \boldsymbol{\mathcal{V}}_y := g^{\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s},\theta_1,\theta_2,\boldsymbol{h})})$, where

$$\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \theta_1, \theta_2, \boldsymbol{h}) := (c_0(s_0, \theta_1, \theta_2, \hbar), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{Z}_N^{\omega_1+1},$$
$$c_0(s_0, \theta_1, \theta_2, \hbar) := s_0(\theta_1\hbar + \theta_2) \quad \text{and} \quad \hbar := H(m, y).$$

The verification is the same as before, i.e., the signature is accepted if $e(\boldsymbol{\delta}_y, \boldsymbol{\mathcal{V}}_y) = \mathcal{V}_{\mathrm{INT}}$, else rejected. For correctness of the verification, we assume that $c_{y,\iota}(\boldsymbol{s}, \boldsymbol{h}) = s_0$ for some $\iota \in [\omega_1]$.

Step 2. For perfect privacy, the authors of [29] assume the perfectly hiding property of the underlying non-interacting witness-indistinguishable (NIWI) scheme. For the ABS schemes of [31–33], an additional secret sharing (0-sharing) was used to assure perfect privacy. For perfect privacy of the proposed signature, we explore a novel approach (for details, refer to Section 3.4) which works irrespective of the predicate families. This is done by uniformly sampling from the orthogonal space $(\mathbf{V}_{\mathrm{M}})^{\perp}$ of $\mathbf{V}_{\mathrm{M}} := \{\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \theta_1, \theta_2, \boldsymbol{h}) \in \mathbb{Z}_N^{\omega_1+1} \mid \boldsymbol{s} := (s_0, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}\}$. The final signature of the proposed construction (for a complete description, refer to Section 3.3) has the form $\boldsymbol{\delta}_y = g^{\boldsymbol{v}+\boldsymbol{v}_{\mathrm{sp}}} \cdot \widetilde{\boldsymbol{R}_3} \in \mathbb{G}^{\omega_1+1}$, where $\boldsymbol{v}_{\mathrm{sp}} \xleftarrow{\mathrm{U}} (\mathbf{V}_{\mathrm{M}})^{\perp}$. The verification is the same as before, where $\boldsymbol{v}_{\mathrm{sp}}$ gets canceled due to the orthogonality of $\boldsymbol{v}_{\mathrm{sp}}$ and $\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \theta_1, \theta_2, \boldsymbol{h})$.

We show that uniformly sampling from $(\mathbf{V}_{\mathrm{M}})^{\perp}$ is done by solving the homogeneous system $\boldsymbol{A}^{\top}\boldsymbol{X} = \boldsymbol{0}$, where $\boldsymbol{A} \in \mathbb{Z}_N^{\omega_1\times(\omega_2+1)}$. For $1 \le \iota \le \omega_1$ and $0 \le j \le \omega_2$, the (ι, j)-th entry of the matrix $\boldsymbol{A}$ is of the form $a_{\iota,j} + \sum_{i\in[n]} a_{\iota,j,i}h_i$, where $a_{\iota,j}$ and $a_{\iota,j,i}$ are coefficients of the ι-th polynomial of $\boldsymbol{c}_y$. The only available information to solve the system are $a_{\iota,j}$, $a_{\iota,j,i}$ and g^{h_i} for $1 \le \iota \le \omega_1$, $0 \le j \le \omega_2$ and $i \in [n]$. Since h_i are not given explicitly, applying Gaussian elimination on $\boldsymbol{A}$ is troublesome. Although h_i are not given explicitly, we manage to solve the system $\boldsymbol{A}^{\top}\boldsymbol{X} = \boldsymbol{0}$ perfectly. For that, we impose a restriction on the underlying pair encoding scheme, which is very natural. This restriction is given as condition (3) in Section 3.1. To the best of our knowledge, most of the pair encodings (in fact, all the pair encodings of [1, 5, 39]) satisfy this condition.

One of the motivations of this paper is to achieve adaptive security. We utilize the dual system proofs of [37] in a novel way to guarantee adaptive unforgeability of the proposed construction. For the proof of adaptive unforgeability of the proposed signature, we abstract out the dual system proof technique as a signature analogue of [1]. Hybrid arguments over the sequence of games considered in this signature analogue follow the style of [31, 33]. However, the hybrid arguments in [31, 33] were handled for a particular ABS through linear secret sharing scheme (LSSS). But here, we manage the dual system proof technique generically for arbitrary predicates. In this style, we consider semi-functional (mimic) forms of the original objects, viz., verification text, signatures and keys. Using hybrid arguments, we finally reach a game where $\mathcal{V}_{\mathrm{INT}}$ is chosen independently and uniformly at random from $\mathbb{G}_T$. This ensures that the forgery will be invalid with respect to the verification text $\mathcal{V}$.

Related works. In addition to the fully CPA-secure construction of PE, Attrapadung [1] showed a dual conversion for pair encodings. If the source pair encoding P is perfectly secure, then the dual of P, denoted by $\mathbb{D}(\mathsf{P})$ is also perfectly secure encoding. Using this conversion, full security of the dual of PE, denoted by $\mathbb{D}(\mathsf{PE})$, is guaranteed if the underlying pair encoding P has perfect security. However, there are many PE schemes for

which perfectly secure encodings were not known, so the fully secure realizations of their dual form were unsolved. Later, Attrapadung and Yamada [5] showed that the same dual conversion of [1] actually works for the computationally secure encodings. By applying this conversion on the underlying pair encoding of the previously proposed KP-ABE [1], the authors achieved the first fully secure unbounded CP-ABE and a CP-ABE with short keys for Boolean formulas. Recently, Chen, Gay and Wee [16] and Attrapadung [2] proposed new generic frameworks for achieving adaptively secure ABE in the prime order bilinear groups, which are nothing but the prime order version of [39] and [1], respectively. The main difference between the frameworks of [16] and [2] is that the former deals with only perfectly secure encodings, whereas the latter can deal with computationally secure encodings.

Attribute-based signature. In the literature, many ABS schemes [19, 27–29, 31, 32, 35, 36, 36, 40, 40] have been studied. Among them, only the schemes of [19, 29, 31] were known to achieve signer privacy, adaptive unforgeability in the standard model and support general access structures. In [29], the authors proposed a general framework for ABS using a credential bundle and a NIWI scheme as primitives. This general framework provides the attribute-based signatures for monotone span programs in signature-policy form. The authors showed two practical instantiations of ABS in the standard model using Groth–Sahai proof system [21] for satisfiability of pairing product equations. In the first instantiation, they used a Boneh–Boyen signature [9] as the candidate for a credential bundle, whereas, in the second instantiation, another Boneh–Boyen signature [8] was used. The ABS construction of [31] is based on the concept of the dual pairing vector space of [30] and relies on the DLIN assumption. The authors first utilized the dual system methodology [37] in ABS for adaptive unforgeability. The ABS of [31] is more efficient than the one of [29] since the latter uses the Groth–Sahai non-interactive zero-knowledge (NIZK) proof systems [21] as building blocks. Although the performance of the ABS construction [31] defeats that of [29], the scheme of [31] has the following drawbacks. The size of the public parameters is linear to the size of the sub-universe, and a bound is imposed on the number of times an attribute could appear in a policy. The ABS schemes of [19, 29, 31] have signature-policy form; among them, the schemes of [29, 31] support large universes.

Functional signature. Bellare and Fuchsbauer [6] proposed a notion of policy-based signature which unifies the existing signatures, e.g., group signatures [15], mess signatures [12], attribute-based signatures [29], etc. For a policy-based signature (PBS) scheme, the authors defined the policy language $\mathcal{L}$ to be any member of the complexity class **NP**. In this scheme, a key $\mathcal{SK}_p$ which is associated with policy p can sign a message m (without revealing p) if $(p, m) \in \mathcal{L}$. Since $\mathcal{L} \in \mathbf{NP}$, the message m together with the witness w is to be supplied while generating the signature. If we restrict the policy language to come from the complexity class **P** ($\subseteq$ **NP**), then what we have is nothing but the predicate signatures, where the witness is computed in polynomial time. At the same time, Boyle, Goldwasser and Ivan [13] introduced a concept of functional signatures. In this signature, a key $\mathcal{SK}_f$ is associated with a function f, and the key $\mathcal{SK}_f$ has the power to sign a message m if m belongs to its range. This can be considered as a special case of PBS, in which the policy language $\mathcal{L}$ is the set of all pairs (f, m) such that m is in the range of f and the witness for (f, m) is a pre-image m under f.

The authors in [6] showed a generic construction of attribute-based signature from PBS, but they did not explicitly mention the practical instantiation of ABS. If we instantiate the ABS of [6] using the Groth–Ostrovsky–Sahai proof system [20] for **NP**-complete languages such as circuit satisfiability, then there is a huge blowup in the size of the signature due to Karp reduction. On the other hand, if we use the Groth–Sahai proof system [21] for satisfiability of pairing product equations, then ABS supports only the restricted predicate family, viz., conjunction and disjunction of pairing product equations. Recently, Sakai, Attrapadung and Hanaoka [34] proposed an efficient ABS for arbitrary circuits from the symmetric external Diffie–Hellman assumption. Their ABS construction is based on the efficiency of the Groth–Sahai proof system [21] and the expressiveness of the Groth–Ostrovsky–Sahai proof system [20].

Organization. This paper is organized as follows. Basic notations, composite order bilinear groups, hardness assumptions, the syntaxes and security definitions of predicate signature and pair encoding schemes and other related things are given in Section 2. Framework, security and instantiations of predicate signature are respectively provided in Sections 3, 4 and 5.

2 Preliminaries

2.1 Notations

For a set X, $x \xleftarrow{R} X$ denotes that x is randomly picked from X according to the distribution R. Likewise, $x \xleftarrow{U} X$ indicates that x is uniformly selected from X. For an algorithm A and variables x, y, the notation $x \leftarrow A(y)$ (or $A(y) \rightarrow x$) carries the meaning that, when A is run on the input y, it outputs x. The symbol PPT stands for probabilistic polynomial-time. For $a, b \in \mathbb{N}$, define $[a, b] := \{i \in \mathbb{N} \mid a \leq i \leq b\}$ and $[b] := [1, b]$.

Throughout this paper, bold characters denote vector objects. For $\boldsymbol{h} \in \mathbb{Z}_N^n$ and $p \mid N$, we define

$$\boldsymbol{h} \bmod p := (h_1 \bmod p, \ldots, h_n \bmod p).$$

For a vector $\boldsymbol{x}$ (resp. $\boldsymbol{x}_k$), the i-th component is denoted by x_i (resp. x_{ki}). For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}_N^n$, we define

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle := \sum_{i=1}^{n} x_i \cdot y_i.$$

For $S \subset \mathbb{Z}_N^n$ and $\boldsymbol{\alpha} \in \mathbb{Z}_N^n$, we define $\boldsymbol{\alpha} + S := \{\boldsymbol{\alpha} + \boldsymbol{\beta} \mid \boldsymbol{\beta} \in S\}$.

For a matrix $\boldsymbol{M}$, the notations $\boldsymbol{M}^\top$ and M_{ij} denotes the transpose of $\boldsymbol{M}$ and an entry of $\boldsymbol{M}$ at the (i, j)-th position, respectively. The notation $\boldsymbol{M}_i$ denotes the i-th row of the matrix $\boldsymbol{M}$, and $\mathrm{Null}(\boldsymbol{M})$ represents the nullity of the matrix $\boldsymbol{M}$. The notation $\boldsymbol{0}_{m \times n}$ stands for an $m \times n$ matrix with all the entries as 0. For a group $\mathbb{G}$ and $n \in \mathbb{N}$, the entries from $\mathbb{G}^n$ are assumed to be the row vectors.

Let $\mathbb{G}$ be a cyclic group of order N with respect to the group operation "$\cdot$". For $g \in \mathbb{G}$ and $\boldsymbol{h} \in \mathbb{Z}_N^n$, we define $g^{\boldsymbol{h}} := (g^{h_1}, \ldots, g^{h_n})$. For $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{G}^n$, the notation $\boldsymbol{X} \cdot \boldsymbol{Y}$ stands for component-wise group operations, i.e., $\boldsymbol{X} \cdot \boldsymbol{Y} := (X_1 \cdot Y_1, \ldots, X_n \cdot Y_n) \in \mathbb{G}^n$. For $\boldsymbol{W} \in \mathbb{G}^n$ and $\boldsymbol{E} \in \mathbb{Z}_N^{n \times m}$, we define $\boldsymbol{W}^{\boldsymbol{E}} := \boldsymbol{z} \in \mathbb{G}^m$, where $z_i := W_1^{E_{1i}} \cdots W_n^{E_{ni}}$. If $\boldsymbol{W} = g^{\boldsymbol{w}}$, for $g \in \mathbb{G}$ and $\boldsymbol{w}^\top \in \mathbb{Z}_N^n$, then we can write $\boldsymbol{W}^{\boldsymbol{E}} = g^{\boldsymbol{w}\boldsymbol{E}}$.

For a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{\ell \times \vartheta}$, we define the linear space $\mathrm{Ker}(\boldsymbol{A}) := \{\boldsymbol{u} \in \mathbb{Z}_q^\ell \mid \boldsymbol{u}^\top \boldsymbol{A} = \boldsymbol{0}\}$. For $(\boldsymbol{X}, \boldsymbol{x}) \in \mathbb{Z}_q^{\ell \times \vartheta} \times \mathbb{Z}_q^\ell$, an affine space generated by $(\boldsymbol{X}, \boldsymbol{x})$ is defined by $\mathrm{Aff}(\boldsymbol{X}, \boldsymbol{x}) := \{\boldsymbol{X}\boldsymbol{u} + \boldsymbol{x} \mid \boldsymbol{u} \in \mathbb{Z}_q^\vartheta\} \subset \mathbb{Z}_q^\ell$. The nullity of a matrix $\boldsymbol{A}$ is defined by $\mathrm{Null}(\boldsymbol{A})$, which is the dimension of $\mathrm{Ker}(\boldsymbol{A}^\top)$.

2.2 Composite order bilinear groups

Composite order bilinear groups [10, 26] are defined to be a tuple $\mathcal{J} := (N := p_1 p_2 p_3, \mathbb{G}, \mathbb{G}_T, e)$, where p_1, p_2, p_3 are three distinct primes and $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N and $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a map with the following properties:

(1) *Bilinear.* For all $g, h \in \mathbb{G}$ and all $s, t \in \mathbb{Z}_p$, we have $e(g^s, h^t) = e(g, h)^{st}$.
(2) *Non-degenerate.* There exists an element $g \in \mathbb{G}$ such that $e(g, g)$ has order N in $\mathbb{G}_T$.
(3) *Computable.* There is an efficient algorithm for computing $e(g, h)$ for all $g, h \in \mathbb{G}$.

Let $\mathcal{G}_{\mathrm{cbg}}$ denote an algorithm which takes 1^κ as a security parameter and returns a description of composite order bilinear groups $\mathcal{J} = (N = p_1 p_2 p_3, \mathbb{G}, \mathbb{G}_T, e)$. Composite order bilinear groups enjoy the orthogonal property defined below.

Definition 2.1 (Orthogonal property). Let $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$ denote subgroups of $\mathbb{G}$ of order p_1, p_2 and p_3, respectively. The subgroups $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$ are said to have orthogonal property if, for all $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$ with $i, j \in \{1, 2, 3\}$ and $i \neq j$, it holds that $e(h_i, h_j) = 1$.

Additional notations. Let $1_{\mathbb{G}}$ and 1 denote the identity elements of $\mathbb{G}$ and $\mathbb{G}_T$, respectively. For $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{G}^n$, we define $e(\boldsymbol{X}, \boldsymbol{Y}) := \prod_{i=1}^n e(X_i, Y_i)$. For three distinct primes, p_1, p_2 and p_3, a cyclic group $\mathbb{G}$ of order $N = p_1 p_2 p_3$ can be written as $\mathbb{G} = \mathbb{G}_{p_1} \mathbb{G}_{p_2} \mathbb{G}_{p_3}$, where $\mathbb{G}_{p_i}$ are subgroups of $\mathbb{G}$ of order p_i. So each element $X \in \mathbb{G}$ can be expressed as $X = X_1 X_2 X_3$, where $X_i \in \mathbb{G}_{p_i}$. For $X \in \mathbb{G}$, the notation $X|_{\mathbb{G}_{p_i}}$ means the projection of X over $\mathbb{G}_{p_i}$, i.e., $X_i = X|_{\mathbb{G}_{p_i}}$. For $\boldsymbol{Y} \in \mathbb{G}^n$, let $\boldsymbol{Y}|_{\mathbb{G}_{p_i}}$ denote $(Y_1|_{\mathbb{G}_{p_i}}, \ldots, Y_n|_{\mathbb{G}_{p_i}})$. Let g_T stand for the element $e(g, g)$, where $g \in \mathbb{G}_{p_1}$.

2.3 Hardness assumptions in composite order bilinear groups

We describe here three decisional subgroup (DSG) assumptions [25] for 3 primes, DSG1, DSG2 and DSG3, in composite order bilinear groups. Let $\mathcal{J} := (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \xleftarrow{U} \mathcal{G}_{\mathrm{cbg}}(1^\kappa)$ be the common parameters for each assumptions. In the following, we define an instance for each assumption.

DSG1. Let $g \xleftarrow{U} \mathbb{G}_{p_1}, Z_3 \xleftarrow{U} \mathbb{G}_{p_3}, T_0 \xleftarrow{U} \mathbb{G}_{p_1}, T_1 \xleftarrow{U} \mathbb{G}_{p_1p_2}$. Define $\mathcal{D} := (\mathcal{J}, g, Z_3)$.

DSG2. Let $g, Z_1 \xleftarrow{U} \mathbb{G}_{p_1}, Z_2, W_2 \xleftarrow{U} \mathbb{G}_{p_2}, W_3, Z_3 \xleftarrow{U} \mathbb{G}_{p_3}, T_0 \xleftarrow{U} \mathbb{G}_{p_1p_3}, T_1 \xleftarrow{U} \mathbb{G}$. Define

$$\mathcal{D} := (\mathcal{J}, g, Z_1Z_2, W_2W_3, Z_3).$$

DSG3. Let $\alpha, s \xleftarrow{U} \mathbb{Z}_N, g \xleftarrow{U} \mathbb{G}_{p_1}, W_2, Y_2, g_2 \xleftarrow{U} \mathbb{G}_{p_2}, Z_3 \xleftarrow{U} \mathbb{G}_{p_3}, T_0 := e(g,g)^{\alpha s}, T_1 \xleftarrow{U} \mathbb{G}_T$. Define

$$\mathcal{D} := (\mathcal{J}, g, g^\alpha Y_2, g^s W_2, g_2, Z_3).$$

The advantage of an algorithm $\mathcal{A}$ in breaking DSGi, for $i = 1, 2, 3$ is defined by

$$\mathrm{Adv}^{\mathrm{DSG}i}_{\mathcal{A}}(\kappa) = |\Pr[\mathcal{A}(\mathcal{D}, T_0) = 1] - \Pr[\mathcal{A}(\mathcal{D}, T_1) = 1]|.$$

We say that the DSGi assumption holds in $\mathcal{J}$ if, for every PPT algorithm $\mathcal{A}$, the advantage $\mathrm{Adv}^{\mathrm{DSG}i}_{\mathcal{A}}(\kappa)$ is negligible in the security parameter κ.

2.4 Some results of linear algebra

We recall the three types of elementary row operations (for details, refer to [23]) on a matrix.

Type 1. Interchange rows i and j (in short, we write $R_i \leftrightarrow R_j$).

Type 2. Multiply row i by k, with $k \neq 0$ (in short, $R_i \leftarrow kR_i$).

Type 3. Add k-times row j to row i (in short, $R_i \leftarrow R_i + kR_j$).

Similarly, we can define three types of elementary column operations. Let $\mathcal{E}$ be a matrix obtained by applying a single elementary row operation on the identity matrix, called elementary matrix. Note that the effect of a single elementary row (resp. column) operation on a matrix $\boldsymbol{B}$ can also be obtained by pre- (resp. post-) multiplying the matrix $\boldsymbol{B}$ by the corresponding elementary matrix $\mathcal{E}$ (resp. $\mathcal{E}^\top$).

Definition 2.2. A matrix $\boldsymbol{M}$ is said to be row (resp. column) equivalent to a matrix $\boldsymbol{B}$ if $\boldsymbol{M}$ is obtained from $\boldsymbol{B}$ by applying a finite sequence of elementary row (resp. column) operations.

Definition 2.3. A non-zero row of a matrix R is said to be row-reduced if (1) the first non-zero entry of the row is equal to 1 (called leading 1) and (2) the column containing the leading 1 has all its other entries 0.

Definition 2.4. A matrix R is said to be row-reduced if each of its non-zero rows is row-reduced.

A well-known result that will be used very often is given below.

Theorem 2.1. *If two matrices $\boldsymbol{B}$ and $\boldsymbol{M}$ are row equivalent, then the systems $\boldsymbol{BX} = \boldsymbol{0}$ and $\boldsymbol{MX} = \boldsymbol{0}$ have the same solutions.*

But the scenario is slightly changed in case of column equivalence.

Theorem 2.2. *Suppose the matrix $\boldsymbol{M}$ is obtained from $\boldsymbol{B}$ by applying n elementary column operations, i.e., $\boldsymbol{B}\mathcal{E}_1^\top\mathcal{E}_2^\top\cdots\mathcal{E}_n^\top = \boldsymbol{M}$, where $\mathcal{E}_i$ are elementary matrices. Then $\boldsymbol{v}$ is a solution of the system $\boldsymbol{MX} = \boldsymbol{0}$ if and only if $\mathcal{E}_1^\top\mathcal{E}_2^\top\cdots\mathcal{E}_n^\top\boldsymbol{v}$ is a solution of $\boldsymbol{BX} = \boldsymbol{0}$.*

Theorem 2.3. *Let R be a ring with 1. Let $\boldsymbol{B} \in R^{m\times n}$ be a matrix such that, for $i \in [m]$, $B_{i1} = 1$ if $i = 1$, else 0. For $t \in R$, define $\tilde{\boldsymbol{t}} := (t, 0, \ldots, 0)^\top \in R^{m\times 1}$, and let $\boldsymbol{B}_{\mathrm{M}} := [\tilde{\boldsymbol{t}} : \boldsymbol{B}] \in R^{m\times(n+1)}$ be the augmented matrix. Then $(v_1, \ldots, v_n)^\top$ is a solution of $\boldsymbol{BX} = \boldsymbol{0}$ if and only if, for each $v_0 \in R$, $(v_0, -tv_0 + v_1, v_2, \ldots, v_n)^\top$ is a solution of the system $\boldsymbol{B}_{\mathrm{M}}\boldsymbol{X} = \boldsymbol{0}$.*

Proof. The proof is straightforward. □

Remark 2.1. From the above theorem, we have $\mathrm{Null}(\boldsymbol{B}_{\mathrm{M}}) = \mathrm{Null}(\boldsymbol{B}) + 1$.

Assumption: The factorization problem is intractable. For our purpose, we mainly apply the elementary row operations of type 2 and type 3. However, for simple representation of the solutions, one may use elementary row and column operations of type 1. Theorems 2.1 and 2.2 assume the fact that $k \neq 0$ (involved in type 2 operation) which implies that k is invertible. When matrices are considered over a field, then $k \neq 0$ implies that k is invertible. But if the matrices are not defined over the underlying field, then we may be in trouble. Here we consider the matrix $\boldsymbol{A}$ over $\mathbb{Z}_N$ with $N = p_1p_2p_3$, which is not a field. Since we assume that the factorization problem is intractable, perhaps it can help out from the said trouble. Let $0 \neq k \in \mathbb{Z}_N$. It is sufficient to show that k is co-prime to N. If k is not a co-prime to N, then we can establish an algorithm for breaking the factorization problem in polynomial time of the parameter κ. In fact, $\gcd(k, N)$ is a non-trivial factor of N, which is a contradiction.

2.5 Predicate family

To define a predicate-based cryptosystem, we have to define a predicate family. The predicate family is defined for an index set Λ. For most of the predicate families, the index sets are considered to be subsets of $\{\boldsymbol{j} : \boldsymbol{j} \in \mathbb{N}^i \text{ and } i \in \mathbb{N}\}$. The following definition of a predicate family is adopted from [1, 7].

Definition 2.5 (Predicate family). We define the predicate family to be $\sim := \{\sim_{\boldsymbol{j}}\}_{\boldsymbol{j}\in\Lambda}$ for an arbitrary index set Λ, where $\sim_{\boldsymbol{j}} : \mathcal{X}_{\boldsymbol{j}} \times \mathcal{Y}_{\boldsymbol{j}} \to \{0, 1\}$ is an indicator function, and $\mathcal{X}_{\boldsymbol{j}}$ and $\mathcal{Y}_{\boldsymbol{j}}$ are respectively called key space and associative data space.

The function $\sim_{\boldsymbol{j}}$ is also called predicate or binary relation over $\mathcal{X}_{\boldsymbol{j}} \times \mathcal{Y}_{\boldsymbol{j}}$. For $(x, y) \in \mathcal{X}_{\boldsymbol{j}} \times \mathcal{Y}_{\boldsymbol{j}}$, we write $x \sim_{\boldsymbol{j}} y$ if $\sim_{\boldsymbol{j}}(x, y) = 1$, else $x \not\sim_{\boldsymbol{j}} y$. For a predicate family, the corresponding index set Λ is called system-index space. A member $\boldsymbol{j}$ of the index space Λ is called index for the system parameter or simply system index. To design a predicate-based scheme for some predicate family, first a system index $\boldsymbol{j}$ is fixed for that family. Then this index will define a predicate tuple $(\sim_{\boldsymbol{j}}, \mathcal{X}_{\boldsymbol{j}}, \mathcal{Y}_{\boldsymbol{j}})$ for the corresponding predicate-based scheme. For example, the system indices for predicate families, regular languages, circuits, access structures, inner product and doubly spatial relation are respectively alphabet, maximum depth and number variables for circuits, attribute universe or size of the attribute universe, length of vectors and dimension of affine space.

In the current study, there are many predicate families which are used to provide access control over data. In the following, we describe some of the predicates. Note that, for most of the relations described below, the system indices are not given explicitly as it will be understood from the context.

- *Equality relation.* Let $\mathcal{X} = \mathcal{Y} = \{0, 1\}^*$. For $x, y \in \{0, 1\}^*$, we define $x \sim y$ if and only if $x = y$. The well-known predicate encryption for the equality relation is called identity-based encryption (IBE).
- *Inner product relation.* Let $\mathcal{X} = \mathcal{Y} = \mathbb{Z}_q^\ell$. For $x = (x_1, \ldots, x_\ell) \in \mathcal{X}$ and $y = (y_1, \ldots, y_\ell) \in \mathcal{Y}$, we define $x \sim y$ if and only if $\langle x, y\rangle = 0$. This relation is called zero inner product relation. Similarly, a non-zero inner product relation is defined by $x \sim y$ if and only if $\langle x, y\rangle \neq 0$. The corresponding encryption schemes are known as inner-product encryption (IPE).
- *(Doubly) spatial relation.* $\mathcal{X} = \mathcal{Y} := \{\mathrm{Aff}(\boldsymbol{A}, \boldsymbol{a}) \mid (\boldsymbol{A}, \boldsymbol{a}) \in \mathbb{Z}_q^{\ell\times k} \times \mathbb{Z}_q^\ell,\ 0 \leq k \leq \ell\}$. For $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, a doubly spatial relation is defined by $x \sim^{\mathrm{ds}} y$ if and only if $y \cap x \neq \emptyset$. For the spatial relation, we restrict $\mathcal{Y}$ to be $\mathbb{Z}_q^\ell$. In [17], the doubly spatial relation was defined over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} := \{\mathrm{Ker}(\boldsymbol{X}) \mid \boldsymbol{X} \in \mathbb{Z}_q^{\ell\times k},\ 0 \leq k \leq \ell\}$ and $\mathcal{Y} := \{\mathrm{Aff}(\boldsymbol{A}, \boldsymbol{a}) \mid (\boldsymbol{A}, \boldsymbol{a}) \in \mathbb{Z}_q^{\ell\times k} \times \mathbb{Z}_q^\ell,\ 0 \leq k \leq \ell\}$. The predicate encryption using the (doubly) spatial relation is called (doubly) spatial encryption ((D)SE). The authors in [17] showed that predicate encryption for the doubly spatial relation defined later generalizes the predicate encryption for the formerly defined doubly spatial relation.
- *Access structure based relation.* Let $\mathcal{U}$ be a universe of attributes. Define $\mathcal{X} = 2^{\mathcal{U}}$ and $\mathcal{Y}$ to be the set of all access structures over $\mathcal{U}$. For $A \in \mathcal{X}$ and $\Gamma \in \mathcal{Y}$, we define a binary relation $A \sim \Gamma$ if and only if $A \in \Gamma$. The encryption scheme realizing this relation is called attribute-based encryption (ABE) for access structures.
- *Policy over doubly spatial relation.* We have defined the access structure based relation above through the equality relation over a universe of attributes. Here we define a new access structure based relation of [1], called policy over doubly spatial relation, using the doubly spatial relation over a universe of affine

subspaces. This predicate generalizes the former access structure based relation. Let ℓ be a system index for this new access structure based relation. We define $\mathcal{U} := \{\mathrm{Aff}(\boldsymbol{A}, \boldsymbol{a}) \mid (\boldsymbol{A}, \boldsymbol{a}) \in \mathbb{Z}_q^{\ell \times k} \times \mathbb{Z}_q^{\ell},\ 0 \le k \le \ell\}$. Let $\mathcal{X} := 2^{\mathcal{U}}$ and $\mathcal{Y}$ be the set of all policies of the form $(\boldsymbol{M}, \rho)$, where $\boldsymbol{M} \in \mathbb{Z}_q^{d \times r}$ and $\rho\colon [d] \to \mathcal{U}$ is a row labeling function. For $S := \{Y_1, \ldots, Y_t\} \in \mathcal{X}$ and $\mathbb{A} := (\boldsymbol{M}, \rho) \in \mathcal{Y}$, we define $S \sim \mathbb{A}$ if and only if there exist coefficients $\{\mu_i\}_{i \in \mathcal{I}}$ with $\mathcal{I} = \{i \in [d] \mid \text{there exists } Y_j \in S \text{ with } \rho(i) \sim^{\mathrm{ds}} Y_j\}$ such that $\sum_{i \in \mathcal{I}} \mu_i \boldsymbol{M}_i = (1, \boldsymbol{0})$. The encryption scheme realizing this relation is called policy over doubly spatial encryption [1, 5].

- *Acceptance relation in regular language.* A deterministic finite automaton M is defined to be a quintuple $(Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is a finite set of symbols, called alphabet, $q_0 \in Q$ is called the start state, $F \subseteq Q$ is called the set of final states and $\delta\colon Q \times \Sigma \to Q$ is called transition function. The language, also called *regular language*, recognized by a deterministic finite automaton (DFA) M, is defined as
$$\mathcal{L}(M) = \{\sigma_1 \sigma_2 \cdots \sigma_n \in \Sigma^* \mid \delta(\cdots \delta(\delta(q_0, \sigma_1), \sigma_2) \cdots \sigma_n) \in F\}.$$
Let Tr denote the set of all transitions $(q_x, q_y, \sigma) \in Q \times Q \times \Sigma$ with the understanding that $\delta(q_x, \sigma) = q_y$. If we identify the δ by Tr, then a DFA M can always be represented by $(Q, \Sigma, \mathrm{Tr}, q_0, F)$. Let Σ be an alphabet, and let $\mathcal{X} := \Sigma^*$ and $\mathcal{Y}$ be the set of all DFAs with the same alphabet Σ. For $w \in \mathcal{X}$ and $M \in \mathcal{Y}$, we define a binary relation $w \sim M$ if $w \in \mathcal{L}(M)$. We also call this relation a DFA-based relation. The corresponding encryption scheme is known as functional encryption (FE) [38] for regular languages.

A relation defined over $\mathcal{X} \times \mathcal{Y}$ is called symmetric if $\mathcal{X} = \mathcal{Y}$ and $x \sim y \Leftrightarrow y \sim x$ for all $x, y \in \mathcal{X}$; otherwise, it is called asymmetric. For an asymmetric relation, we can define its dual relation as follows.

Definition 2.6 (Dual predicate). For a predicate tuple $(\sim, \mathcal{X}, \mathcal{Y})$, its dual predicate tuple $(\tilde{\sim}, \bar{\mathcal{X}}, \bar{\mathcal{Y}})$ is defined by $\bar{\mathcal{X}} := \mathcal{Y}$, $\bar{\mathcal{Y}} := \mathcal{X}$, and for $(x, y) \in \bar{\mathcal{X}} \times \bar{\mathcal{Y}}$, $x \,\tilde{\sim}\, y$ holds if and only if $y \sim x$ holds. The predicate $\tilde{\sim}$ is called dual predicate of $\sim$.

Remark 2.2. In this paper, we consider a predicate signature for all the relations described above and their dual (for asymmetric relations). If the underlying predicate or relation of the PS is not clearly stated, we assume that the PS stands for one of the aforementioned relations.

Here we are interested to design a predicate signature over composite order bilinear groups (CBG) and let N be the order of the groups. This N describes some domain; for example, the domain of IBE is $\mathbb{Z}_N$ with equality predicate. We therefore reserve the first entry of $\boldsymbol{j}$ to be N as described in [1]. For notational simplicity, we omit $\boldsymbol{j}$ and write $(\sim_N, \mathcal{X}_N, \mathcal{Y}_N)$ or simply $(\sim, \mathcal{X}, \mathcal{Y})$ depending upon requirements.

Definition 2.7 (Domain-transferable [1]). We say that $\sim$ is domain-transferable if, for p dividing N, the projection maps $f_1\colon \mathcal{X}_N \to \mathcal{X}_p$ and $f_2\colon \mathcal{Y}_N \to \mathcal{Y}_p$ such that, for all $(x, y) \in \mathcal{X}_N \times \mathcal{Y}_N$, we have

- *Completeness.* If $x \sim_N y$, then $f_1(x) \sim_p f_2(y)$.
- *Soundness.* (1) If $x \not\sim_N y$, then $f_1(x) \not\sim_p f_2(y)$, or (2) there exists an algorithm which takes (x, y) as input, where (1) does not hold, outputs a non-trivial factor F such that $p \mid F \mid N$.

Remark 2.3. Attrapadung [1] showed that the equality predicate (for IBE) is domain-transferable. Since all other predicates are defined through the equality predicate, all the predicates of [1, 39] are domain-transferable.

2.6 Predicate signature

A predicate signature (PS) scheme for a predicate family $\sim$ consists of four PPT algorithms – Setup, KeyGen, Sign and Ver.

- Setup takes a security parameter κ and a system index $\boldsymbol{j}$ as input and outputs public parameters $\mathcal{PP}$ and master secret key $\mathcal{MSK}$.
- KeyGen takes $\mathcal{PP}$, $\mathcal{MSK}$ and a key index $x \in \mathcal{X}$ as input and outputs a secret key $\mathcal{SK}_x$ corresponding to x.
- Sign takes $\mathcal{PP}$, a message $m \in \mathcal{M}$, a secret key $\mathcal{SK}_x$ and an associated data index $y \in \mathcal{Y}$ with $x \sim y$ and returns a signature δ.

- Ver receives $\mathcal{PP}$, a message $m \in \mathcal{M}$, a signature δ and a claimed associated data index y as input. It returns a Boolean value 1 for acceptance or 0 for rejection.

Correctness. For all $(\mathcal{PP}, \mathcal{MSK}) \leftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$, all $m \in \mathcal{M}$, all $x \in \mathcal{X}$, $\mathcal{SK}_x \leftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$ and all $y \in \mathcal{Y}$ with $x \sim y$, it is required that $\mathsf{Ver}(\mathcal{PP}, m, \mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_x, y), y) = 1$.

Remark 2.4. As in ABS of [29], we assume that the signer sends both signature and data index y to the receiver.

Public data index PS. The predicate signature defined above allows the data index to be publicly available to the receiver. This form of predicate signature is called *public data index* PS or PS *with public data index*. From now onwards, by predicate signature, we mean the predicate signature with public data index.

Form of PS. A predicate signature for the access structure based relation is called signature-policy attribute-based signature (SP-ABS) for access structures, and its dual form is called key-policy attribute-based signature (KP-ABS) for access structures. A predicate signature for the acceptance relation in regular languages is called SP-PS for regular languages, and its dual form is called KP-PS for regular languages. A predicate signature for the policy over doubly spatial relation is called signature policy over doubly spatial signature (SP-DSS), and its dual form is called key policy over doubly spatial signature (KP-DSS).

2.7 Security of predicate signature

Definition 2.8 (Signer privacy). A PS scheme is called perfectly private if, for all $(\mathcal{PP}, \mathcal{MSK}) \leftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$, all $x_1, x_2 \in \mathcal{X}$, $\mathcal{SK}_{x_1} \leftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x_1)$, $\mathcal{SK}_{x_2} \leftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x_2)$, all $m \in \mathcal{M}$ and all $y \in \mathcal{Y}$ with $x_1 \sim y$ and $x_2 \sim y$, the distribution of $\mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_{x_1}, y)$ and the distribution of $\mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_{x_2}, y)$ are identical, where the random coins of the distribution are only the random coins involved in the Sign algorithm.

Note that the signer privacy defined above is also called perfect privacy. A predicate signature scheme with signer privacy is called perfectly private.

Definition 2.9 (Adaptive unforgeability). A PS scheme is said to be existential unforgeable in adaptive model (or Ad-EUF-CMA) if, for all PPT algorithms $\mathscr{A}$, the advantage

$$\mathrm{Adv}^{\text{Ad-EUF-CMA}}_{\mathscr{A},\mathsf{PS}}(\kappa) := \Pr[\mathsf{Ver}(\mathcal{PP}, m^*, \delta^*, y^*) = 1 \wedge \mathrm{NRn}]$$

in $\mathrm{Exp}^{\text{Ad-EUF-CMA}}_{\mathscr{A},\mathsf{PS}}(\kappa)$ defined in Figure 1 is a negligible function in κ, where $\mathscr{A}$ is provided access to the KeyGen oracle $\mathcal{O}_{\mathrm{K}}$ and the Sign oracle $\mathcal{O}_{\mathrm{Sg}}$ (described below), and NRn is a natural restriction that (m^*, x, y^*) with $x \sim y^*$ was never queried to $\mathcal{O}_{\mathrm{Sg}}$ oracle, and for each key index x queried to $\mathcal{O}_{\mathrm{K}}$, it holds that $x \nsim y^*$.

- KeyGen *oracle* $\mathcal{O}_{\mathrm{K}}$. Given a key index x, the oracle returns $\mathcal{SK}_x \leftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$.
- Sign *oracle* $\mathcal{O}_{\mathrm{Sg}}$. Given (m, x, y), it runs $\mathcal{SK}_x \leftarrow \mathsf{KeyGen}(\mathcal{MSK}, x)$ if $\mathcal{SK}_x$ has not been generated previously[2] and then returns $\mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_x, y)$.

We may refer the above security model as the Ad-EUF-CMA security model in this paper.

$\mathrm{Exp}^{\text{Ad-EUF-CMA}}_{\mathscr{A},\mathsf{PS}}(\kappa)$

- $(\mathcal{PP}, \mathcal{MSK}) \leftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$
- $(\delta^*, m^*, y^*) \leftarrow \mathscr{A}^{\{\mathcal{O}_{\mathrm{K}}, \mathcal{O}_{\mathrm{Sg}}\}}(\mathcal{PP})$

Figure 1: Experiments for unforgeability.

2 The challenger maintains a log for storing the pairs of the forms $(x, \mathcal{SK}_x)$. Before generating a key for an index x, it searches x in the log. If x is not found, then it runs $\mathcal{SK}_x \leftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$ and inserts $(x, \mathcal{SK}_x)$ in the log; otherwise, it answers the query using $\mathcal{SK}_x$ available in the log.

2.8 Pair encoding scheme

A pair encoding scheme P (see [1]) for a predicate family $\sim$ consists of four deterministic algorithms, Param, Enc1, Enc2 and Pair.

- $\mathsf{Param}(\boldsymbol{j}) \to n \in \mathbb{N}$, where n describes the number of common variables involved in Enc1 and Enc2. Let $\boldsymbol{h} := (h_1, \ldots, h_n) \in \mathbb{Z}_N^n$ denote the common variables in Enc1 and Enc2.
- $\mathsf{Enc1}(x \in \mathcal{X}, N) \to (\boldsymbol{k}_x := (k_1, \ldots, k_{m_1}), m_2)$, where k_ι for $\iota \in [m_1]$ are polynomial over $\mathbb{Z}_N$ and $m_2 \in \mathbb{N}$ specifies the number of its own variables. We require that each polynomial k_ι is a linear combination of monomials $\alpha, r_j, h_i r_j$, where $\alpha, r_1, \ldots, r_{m_2}, h_1, \ldots, h_n$ are variables. In other words, it outputs a set of coefficients $\{b_\iota, b_{\iota,j}, b_{\iota,j,i}\}_{\iota\in[m_1],j\in[m_2],i\in[n]}$ which define the sequence of polynomials
$$\left(k_\iota(\alpha, \boldsymbol{r}, \boldsymbol{h}) := b_\iota \alpha + \left(\sum_{j\in[m_2]} b_{\iota,j} r_j\right) + \left(\sum_{\substack{j\in[m_2]\\ i\in[n]}} b_{\iota,j,i} h_i r_j\right)\right)_{\iota\in[m_1]}, \quad \text{where } \boldsymbol{r} := (r_1, \ldots, r_{m_2}).$$
- $\mathsf{Enc2}(y \in \mathcal{Y}, N) \to (\boldsymbol{c}_y := (c_1, \ldots, c_{\omega_1}), \omega_2)$, where c_ι for $\iota \in [\omega_1]$ are polynomial over $\mathbb{Z}_N$ and $\omega_2 \in \mathbb{N}$ specifies the number of its own variables. We require that each polynomial c_ι is a linear combination of monomials $s_j, h_i s_j$, where $s_0, \ldots, s_{\omega_2}, h_1, \ldots, h_n$ are variables. In other words, it outputs a set of coefficients $\{a_{\iota,j}, a_{\iota,j,i}\}_{\iota\in[\omega_1],j\in[0,\omega_2],i\in[n]}$ which define the sequence of polynomials
$$\left(c_\iota(\boldsymbol{s}, \boldsymbol{h}) := \sum_{j\in[0,\omega_2]} a_{\iota,j} s_j + \sum_{\substack{j\in[0,\omega_2]\\ i\in[n]}} a_{\iota,j,i} h_i s_j\right)_{\iota\in[\omega_1]}, \quad \text{where } \boldsymbol{s} := (s_0, \ldots, s_{\omega_2}).$$
- $\mathsf{Pair}(x, y, N) \to \boldsymbol{E} \in \mathbb{Z}_N^{m_1\times\omega_1}$.

Correctness. For all $N \in \mathbb{N}$, $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x, N)$, $(\boldsymbol{c}_y, \omega_2) \leftarrow \mathsf{Enc2}(y, N)$ and $\boldsymbol{E} \leftarrow \mathsf{Pair}(x, y, N)$, we have $\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}\boldsymbol{c}_y^\top(\boldsymbol{s}, \boldsymbol{h}) = \alpha s_0$ if $x \sim y$.

Properties of pair encoding scheme. We define two properties of a pair encoding scheme as follows:

- param-vanishing: $\boldsymbol{k}(\alpha, \boldsymbol{0}, \boldsymbol{h}) = \boldsymbol{k}(\alpha, \boldsymbol{0}, \boldsymbol{0})$,
- linearity: $\boldsymbol{k}(\alpha_1, \boldsymbol{r}_1, \boldsymbol{h}) + \boldsymbol{k}(\alpha_2, \boldsymbol{r}_2, \boldsymbol{h}) = \boldsymbol{k}(\alpha_1 + \alpha_2, \boldsymbol{r}_1 + \boldsymbol{r}_2, \boldsymbol{h})$ and $\boldsymbol{c}(\boldsymbol{s}_1, \boldsymbol{h}) + \boldsymbol{c}(\boldsymbol{s}_2, \boldsymbol{h}) = \boldsymbol{c}(\boldsymbol{s}_1 + \boldsymbol{s}_2, \boldsymbol{h})$.

2.9 Security of pair encoding scheme

We consider two forms of security, viz., perfect security and computational security as defined in [1].

Perfect security. A pair encoding scheme is said to be *perfectly master-key hiding* (PMH) if, for $N \in \mathbb{N}$, $x \not\sim_N y$, $n \leftarrow \mathsf{Param}(\boldsymbol{j})$, $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x, N)$ and $(\boldsymbol{c}_y, \omega_2) \leftarrow \mathsf{Enc2}(y, N)$, the following two distributions are identical:
$$\{\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}), \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\} \quad \text{and} \quad \{\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}), \boldsymbol{k}_x(0, \boldsymbol{r}, \boldsymbol{h})\},$$
where the random coins of the distributions are $\alpha \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n$, $\boldsymbol{s} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$ and $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$.

Computational security. Here we consider two types of computational security, viz., *selectively master-key hiding* (SMH) and *co-selectively master-key hiding* (CMH). A pair encoding scheme is said to have G security for $G \in \{\mathrm{SMH}, \mathrm{CMH}\}$ if, for $b \xleftarrow{\mathrm{U}} \{0, 1\}$, all PPT adversaries $\mathscr{A} := (\mathscr{A}_1, \mathscr{A}_2)$, the advantage
$$\mathrm{Adv}^G_{\mathscr{A},\mathsf{P}}(\kappa) := |\Pr[\mathrm{Exp}^G_{\mathscr{A},0}(\kappa) = 1] - \Pr[\mathrm{Exp}^G_{\mathscr{A},1}(\kappa) = 1]|$$
in the experiment $\mathrm{Exp}^G_{\mathscr{A},b}(\kappa)$ defined below is a negligible function in the security parameter κ,
$$\mathrm{Exp}^G_{\mathscr{A},b}(\kappa) := \left(\begin{array}{c} (N := p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathrm{cbg}}(1^\kappa),\\ (g, g_2, g_3) \xleftarrow{\mathrm{u}} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3},\\ \alpha \xleftarrow{\mathrm{U}} \mathbb{Z}_N,\ n \leftarrow \mathsf{Param}(\boldsymbol{j}),\ \boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n,\\ st \leftarrow \mathscr{A}_1^{\mathcal{O}^1_{G,b,\alpha,\boldsymbol{h}}(\cdot)}(g, g_2, g_3),\\ b' \leftarrow \mathscr{A}_2^{\mathcal{O}^2_{G,b,\alpha,\boldsymbol{h}}(\cdot)}(st) \end{array}\right),$$
where $\mathscr{A}$ is provided access to two oracles $\mathcal{O}^1_{G,b,\alpha,\boldsymbol{h}}(\cdot)$ and $\mathcal{O}^2_{G,b,\alpha,\boldsymbol{h}}(\cdot)$ defined below.

- For selective security, $\mathcal{O}^1$ is allowed only once, while $\mathcal{O}^2$ is allowed to query polynomially many times.
  - $\mathcal{O}^1_{\mathrm{SMH},b,\alpha,\boldsymbol{h}}(y^*)$ runs $(\boldsymbol{c}_{y^*}, \omega_2) \leftarrow \mathsf{Enc2}(y^*, p_2)$, picks $\boldsymbol{s} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$ and returns $\boldsymbol{C}_{y^*} := g_2^{\boldsymbol{c}_{y^*}(\boldsymbol{s},\boldsymbol{h})}$.
  - $\mathcal{O}^2_{\mathrm{SMH},b,\alpha,\boldsymbol{h}}(x)$ returns $\perp$ if $x \nprec_{p_2} y^*$, runs $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x, p_2)$, picks $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and then returns

$$\boldsymbol{K}_x := \begin{cases} g_2^{\boldsymbol{k}_x(0,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 0, \\ g_2^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 1. \end{cases}$$

- For co-selective security, both the oracles $\mathcal{O}^1$ and $\mathcal{O}^2$ are allowed to query only once.
  - $\mathcal{O}^1_{\mathrm{CMH},b,\alpha,\boldsymbol{h}}(x^*)$ runs $(\boldsymbol{k}_{x^*}, m_2) \leftarrow \mathsf{Enc1}(x^*, p_2)$, picks $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and then returns

$$\boldsymbol{K}_{x^*} := \begin{cases} g_2^{\boldsymbol{k}_{x^*}(0,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 0, \\ g_2^{\boldsymbol{k}_{x^*}(\alpha,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 1. \end{cases}$$

  - $\mathcal{O}^2_{\mathrm{CMH},b,\alpha,\boldsymbol{h}}(y)$ returns $\perp$ if $x^* \nprec_{p_2} y$, runs $(\boldsymbol{c}_y, \omega_2) \leftarrow \mathsf{Enc2}(y, p_2)$, picks $\boldsymbol{s} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$ and then returns

$$\boldsymbol{C}_y := g_2^{\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h})}.$$

Remark 2.5. In the above definition of computational security, if the oracles $\mathcal{O}^1$ and $\mathcal{O}^2$ are allowed to access respectively t_1 and t_2 times, then SMH (resp. CMH)-security, will be referred to as (t_1, t_2)-SMH (resp. (t_1, t_2)-CMH) security. What is considered in [1] are $(1, \mathrm{poly})$-SMH and $(1, 1)$-CMH security, respectively, for selectively and co-selectively master-key hiding. It is clear from the definitions of PMH and CMH-security that the PMH-security of a pair encoding scheme implies the CMH-security.

3 Framework for predicate signature

For better explanation of the uniform sampling process used in the Sign algorithm, we define a *h-free* variable for the random variables appearing in Enc2 as follows.

Definition 3.1. A variable (or coin) s_j for some $j \in [0, \omega_2]$ appearing in Enc2 of a pair encoding scheme is called "h-free" variable (or coin) if there exists a unique $\iota \in [\omega_1]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j} s_j$; otherwise, it is called "non-h-free" variable (or coin).

3.1 Natural requirements on pair encodings

For the correctness of the proposed construction, we keep a restriction on the underlying pair encoding scheme. Condition (1) defined in Conditions 3.1 is such a restriction on pair encodings. Condition (1) is also used in the security proof to ensure perfectness of the simulation.

One of the important features considered in the proposed predicate signature is signer privacy. To ensure perfect privacy of the signer in the proposed construction, we have to uniformly sample from

$$\mathbf{V}^\perp := \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \langle \boldsymbol{v}, \boldsymbol{u} \rangle = 0 \text{ for all } \boldsymbol{u} \in \mathbf{V}\},$$

where $\mathbf{V} := \{\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}) \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{s} := (s_0, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}\}$. Now finding elements of $\mathbf{V}^\perp$ is nothing but solving the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$, where $\boldsymbol{A}$ is a matrix of dimension $\omega_1 \times (\omega_2 + 1)$. More precisely, the matrix $\boldsymbol{A}$ is completely given by

$$\boldsymbol{A} := \left(a_{\iota,j} + \sum_{i \in [n]} a_{\iota,j,i} h_i\right)_{\substack{1 \le \iota \le \omega_1 \\ 0 \le j \le \omega_2}}.$$

Note that h_i are not given explicitly, but available in the form of g^{h_i}, where g is a generator for the underlying group. To solve the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$, we will apply the Gaussian elimination method, which is simply a sequence of elementary row (and/or column) operations. Since h_i are not known, it is difficult to find the inverses of some elements of $\boldsymbol{A}$ which are required for the elementary operations of type 2. So, to smooth

- Param $\to$ 6. Let $\boldsymbol{h} := (h_0, h_1, \phi_1, \phi_2, \phi_3, \eta)$.
- $\mathsf{Enc1}(\Gamma := (\boldsymbol{M}, \rho)) \to \boldsymbol{k}(\alpha, \boldsymbol{r}, \boldsymbol{h}) = (k_1, k_2, k_3, \{k_{4,i}, k_{5,i}, k_{6,i}\}_{i\in[\ell]})$, where $k_1 := \alpha + r\phi_1 + u\eta$, $k_2 := u$, $k_3 := r$, $k_{4,i} := \boldsymbol{M}_i\boldsymbol{v}^\top + r_i\phi_3$, $k_{5,i} := r_i$, $k_{6,i} := r_i(h_0 + h_1\rho(i))$ and $v_1 := r\phi_2$, $\boldsymbol{r} := (r, u, r_1, \ldots, r_\ell, v_2, \ldots, v_k)$, $\boldsymbol{v} := (v_1, \ldots, v_k)$.
- $\mathsf{Enc2}(S \subseteq \mathbb{Z}_N) \to \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) = (c_1, c_2, c_3, c_4, \{c_{5,y}\}_{y\in S}, \{c_{6,y}\}_{y\in S})$, where $c_1 := s$, $c_2 := s\eta$, $c_3 := s\phi_1 + w\phi_2$, $c_4 := w$, $c_{5,y} := w\phi_3 + s_y(h_0 + h_1 y)$, $c_{6,y} := s_y$ and $\boldsymbol{s} := (s, w, \{s_y\}_{y\in S})$.
- Correctness: If $x \sim y$, i.e., $\Gamma(S) = \text{True}$, there exist reconstruction coefficients $\{\mu_i\}_{i\in\mathcal{I}}$, with $\mathcal{I} := \{i \in [\ell] \mid \rho(i) \in S\}$ such that $\sum_{i\in\mathcal{I}} \mu_i \boldsymbol{M}_i\boldsymbol{v}^\top = v_1 = r\phi_2$. So the following linear combination reveals αs as $k_1c_1 - k_2c_2 - k_3c_3 + \sum_{i\in\mathcal{I}} \mu_i(k_{4,i}c_4 - k_{5,i}c_{5,\rho(i)} + k_{6,i}c_{6,\rho(i)}) = \alpha s - rw\phi_2 + \sum_{i\in\mathcal{I}} \mu_i(\boldsymbol{M}_i\boldsymbol{v}^\top w) = \alpha s$.

Figure 2: Pair Encoding Scheme 4 used in unbounded KP-ABE with large universes.

the process of elementary operations, we impose a restriction on pair encodings. Condition (3) given in Conditions 3.1 is such a restriction of pair encodings.

The security of the proposed construction is proven using the dual system methodology of Waters [37]. In this methodology, by applying hybrid arguments over a sequence of games, we reach a final game. The last game change (from previous to final game) relies on the DSG3 assumption. In the final game change, to maintain the correct distribution of semi-functional signatures, we impose condition (2) defined in Conditions 3.1 on pair encodings.

A pair encoding which satisfies condition (1) is referred to as normal in [5]. The authors of [14] used conditions (1) and (2) for showing CCA-security of their predicate encryption based on pair encodings. Condition (3) is newly introduced here for the predicate signature. For simplicity of explanation, we keep all of them under Conditions 3.1 defined next.

Conditions 3.1 (Sufficient). We have the following conditions:

(1) $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = s_0$ for some $\iota \in [\omega_1]$. Without loss of generality, assume that $c_1(\boldsymbol{s}, \boldsymbol{h}) = s_0$.

(2) For $(x, y) \in \mathcal{X} \times \mathcal{Y}$ with $x \sim y$, $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x, N)$ and $\boldsymbol{E} \leftarrow \mathsf{Pair}(x, y, N)$, we require that

$$\boldsymbol{k}_x(\alpha, \boldsymbol{0}, \boldsymbol{0})\boldsymbol{E} := (*, 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}, \quad \text{where } * \text{ is any entry from } \mathbb{Z}_N.$$

(3) For $j \in [0, \omega_2]$,
   (a) either (s_j *is h-free*) there is a unique $\iota \in [\omega_1]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j}s_j$,
   (b) or (s_j *is non-h-free*): first, case (a) has not happened; then if $a_{\iota,j,i'} \neq 0$ (appearing at the (ι, j)-th position of the matrix $\boldsymbol{A}$) for some $\iota \in [\omega_1]$, $i' \in [n]$, we require that i' must be unique, and for all $\iota \in [\omega_1]$, $i \in [n]$ with $i \neq i'$, $a_{\iota,j,i} = 0$, $a_{\iota,j} = 0$ (appearing at the (ι, j)-th position of the matrix $\boldsymbol{A}$), and $h_{i'}$ is co-prime to N.

We note that the first and third conditions are put on Enc2 and the second condition is imposed on Enc1 and Pair. Most of the pair encoding schemes considered in [1, 5, 39] satisfy condition (3) (a), i.e., for $j \in [0, \omega_2]$, the coin s_j is h-free. For better understanding, we work out the following pair encoding schemes of [1].

The pair encoding scheme given in Figure 2 was used to realize unbounded KP-ABE with large universes. We show that this pair encoding satisfies Conditions 3.1. Condition (1) is obvious. To verify condition (3), we see that, for each random variable s_i, there is a component c_ι such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = s_i$. Therefore, this is an example, where all the coins are h-free. For verifying condition (2), we first notice that $\boldsymbol{k}_x(\alpha, \boldsymbol{0}, \boldsymbol{0}) = (\alpha, \boldsymbol{0}, \boldsymbol{0})$. Hence we have to show that $E_{1j} = 0$ for $j \in [2, \omega_1]$. From the correctness of the scheme, we find that the monomials containing k_1 that appear in the correctness are exactly k_1c_1, so the first row of the matrix $\boldsymbol{E}$ must be $(1, \boldsymbol{0})$. Hence we are done.

Attrapadung [1] extracted Pair Encoding Scheme 10 (given in Figure 3) from the fully secure CP-ABE [25]. Again, condition (1) is obvious. For the random variables $s, s'_1, \ldots, s'_\ell$, condition (3) (a) holds. But, for $v_2, \ldots, v_k$, condition (3) (b) holds. For all v_j, the unique $h_{i'}$ is ϕ (for a clear view, see the matrix $\boldsymbol{A}^\top$ in Example 3.7). So we require that, during setup, ϕ is chosen to be co-prime to N. Condition (2) works similarly to Pair Encoding Scheme 4.

- $\mathsf{Param}(|\mathcal{U}|) \to |\mathcal{U}| + 1$. Let $\boldsymbol{h} := (\phi, \{h_i\}_{i\in\mathcal{U}})$.
- $\mathsf{Enc1}(S \subseteq \mathcal{U}) \to \boldsymbol{k}(\alpha, \boldsymbol{r}, \boldsymbol{h}) = (k_1 := \alpha + \phi r, \{k_{2,x} := r h_x\}_{x\in S}, k_3 := r)$, where $\boldsymbol{r} := r$.
- $\mathsf{Enc2}(\Gamma := (\boldsymbol{M}, \rho)) \to \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) = (c_1, \{c_{2,i}, c_{3,i}\}_{i\in[\ell]})$, where $\boldsymbol{M} \in \mathbb{Z}_N^{\ell\times k}$, $c_1 := s$, $c_{2,i} := \phi \boldsymbol{M}_i \boldsymbol{v}^\top + s_i' h_{\rho(i)}$, $c_{3,i} := s_i'$ and $\boldsymbol{s} := (s, v_2, \ldots, v_k, s_1', \ldots, s_\ell')$, $\boldsymbol{v} := (s, v_2, \ldots, v_k)$.
- Correctness: If $\Gamma(S) = \text{True}$, we have $\sum_{i\in\mathcal{I}} \mu_i \boldsymbol{M}_i \boldsymbol{v}^\top = \alpha$. So the following linear combination reveals αs as $k_1 c_1 + \sum_{i\in\mathcal{I}} \mu_i (k_3 c_{2,i} - k_{2,\rho(i)} c_{3,i}) = \alpha s$.

Figure 3: Pair Encoding Scheme 10 used in CP-ABE with small universes.

3.2 Dual conversion of pair encodings

We illustrate the dual conversion technique [1, 5] for converting a pair encoding for $\sim$ to another pair encoding for its dual predicate $\tilde{\sim}$ (Definition 2.6). For a pair encoding scheme P, its dual pair encoding scheme is denoted by $\mathbb{D}(\mathsf{P})$.

Let P be a given pair encoding scheme for the predicate $\sim$. A pair encoding scheme $\mathbb{D}(\mathsf{P})$ for the predicate $\tilde{\sim}$ is constructed as follows: For $(n, \boldsymbol{h}) \leftarrow \mathsf{Param}$, we define $\overline{\mathsf{Param}} := (n + 1, \bar{\boldsymbol{h}})$, where $\bar{\boldsymbol{h}} := (\boldsymbol{h}, \phi)$ and ϕ is a new variable.

- $\overline{\mathsf{Enc1}}(x, N)$ runs $(\boldsymbol{c}_x'(\boldsymbol{s}', \boldsymbol{h}), \omega_2) \leftarrow \mathsf{Enc2}(x, N)$, where $\boldsymbol{s}' := (s_0', \ldots, s_{\omega_2}')$, then sets
$$\boldsymbol{r} := \boldsymbol{s}' \quad \text{and} \quad \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \bar{\boldsymbol{h}}) := (\boldsymbol{c}_x'(\boldsymbol{s}', \boldsymbol{h}), \alpha + \phi . s').$$
Finally, it outputs $(\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \bar{\boldsymbol{h}}), \omega_2)$, where α is a new variable.
- $\overline{\mathsf{Enc2}}(y, N)$ runs $(\boldsymbol{k}_y'(\alpha', \boldsymbol{r}', \boldsymbol{h}), m_2) \leftarrow \mathsf{Enc1}(y, N)$, then sets
$$\boldsymbol{s} := (s_0, \boldsymbol{r}') \quad \text{and} \quad \boldsymbol{c}_y(\boldsymbol{s}, \bar{\boldsymbol{h}}) := (\boldsymbol{k}_y'(\phi . s_0, \boldsymbol{s}, \bar{\boldsymbol{h}}), s_0)$$
and returns $(\boldsymbol{c}_y(\boldsymbol{s}, \bar{\boldsymbol{h}}), m_2)$, where s_0 is a new variable.

The correctness is verified as follows: If $x \tilde{\sim} y$, then $y \sim x$, so, from the correctness of P, we have
$$\boldsymbol{k}_y'(\alpha', \boldsymbol{r}', \boldsymbol{h}) \boldsymbol{E}' \boldsymbol{c}_x'^\top(\boldsymbol{s}', \boldsymbol{h}) = \alpha' s_0' = (\phi \cdot s_0) s_0'.$$
Then, using the additional components, we have $(\alpha + \phi \cdot s_0')(s_0) - (\phi \cdot s_0) s_0' = \alpha s_0$.

Proposition 3.1 ([1]). *If a pair encoding scheme P for $\sim$ is perfectly master-key hiding, then the pair encoding scheme $\mathbb{D}(\mathsf{P})$ for $\tilde{\sim}$ is also perfectly master-key hiding.*

Proposition 3.2 ([5]). *If a pair encoding scheme P for $\sim$ is normal and $(1, 1)$-co-selectively master-key hiding, then the pair encoding scheme $\mathbb{D}(\mathsf{P})$ for $\tilde{\sim}$ is $(1, 1)$-selectively master-key hiding.*

Proposition 3.3 ([5]). *If a pair encoding scheme P for $\sim$ is normal and $(1, 1)$-selectively master-key hiding, then the pair encoding scheme $\mathbb{D}(\mathsf{P})$ for $\tilde{\sim}$ is $(1, 1)$-co-selectively master-key hiding.*

Observation 3.2. We first note that the pair encoding scheme $\mathbb{D}(\mathsf{P})$ satisfies condition (1) of Conditions 3.1 due to the newly added variable s_0. Let us examine condition (2). Without loss of generality, we set $c_{y,1} = s_0$ and $k_{x,1} = \alpha + \phi \cdot s_0'$. The correctness of $\mathbb{D}(\mathsf{P})$ says that
$$\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h}) \boldsymbol{E} \boldsymbol{c}_y^\top(\boldsymbol{s}, \boldsymbol{h}) = k_{x,1} \cdot c_{y,1} - \boldsymbol{k}_y'(\alpha', \boldsymbol{r}', \boldsymbol{h}) \boldsymbol{E}' \boldsymbol{c}_x'^\top(\boldsymbol{s}', \boldsymbol{h}) = \alpha s_0.$$
If $\boldsymbol{E}'$ has dimension $(m_1' \times \omega_1')$, then the dimension of $\boldsymbol{E}$ is $(m_1 \times \omega_1)$, where $m_1 = \omega_1' + 1$ and $\omega_1 = m_1' + 1$. Hence the matrix $\boldsymbol{E}$ has the form
$$E_{ij} := \begin{cases} 1 & \text{if } i = 1,\ j = 1 \\ 0 & \text{if } i = 1,\ j \in [2, \omega_1] \\ 0 & \text{if } i \in [2, m_1],\ j = 1 \\ -E'_{(j-1)(i-1)} & \text{if } i \in [2, m_1],\ j \in [2, \omega_1]. \end{cases}$$

Therefore, it is straightforward to check that the dual pair encoding scheme $\mathbb{D}(\mathsf{P})$ satisfies condition (2) of Conditions 3.1. We note that condition (3) of Conditions 3.1 is imposed on $\mathsf{Enc2}$; similarly, it could be defined over $\mathsf{Enc1}$, and let us call it condition $(\bar{3})$. One can verify that if a pair encoding scheme P for predicate $\sim$ fulfills condition $(\bar{3})$, then its dual $\mathbb{D}(\mathsf{P})$ for $\simeq$ satisfies condition (3). So far, we have checked that duals of all the pair encoding schemes [1, 5, 39] satisfy Conditions 3.1. Therefore, all the pair encoding schemes of [1, 5, 39] and their duals satisfy Conditions 3.1 and have either computational security (CMH and SMH) or PMH security.

3.3 Predicate signature from pair encoding scheme

Terminology. For fixed $\theta_1, \theta_2, \hbar \in \mathbb{Z}_N$ and $\boldsymbol{h} \in \mathbb{Z}_N^n$, we define

$$\boldsymbol{h}_{\mathrm{M}} := (\theta_1, \theta_2, \boldsymbol{h}), \quad \boldsymbol{\theta} := (\theta_1, \theta_2, \hbar) \quad \text{and} \quad c_0(z, \boldsymbol{\theta}) := z(\theta_1\hbar + \theta_2),$$

where z is an independent variable. Note that θ_1, θ_2, $\hbar$ and $\boldsymbol{h}$ will be understood from the context. For $(\boldsymbol{c}_y, \omega_2) \leftarrow \mathsf{Enc2}(y, N)$, define

$$\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}}) = (c_0(s_0, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})),$$

where $\boldsymbol{s} := (s_0, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}$. We set $\boldsymbol{c}_y^{\mathrm{M}} := (c_0, \boldsymbol{c}_y)$; then $|\boldsymbol{c}_y^{\mathrm{M}}| = \omega_1 + 1$ if $|\boldsymbol{c}_y| = \omega_1$. We define[3]

$$\mathbf{V}_{\mathrm{M}} := \{\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}}) \in \mathbb{Z}_N^{\omega_1+1} \mid \boldsymbol{s} := (s_0, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}\}.$$

Now we define an orthogonal set to be $(\mathbf{V}_{\mathrm{M}})^{\perp} := \{\boldsymbol{v}_{\mathrm{sp}} \in \mathbb{Z}_N^{\omega_1+1} \mid \langle \boldsymbol{v}_{\mathrm{sp}}, \boldsymbol{u}\rangle = 0 \text{ for all } \boldsymbol{u} \in \mathbf{V}_{\mathrm{M}}\}$. The process of sampling from $(\mathbf{V}_{\mathrm{M}})^{\perp}$ is given in Section 3.4.

Let $\mathsf{P} := (\mathsf{Param}, \mathsf{Enc1}, \mathsf{Enc2}, \mathsf{Pair})$ be a primitive pair encoding scheme which satisfies Conditions 3.1.

- $\mathsf{Setup}(1^\kappa, \boldsymbol{j})$ executes $\mathcal{J} := (N := p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathrm{cbg}}(1^\kappa)$ and chooses $g \xleftarrow{\mathrm{U}} \mathbb{G}_{p_1}; Z_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}$, then runs $n \leftarrow \mathsf{Param}(\boldsymbol{j})$ and picks $\boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n$, again picks $\alpha, \theta_1, \theta_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and sets $\boldsymbol{h}_{\mathrm{M}} := (\theta_1, \theta_2, \boldsymbol{h}) \in \mathbb{Z}_N^{n+2}$. Let $H\colon \{0,1\}^* \to \mathbb{Z}_N$ be a hash function. The public parameters and master secret key are given by

$$\mathcal{PP} := (\mathcal{J}, g, g^{\boldsymbol{h}_{\mathrm{M}}}, g_T^\alpha := e(g,g)^\alpha, Z_3, H) \quad \text{and} \quad \mathcal{MSK} := (\alpha).$$

- $\mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$ runs $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x, N)$. Let $|\boldsymbol{k}_x| = m_1$. It picks $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$ and outputs the secret key

$$\mathcal{SK}_x := (x, \boldsymbol{K}_x := g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})} \cdot \boldsymbol{R}_3).$$

- $\mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_x, y)$ returns $\perp$ if $x \nsim y$. Let $\mathcal{SK}_x = (x, \boldsymbol{K}_x)$. It runs[4]

$$\boldsymbol{K}_x := g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})} \cdot \boldsymbol{R}_3 \leftarrow \mathsf{Re\text{-}Randomize}(\boldsymbol{K}_x) \quad \text{and} \quad \mathsf{Pair}(x, y) \to \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1},$$

then computes $\hbar := H(m, y)$. It picks $\tau \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{v}_{\mathrm{sp}} \xleftarrow{\mathrm{U}} (\mathbf{V}_{\mathrm{M}})^{\perp}$ and $\boldsymbol{R}_3' \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{\omega_1+1}$ and sets

$$\boldsymbol{v} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}) \in \mathbb{Z}_N^{\omega_1+1},$$

where $\boldsymbol{\psi} := (\tau(\theta_1\hbar + \theta_2), 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$. The signature is given by

$$\boldsymbol{\delta}_y := g^{\boldsymbol{v}+\boldsymbol{v}_{\mathrm{sp}}} \cdot (1_{\mathbb{G}}, \boldsymbol{R}_3^{\boldsymbol{E}}) \cdot \boldsymbol{R}_3' \in \mathbb{G}^{\omega_1+1},$$

where $1_{\mathbb{G}}$ is the zero element of the source group $\mathbb{G}$. We note that $\boldsymbol{\delta}_y$ can be easily computed from $\mathcal{SK}_x$, $g^{\boldsymbol{h}_{\mathrm{M}}}$, $\boldsymbol{E}$ and the random coins involved in the sign algorithm. In fact, $\boldsymbol{\delta}_y$ is computed as follows:

$$\boldsymbol{\delta}_y = \underbrace{(g^{-\tau}, 1_{\mathbb{G}}, \ldots, 1_{\mathbb{G}})}_{\mathbb{G}^{\omega_1+1}} \cdot \underbrace{(1_{\mathbb{G}}, (g^{\theta_1})^{\tau\hbar} \cdot (g^{\theta_2})^{\tau}, 1_{\mathbb{G}}, \ldots, 1_{\mathbb{G}})}_{\mathbb{G}^{\omega_1+1}} \cdot \underbrace{(1_{\mathbb{G}}, \boldsymbol{K}_x^{\boldsymbol{E}})}_{\mathbb{G}^{\omega_1+1}} \cdot \underbrace{g^{\boldsymbol{v}_{\mathrm{sp}}}}_{\mathbb{G}^{\omega_1+1}} \cdot \underbrace{\boldsymbol{R}_3'}_{\mathbb{G}^{\omega_1+1}}.$$

---

3 We note that the set $\mathbf{V}_{\mathrm{M}}$ depends on $\boldsymbol{c}_y^{\mathrm{M}}$. A natural notation for the set could be $\mathbf{V}_{\boldsymbol{c}_y^{\mathrm{M}}}$, but for simplicity, we use $\mathbf{V}_{\mathrm{M}}$.

4 The linear property of the pair encodings guarantees the re-randomization of the keys. In fact, let $\boldsymbol{K}_x = g^{\boldsymbol{k}_x(\alpha, \tilde{\boldsymbol{r}}, \boldsymbol{h})} \cdot \widetilde{\boldsymbol{R}_3}$, where $\tilde{\boldsymbol{r}} \in \mathbb{Z}_N^{m_2}$, $\widetilde{\boldsymbol{R}_3} \in \mathbb{G}_{p_3}^{m_1}$ and $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x, N)$. $\mathsf{Re\text{-}Randomize}$ picks $\boldsymbol{r}' \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3' \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$, and sets

$$\boldsymbol{K}_x := g^{\boldsymbol{k}_x(\alpha, \tilde{\boldsymbol{r}}, \boldsymbol{h})} \cdot \widetilde{\boldsymbol{R}_3} \cdot g^{\boldsymbol{k}_x(0, \boldsymbol{r}', \boldsymbol{h})} \cdot \boldsymbol{R}_3' = g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})} \cdot \boldsymbol{R}_3,$$

where $\boldsymbol{r} := \tilde{\boldsymbol{r}} + \boldsymbol{r}' \in \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 := \widetilde{\boldsymbol{R}_3} \cdot \boldsymbol{R}_3' \in \mathbb{G}_{p_3}^{m_1}$.

- $\mathsf{Ver}(\mathcal{PP}, m, \boldsymbol{\delta}_y, y)$ runs $(\boldsymbol{c}_y, \omega_2) \leftarrow \mathsf{Enc2}(y, N)$ and picks $\boldsymbol{s} := (s_0, s_1, \ldots, s_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$, computes

$$\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}}) := (c_0(s_0, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{Z}_N^{\omega_1+1},$$

where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, $\hbar := H(m, y)$ and $c_0(s_0, \boldsymbol{\theta}) := s_0(\theta_1\hbar + \theta_2)$, then computes a verification text $\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := g_T^{\alpha s_0}, \mathcal{V}_y := g^{\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})})$. It returns 1 if $e(\boldsymbol{\delta}_y, \mathcal{V}_y) = \mathcal{V}_{\mathrm{INT}}$, else 0.

Correctness. For $x \sim_N y$ ($\Rightarrow x \sim_{p_1} y$ by domain-transferability), we have

$$\begin{aligned}
e(\boldsymbol{\delta}_y, \mathcal{V}_y) &= g_T^{\langle \boldsymbol{v}+\boldsymbol{v}_{\mathrm{sp}}, \boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})\rangle} && \text{(by orthogonality of CBG)}\\
&= g_T^{\langle \boldsymbol{v}, \boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})\rangle} && \text{(since } \boldsymbol{v}_{\mathrm{sp}} \in (\mathbf{V}_{\mathrm{M}})^{\perp}\text{)}\\
&= g_T^{\langle (-\tau, 0, \ldots, 0) + (0, \boldsymbol{\psi}) + (0, \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}), \boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})\rangle} && \text{(by definition of } \boldsymbol{v}\text{)}\\
&= g_T^{-\tau c_0(s, \boldsymbol{\theta}) + \tau(\theta_1\hbar+\theta_2)c_{y,1}(\boldsymbol{s}, \boldsymbol{h}) + \langle \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}, \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})\rangle}\\
&= g_T^{-\tau s_0(\theta_1\hbar+\theta_2) + \tau s_0(\theta_1\hbar+\theta_2) + \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}\boldsymbol{c}_y^{\top}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})} && \text{(since } c_{y,1}(\boldsymbol{s}, \boldsymbol{h}) = s_0\text{)}\\
&= g_T^{\alpha s_0} && \text{(by correctness of P)}
\end{aligned}$$

Remark 3.3. In the Sign algorithm, two random coins τ and $\boldsymbol{v}_{\mathrm{sp}}$ are used; among them, $\boldsymbol{v}_{\mathrm{sp}}$ is assigned only for signer privacy, and τ is the only coin that provides randomness in unforgeability. If signer privacy is not required, we can ignore $\boldsymbol{v}_{\mathrm{sp}}$.

Fact 3.4. We note that the size of the signature for a message (m, y) is $\omega_1 + 1$, where $|\boldsymbol{c}_y| = \omega_1$, and the number of pairings in Ver is $\omega_1 + 1$. *Therefore, if $\boldsymbol{c}_y$ of the underlying pair encoding scheme is of constant size, then the corresponding signature will be of constant size and the number of pairings in verification will be of constant size*. One example of such pair encodings is [1, Pair Encoding Scheme 5].

3.4 How to uniformly sample from $(\mathbf{V}_{\mathbf{M}})^{\perp}$

Let $\mathbf{V} := \{\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}) \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{s} := (s_0, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}\}$ and $\mathbf{V}^{\perp} := \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \langle \boldsymbol{v}, \boldsymbol{u}\rangle = 0 \text{ for all } \boldsymbol{u} \in \mathbf{V}\}$. Note that there is no known method to sample uniformly from $\mathbf{V}^{\perp}$ for arbitrary pair encoding schemes. However, it is possible if we put a condition on Enc2 of P. Condition (3) of Conditions 3.1 is such a condition. Let $\boldsymbol{s} = (s_0, \ldots, s_{\omega_2})$ and $\boldsymbol{h} = (h_1, \ldots, h_n)$. Write $\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}) = \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) = (c_1(\boldsymbol{s}, \boldsymbol{h}), \ldots, c_{\omega_1}(\boldsymbol{s}, \boldsymbol{h}))$, where $c_\iota(\boldsymbol{s}, \boldsymbol{h})$ is given by

$$c_\iota(\boldsymbol{s}, \boldsymbol{h}) := \sum_{j \in [0, \omega_2]} a_{\iota,j} s_j + \sum_{\substack{j \in [0, \omega_2]\\ i \in [n]}} a_{\iota,j,i} h_i s_j.$$

Then $\boldsymbol{c}_y^{\top}(\boldsymbol{s}, \boldsymbol{h})$ can be written as $\boldsymbol{c}_y^{\top}(\boldsymbol{s}, \boldsymbol{h}) = \boldsymbol{A}\boldsymbol{s}^{\top}$, where the matrix $\boldsymbol{A} \in \mathbb{Z}_N^{\omega_1 \times (\omega_2+1)}$ is given by

$$\boldsymbol{A} := \left(a_{\iota,j} + \sum_{i \in [n]} a_{\iota,j,i} h_i\right)_{\substack{1 \le \iota \le \omega_1\\ 0 \le j \le \omega_2}}.$$

For simplicity of the description, we assign labels for the columns of $\boldsymbol{A}$ from 0 to ω_2. We call the matrix $\boldsymbol{A}$ associated matrix for $\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})$. The matrix $\boldsymbol{A}$ is described by $a_{\iota,j}$, $a_{\iota,j,i}$ and h_i, where $\iota \in [\omega_1]$, $j \in [0, \omega_2]$ and $i \in [n]$. Note that $a_{\iota,j}$ and $a_{\iota,j,i}$ are the coefficients of the polynomials c_ι with $(\boldsymbol{c}_y, \omega_2) \leftarrow \mathsf{Enc2}(y, N)$. Therefore, the matrix $\boldsymbol{A}$ is completely determined by $y \in \mathcal{Y}$ and $\boldsymbol{h}$. Since the part $\boldsymbol{h}$ is fixed, we say that $\boldsymbol{A}$ is associated with $y \in \mathcal{Y}$. Then, from the definition of $\mathbf{V}^{\perp}$, we have

$$\begin{aligned}
\mathbf{V}^{\perp} &= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \langle \boldsymbol{v}, \boldsymbol{u}\rangle = 0 \text{ for all } \boldsymbol{u} \in \mathbf{V}\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{v}\boldsymbol{c}_y^{\top}(\boldsymbol{s}, \boldsymbol{h}) = 0 \text{ for all } \boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{v}\boldsymbol{A}\boldsymbol{s}^{\top} = 0 \text{ for all } \boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{v}\boldsymbol{A} = \boldsymbol{0}\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{A}^{\top}\boldsymbol{v}^{\top} = \boldsymbol{0}\}.
\end{aligned}$$

Now sampling from $\mathbf{V}^{\perp}$ boils down to solving the homogeneous system $\boldsymbol{A}^{\top}\boldsymbol{X} = \mathbf{0}$ with $\boldsymbol{X}^{\top} := (x_1, \ldots, x_{\omega_1})$. *Before proceeding further, we note that sampling of* $\mathbf{V}^{\perp}$ *gives rise to the sampling of* $(\mathbf{V}_{\mathrm{M}})^{\perp}$ *if* $c_1(\boldsymbol{s}, \boldsymbol{h}) = s_0$. This is assured using Theorem 2.3, where $\boldsymbol{A}_{\mathrm{M}}^{\top}$ is defined from $\boldsymbol{A}^{\top}$ and $t := \theta_1\hbar + \theta_2$.

Our goal is to compute $g^{\boldsymbol{v}}$, where $\boldsymbol{v} \xleftarrow{\mathrm{U}} \mathbf{V}^{\perp}$. Note that $g^{\boldsymbol{h}}$ is given but not $\boldsymbol{h}$. If each component v_j of $\boldsymbol{v}$ is a linear combination of h_i, then we will be able to compute $g^{\boldsymbol{v}}$. In fact, for each $\iota \in [\omega_1]$, if $v_\iota = \sum_{i=1}^{n} \chi_{\iota,i} h_i$, where $\chi_{\iota,i} \in \mathbb{Z}_N$ for $i \in [n]$, then g^{v_ι} can be computed as $(g^{h_1})^{\chi_{\iota,1}} \cdots (g^{h_n})^{\chi_{\iota,n}}$.

Since h_i are not known, we are not able to compute h_i^{-1} required for the elementary operations of type 2 (for details of the elementary operations, refer to Section 2.4). It may even happen that h_i are not invertible in $\mathbb{Z}_N$. So the only information of $\boldsymbol{A}$ available in the process of elementary operations are a_ι, $a'_{\iota,i}$, $a_{\iota,j}$ and $a_{\iota,j,i}$. Therefore, throughout the elementary operations, we treat h_i as symbols, where the symbols h_i^{-1} are not known. But if we find some row of $\boldsymbol{A}^{\top}$ is a multiple of h_i, then we can multiply the row by h_i^{-1} (provided it exists in $\mathbb{Z}_N$) to make the row h_i-free. Under these multiplications, the solution of the system remains unchanged.

Suppose $\boldsymbol{M}$ is obtained by applying say n elementary column operations on $\boldsymbol{A}^{\top}$. Then we have

$$\boldsymbol{A}^{\top}\mathcal{E}_1^{\top}\mathcal{E}_2^{\top}\cdots\mathcal{E}_n^{\top} = \boldsymbol{M},$$

where $\mathcal{E}_i$ are elementary matrices. If the column operations are other than the type 1 operation, then there is a chance that h_i may appear in the elementary matrix $\mathcal{E}_j^{\top}$. Since, for each solution $\boldsymbol{v} := (v_1, \ldots, v_{\omega_1})^{\top}$ of $\boldsymbol{M}\boldsymbol{X} = \mathbf{0}$, $\mathcal{E}_1^{\top}\mathcal{E}_2^{\top}\cdots\mathcal{E}_n^{\top}\boldsymbol{v}$ is a solution of $\boldsymbol{A}^{\top}\boldsymbol{X} = \mathbf{0}$ and v_ι are a linear combination of h_i, terms like $h_{i_1}h_{i_2}\cdots h_{i_k}$ may appear in $\boldsymbol{v}$ to complicate things. For this reason, we avoid the elementary column operations in the sampling process.

Below, we define the leading h-free column of a matrix which comes in connection with h-free coins (Definition 3.1). The definition says that, for each h-free coin s_j, there is a unique leading h-free column of the matrix $\boldsymbol{A}^{\top}$.

Definition 3.2. A ι-th column of $\boldsymbol{A}^{\top}$ is said to be a "leading h-free" column if there exists a $j \in [0, \omega_2]$ such that all the entries of the ι-th column of $\boldsymbol{A}^{\top}$ are 0 except $A_{j,\iota}^{\top} = a_{\iota,j}$.

For Examples 3.6 and 3.7, the leading h-free columns of $\boldsymbol{A}^{\top}$ are $\{1, 4, 6, 8, 10\}$ and $\{1, 3, 5, 7, 9\}$, respectively.

More notations. We define

$$S_{\mathrm{hf}} := \{\iota \in [\omega_1] \mid \text{there exists } j \in [0, \omega_2] \text{ such that } c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j}s_j\},$$
$$T_{\mathrm{hf}} := \{j \in [0, \omega_2] \mid \text{there exists } \iota \in [\omega_1] \text{ such that } c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j}s_j\}.$$

We remark that S_{hf} and T_{hf} are respectively the collection of indices for h-free columns and h-free coins. Let $S_{\text{non-hf}} := [\omega_1] \setminus S_{\mathrm{hf}}$ and $T_{\text{non-hf}} := [0, \omega_2] \setminus T_{\mathrm{hf}}$. The main task is to find which variables are free and which are not among $x_1, \ldots, x_{\omega_1}$ with $\boldsymbol{X} := (x_1, \ldots, x_{\omega_1})^{\top}$ for the homogeneous system $\boldsymbol{A}^{\top}\boldsymbol{X} = \mathbf{0}$. Let S_{fv} and $S_{\text{non-fv}}$ respectively represent the indices for free variables and non-free variables.

Remark 3.5. Since the factorization problem is assumed to be intractable, all $a_{\iota,j}$ appearing in condition (3) (a) are invertible in $\mathbb{Z}_N$ (as discussed in Section 2.4). For most of the existing pair coding schemes, $a_{\iota,j}$ are found to be 1. When all the variables are h-free, then $T_{\text{non-hf}} = \emptyset$.

Algorithm for sampling. As discussed above, the sampling from $\mathbf{V}^{\perp}$ boils down to solving $\boldsymbol{A}^{\top}\boldsymbol{X} = \mathbf{0}$ with $\boldsymbol{X}^{\top} = (x_1, \ldots, x_{\omega_1})$. The matrix $\boldsymbol{A}$ is completely determined by $a_{\iota,j}$, $a_{\iota,j,i}$ and h_i, where $\iota \in [\omega_1]$, $j \in [0, \omega_2]$ and $i \in [n]$. Since h_i are not known, the input matrix $\boldsymbol{A}$ to the algorithm is supplied by $a_{\iota,j}$, $a_{\iota,j,i}$ and g^{h_i}, where $\iota \in [\omega_1]$, $j \in [0, \omega_2]$ and $i \in [n]$. We call this form of input for the matrix $\boldsymbol{A}$ implicit form of $\boldsymbol{A}$. The algorithm returns $(g^{x_1}, \ldots, g^{x_{\omega_1}})$, where $(x_1, \ldots, x_{\omega_1})$ is a uniform solution of $\boldsymbol{A}^{\top}\boldsymbol{X} = \mathbf{0}$, which we call implicit form of solution for the system. We describe Algorithm 1 for sampling in detail, which takes as input the matrix $\boldsymbol{A}$ in implicit form associated with some $y \in \mathcal{Y}$ and outputs a uniform solution in implicit form of the system $\boldsymbol{A}^{\top}\boldsymbol{X} = \mathbf{0}$. Algorithm 1 separately handles two cases, all s_j are h-free and not all s_j are h-free. The additional comments for the statements of Algorithm 1 are described in detail below.

- *All s_j are h-free.* Lines 2–11 represent the case that all s_j involved in $\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})$ are h-free. For this case, we do not require any elementary operation. In this case, $\mathrm{Null}(\boldsymbol{A}^{\top}) = \omega_1 - (\omega_2 + 1)$. For better understanding this case, we refer to Example 3.6.

Input: $a_{\iota,j}$, $a_{\iota,j,i}$ and g^{h_i}, where $\iota \in [\omega_1]$, $j \in [0, \omega_2]$ and $i \in [n]$ ($\boldsymbol{A}$ in implicit form).
Output: $g^{(x_1,\ldots,x_{\omega_1})}$, where $(x_1, \ldots, x_{\omega_1})^\top$ is a uniform solution of the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$.

1 Compute the sets S_{hf}, $S_{\text{non-hf}}$ and $T_{\text{non-hf}}$;
2 **if** *all* s_j *are h-free* **then**
3 $S_{\text{non-fv}} := S_{\text{hf}}$ and $S_{\text{fv}} := [\omega_1] \setminus S_{\text{non-fv}}$;
4 **for** $i \in S_{\text{fv}}$ **do**
5 $x_i := \chi_i \xleftarrow{\text{U}} \mathbb{Z}_N$; // free variables are assigned uniformly
6 **end**
7 **for** $\iota \in S_{\text{non-fv}}$ **do**
8 $x_\iota := -a_{\iota,j}^{-1} \sum_{i \in S_{\text{fv}}} A_{j,i}^\top \chi_i$; // for each $\iota \in S_{\text{non-fv}}$, there is a unique $j \in [0, \omega_2]$
9 **end**
10 **return** $(g^{x_1}, \ldots, g^{x_{\omega_1}})$; // refer to Example 3.6
11 **end**
12 **else**
13 **for** $j \in T_{\text{non-hf}}$ **do**
14 $\boldsymbol{A}_j^\top \leftarrow h_{i'}^{-1} \boldsymbol{A}_j^\top$; // j-th row of $\boldsymbol{A}^\top$ is multiplied by $h_{i'}^{-1}$
15 **end**
16 **for** $j \in T_{\text{non-hf}}$ **do**
17 $k :=$ the first non-zero (leading) element of the j-th row ;
18 **if** $\gcd(k, N) > 1$ **then**
19 **return** $\gcd(k, N)$; // solves factorization problem for N
20 **end**
21 **else**
22 $\boldsymbol{A}_j^\top \leftarrow k^{-1} \boldsymbol{A}_j^\top$; // change the leading element to 1
23 all other elements of the column containing the leading 1 are changed to 0 ;
24 **end**
25 **end** // $\boldsymbol{M} :=$ matrix obtained by applying above operations on $\boldsymbol{A}^\top$
26 $S_{\text{new}} := \{\iota \in S_{\text{non-hf}} \mid \text{there exists } j \in T_{\text{non-hf}} \text{ such that } M_{i\iota} = \delta_{i,j}\}$;
27 $S_{\text{non-fv}} := S_{\text{hf}} \cup S_{\text{new}}$; // set of non-free variables for $\boldsymbol{MX} = \boldsymbol{0}$
28 $S_{\text{fv}} := [\omega_1] \setminus S_{\text{non-fv}}$; // set of free variables for $\boldsymbol{MX} = \boldsymbol{0}$
29 **for** $i \in S_{\text{fv}}$ **do**
30 $x_i := \chi_i \xleftarrow{\text{U}} \mathbb{Z}_N$; // free variables are assigned uniformly
31 **end**
32 **for** $\iota \in S_{\text{non-fv}}$ **do**
33 $x_\iota := -(M_{j,\iota})^{-1} \sum_{i \in S_{\text{fv}}} M_{j,i} \chi_i$; // for each $\iota \in S_{\text{non-fv}}$ there is a unique $j \in [0, \omega_2]$
34 **end**
35 **return** $(g^{x_1}, \ldots, g^{x_{\omega_1}})$; // refer to Example 3.7
36 **end**

Algorithm 1: An algorithm for uniform sampling from $\mathbf{V}^\perp$.

  - Lines 7–9: For each $\iota \in S_{\text{hf}}$, there is a unique $j \in [0, \omega_2]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j} s_j$ by condition (3) (a). Condition (3) (a) guarantees that no non-free variable contributes during the computation of others.
- *Not all s_j are h-free.* Lines 12–36 represent the case that not all s_j involved in $\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})$ are h-free. In this case, $\text{Null}(\boldsymbol{A}^\top) \le \omega_1 - (\omega_2 + 1)$. For better understanding, we refer to Example 3.7.
  - Line 14: For each $j \in T_{\text{non-hf}}$, there is a unique i' such that $a_{\iota,j,i'} \neq 0$, and for all $\iota \in [\omega_1]$, $i \in [n]$ with $i \neq i'$, $a_{\iota,j,i} = 0$, $a_{\iota,j} = 0$ by condition (3) (b). On line 14, the j-th row of $\boldsymbol{A}$ is multiplied by $h_{i'}^{-1}$ symbolically to make each element of the j-th row free from the h-term. Under these changes, the h-free variables remain h-free as the corresponding leading h-free columns are unaffected. Since $h_{i'}$ is invertible (by condition (3) (b)), the solutions of the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$ remain unaltered.

- Line 16–25 apply the elementary row operations of type 2 and 3 until each row $j \in T_{\text{non-hf}}$ becomes row-reduced.
- Lines 18–20 solve the factorization problem in polynomial time in κ and aborts. In this case, $\gcd(k, N)$ is a factor of N.
- Line 23 applies the elementary row operations of type 3 to reduce all other elements of the column containing the leading 1 to 0.
- Line 25: Under the elementary row operations of type 2 and 3 used in lines 22 and 23, the h-free variables remain h-free as the corresponding leading h-free columns are unaffected, but some non-h-free variables become h-free. These new h-free variables change the free variables to non-free variables.
- Line 26: $S_{\text{non-fv}}$ is the set of new non-free variables.
- Lines 32–34: Note that the set of non-free variables to the system $\boldsymbol{MX} = \boldsymbol{0}$ is $S_{\text{non-fv}} := S_{\text{hf}} \cup S_{\text{new}}$. As in the first case, for each $\iota \in S_{\text{hf}}$, there is a unique $j \in [0, \omega_2]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j} s_j$ by condition (3) (a). For each $\iota \in S_{\text{new}}$, there is a unique $j \in [0, \omega_2]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j} s_j$ by lines 16–25.

Example 3.6. For better understanding of Algorithm 1, we work out Pair Encoding Scheme 4 (given in Section 3.1). We customize a set of attributes to be $S := \{y_2, y_3, y_4\} \subset \mathbb{Z}_N$.

$$\mathsf{Enc2}(S) \to \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) = (c_1 := s,\ c_2 := s\eta,\ c_3 := s\phi_1 + w\phi_2,\ c_4 := w,\ \{c_{5,y}, c_{6,y}\}_{y \in S}),$$

where $c_{5,y} := w\phi_3 + s_y(h_0 + h_1 y)$, $c_{6,y} := s_y$ and $\boldsymbol{s} := (s_0 := s, s_1 := w, s_2, s_3, s_4)$ with $s_i := s_{y_i}$ for $i \geq 2$. The matrix[5] of the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$ is given by

$$\boldsymbol{A}^\top = \begin{array}{c} \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{array} \begin{array}{c} \begin{array}{ccccccccccc} c_1 & c_2 & c_3 & c_4 & c_{5,y_2} & c_{6,y_2} & c_{5,y_3} & c_{6,y_3} & c_{5,y_4} & c_{6,y_4} \end{array} \\ \begin{pmatrix} \boxed{1} & \eta & \phi_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \phi_2 & \boxed{1} & \phi_3 & 0 & \phi_3 & 0 & \phi_3 & 0 \\ 0 & 0 & 0 & 0 & h_0 + h_1 y_2 & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 + h_1 y_3 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_0 + h_1 y_4 & \boxed{1} \end{pmatrix} \end{array}.$$

This is a case where *all the coins are h-free*. Here $\omega_1 = 10$, $\omega_2 = 4$, $S_{\text{hf}} := \{1, 4, 6, 8, 10\}$, $T_{\text{hf}} := \{0, 1, 2, 3, 4\}$. Therefore, $S_{\text{non-fv}} := S_{\text{hf}} = \{1, 4, 6, 8, 10\}$ and $S_{\text{fv}} := [10] \setminus S_{\text{non-fv}} = \{2, 3, 5, 7, 9\}$. For each $i \in S_{\text{fv}}$, we have $x_i := \chi_i \xleftarrow{\text{U}} \mathbb{Z}_N$. The non-free variables are computed as $x_1 := -\eta\chi_2 - \phi_2\chi_3$, $x_4 := -\phi_2\chi_3 - \phi_3(\chi_5 + \chi_7 + \chi_9)$, $x_6 := -(h_0 + h_1 y_2)\chi_5$, $x_8 := -(h_0 + h_1 y_3)\chi_7$, $x_{10} := -(h_0 + h_1 y_4)\chi_9$. Therefore, $(x_1, \ldots, x_{10})^\top$ is a solution of the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$. If $\boldsymbol{v} = (x_1, \ldots, x_{10})$, then $g^{\boldsymbol{v}}$ is computed as

$$g^{x_1} := (g^\eta)^{-\chi_2}.(g^{\phi_2})^{-\chi_3},\quad g^{x_2} := g^{\chi_2},\quad g^{x_3} := g^{\chi_3},\quad g^{x_4} := (g^{\phi_2})^{-\chi_3}.(g^{\phi_3})^{-(\chi_5+\chi_7+\chi_9)},$$
$$g^{x_5} := g^{\chi_5},\quad g^{x_6} := (g^{h_0})^{-\chi_5}.(g^{h_1})^{-y_2\chi_5},\quad g^{x_7} := g^{\chi_7},\quad g^{x_8} := (g^{h_0})^{-\chi_7}.(g^{h_1})^{-y_3\chi_7},$$
$$g^{x_9} := g^{\chi_9} \quad \text{and} \quad g^{x_{10}} := (g^{h_0})^{-\chi_9}.(g^{h_1})^{-y_4\chi_9}.$$

Example 3.7. We also consider Pair Encoding Scheme 10 (described in Section 3.1) which explains other case of Algorithm 1. Let $\Gamma := (\boldsymbol{M}, \rho)$ be a span program, where $\rho : [4] \to \mathcal{U}$ is some row labeling function and $\boldsymbol{M}$ is given by

$$\boldsymbol{M} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 2 & 1 \\ 3 & 1 & 3 \end{pmatrix}.$$

If we run Enc2 of Pair Encoding Scheme 10 on Γ, we have the output $\boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) = (c_1, \{c_{2,i}, c_{3,i}\}_{i \in [4]})$, where $c_1 := s$, $c_{2,i} := \phi \boldsymbol{M}_i \boldsymbol{v}^\top + s'_i h_{\rho(i)}$, $c_{3,i} := s'_i$ and

$$\boldsymbol{s} := (s_0 := s,\ s_1 := v_2,\ s_2 := v_3,\ s_3 := s'_1,\ s_4 := s'_2,\ s_5 := s'_3,\ s_6 := s'_4), \quad \boldsymbol{v} := (s, v_2, v_3).$$

5 The box in the j-th row indicates that the coin s_j is h-free and the corresponding column containing the box is leading h-free column.

The matrix of the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$ is given by

$$\boldsymbol{A}^\top = \begin{array}{c} \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{array} \begin{array}{c} \begin{array}{ccccccccc} c_1 & c_{2,1} & c_{3,1} & c_{2,2} & c_{3,2} & c_{2,3} & c_{3,3} & c_{2,4} & c_{3,4} \end{array} \\ \begin{pmatrix} \boxed{1} & \phi & 0 & 2\phi & 0 & 3\phi & 0 & 3\phi & 0 \\ 0 & 2\phi & 0 & 3\phi & 0 & 2\phi & 0 & \phi & 0 \\ 0 & 3\phi & 0 & 4\phi & 0 & \phi & 0 & 3\phi & 0 \\ 0 & h_{\rho(1)} & \boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_{\rho(2)} & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & h_{\rho(3)} & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_{\rho(4)} & \boxed{1} \end{pmatrix} \end{array}.$$

This is a case, where *all the coins are not h-free*. For all the non-h-free coins (there are only two non-h-free coins, v_2 and v_3), there is a unique h-term which is ϕ. Here $\omega_1 = 9$, $\omega_2 = 6$, $S_{\text{hf}} := \{1, 3, 5, 7, 9\}$, $S_{\text{non-hf}} := [9] \setminus S_{\text{hf}} = \{2, 4, 6, 8\}$, $T_{\text{hf}} := \{0, 3, 4, 5, 6\}$ and $T_{\text{non-hf}} := [0, 6] \setminus T_{\text{hf}} = \{1, 2\}$. Note that the labeling of the rows starts with 0. For each $j \in T_{\text{non-hf}}$, the j-th row is multiplied by ϕ^{-1} to make the j-th row free from ϕ. We now apply the following elementary row operations of type 2 and type 3 to make each row $j \in T_{\text{non-hf}}$ of $\boldsymbol{A}^\top$ row-reduced: $R_2 \leftarrow 2^{-1}R_2$, $R_1 \leftarrow R_1 + (-\phi)R_2$, $R_3 \leftarrow R_3 + (-3)R_2$, $R_4 \leftarrow R_4 + (-h_{\rho(1)})R_2$, $R_3 \leftarrow (-2)R_3$, $R_1 \leftarrow R_1 + (-\phi/2)R_3$, $R_2 \leftarrow R_2 + (-3/2)R_3$, $R_4 \leftarrow R_4 + 3h_{\rho(1)}/2R_3$ and $R_5 \leftarrow R_5 + (-h_{\rho(2)})R_3$. Let $\boldsymbol{M}$ (given below) be the matrix obtained from $\boldsymbol{A}^\top$ after applying the above elementary row operations. The elements appearing in the double boxes of the row-reduced rows of $\boldsymbol{M}$ are the new leading elements of the corresponding rows.

$$\boldsymbol{M} = \begin{pmatrix} \boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 & 4\phi & 0 \\ 0 & \boxed{\boxed{1}} & 0 & 0 & 0 & -5 & 0 & 5 & 0 \\ 0 & 0 & 0 & \boxed{\boxed{1}} & 0 & 4 & 0 & -3 & 0 \\ 0 & 0 & \boxed{1} & 0 & 0 & 5h_{\rho(1)} & 0 & -5h_{\rho(1)} & 0 \\ 0 & 0 & 0 & 0 & \boxed{1} & -4h_{\rho(2)} & 0 & 3h_{\rho(2)} & 0 \\ 0 & 0 & 0 & 0 & 0 & h_{\rho(3)} & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_{\rho(4)} & \boxed{1} \end{pmatrix}.$$

Then $S_{\text{new}} := \{2, 4\}$, so $S_{\text{non-fv}} := S_{\text{hf}} \cup S_{\text{new}} = \{1, 2, 3, 4, 5, 7, 9\}$ and $S_{\text{fv}} := \{6, 8\}$. For each $i \in S_{\text{fv}}$, we have $x_i := \chi_i \xleftarrow{\text{U}} \mathbb{Z}_N$. The non-free variables are computed as $x_1 := -4\phi\chi_8$, $x_2 := 5(\chi_6 - \chi_8)$, $x_3 := -5h_{\rho(1)}(\chi_6 - \chi_8)$, $x_4 := -4\chi_6 + 3\chi_8$, $x_5 := h_{\rho(2)}(4\chi_6 - 3\chi_8)$, $x_7 := -h_{\rho(3)}\chi_6$ and $x_9 := -h_{\rho(4)}\chi_8$. Thus $(x_1, \ldots, x_{10})^\top$ is a solution of the system $\boldsymbol{MX} = \boldsymbol{0}$ and hence a solution of $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$. If $\boldsymbol{v} = (x_1, \ldots, x_9)$, then $g^{\boldsymbol{v}}$ is computed as

$$g^{x_1} := (g^{\phi})^{-4\chi_8}, \quad g^{x_2} := g^{5(\chi_6-\chi_8)}, \quad g^{x_3} := (g^{h_{\rho(1)}})^{-5(\chi_6-\chi_8)}, \quad g^{x_4} := g^{-4\chi_6+3\chi_8},$$
$$g^{x_5} := (g^{h_{\rho(2)}})^{4\chi_6-3\chi_8}, \quad g^{x_6} := g^{\chi_6}, \quad g^{x_7} := (g^{h_{\rho(3)}})^{-\chi_6}, \quad g^{x_8} := g^{\chi_8}, \quad g^{x_9} := (g^{h_{\rho(4)}})^{-\chi_8}.$$

4 Security proof of the proposed predicate signature

4.1 Signer privacy

Theorem 4.1. *Our proposed PS scheme in Section 3.3 is perfectly private (Definition 2.8).*

Proof. For $\boldsymbol{s} := (s_0, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}$, we define $(\mathbf{V}_{\text{M}})_{\alpha s_0} := \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1+1} \mid \langle \boldsymbol{v}, \boldsymbol{c}_y^{\text{M}}(\boldsymbol{s}, \boldsymbol{h}_{\text{M}})\rangle = \alpha s_0\}$. One can easily check that for arbitrary $\tilde{\boldsymbol{v}} \in (\mathbf{V}_{\text{M}})_{\alpha s_0}$, $\tilde{\boldsymbol{v}} + (\mathbf{V}_{\text{M}})^\perp = (\mathbf{V}_{\text{M}})_{\alpha s_0}$. Since the distribution of a signature for (m, y) is

$$\boldsymbol{\delta}_y = g^{\boldsymbol{v}+\boldsymbol{v}_{\text{sp}}} \cdot \boldsymbol{R}_3 \in \mathbb{G}^{\omega_1+1},$$

where $\boldsymbol{v} \in (\mathbf{V}_{\text{M}})_{\alpha s_0}$ for some $\boldsymbol{s} = (s_0, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}$, it is sufficient to prove that $\boldsymbol{v} + \boldsymbol{v}_{\text{sp}}$ is uniformly distributed over $(\mathbf{V}_{\text{M}})_{\alpha s_0}$ for each $\boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}$. Since $\boldsymbol{v}_{\text{sp}}$ is chosen uniformly and independently from $(\mathbf{V}_{\text{M}})^\perp$ and $\boldsymbol{v} + (\mathbf{V}_{\text{M}})^\perp = (\mathbf{V}_{\text{M}})_{\alpha s_0}$, we are done. □

4.2 The proof of adaptive unforgeability

To prove unforgeability of the proposed construction in Section 3.3, we apply a signature variant of the dual system methodology [37] deployed in [1]. This signature variant of dual system is similar to the style of [31, 33]. In this variant, the original unforgeability game is changed to the final game through some intermediate hybrid games. These changes are made under three subgroup decision problems and CMH or PMH-security of the underlying pair encoding scheme. To smooth hybrid arguments over the consecutive games, we use the natural restrictions defined in Conditions 3.1. We note that condition (2) is only used (in Lemma A.8) for reaching the final game from the previous game. In the final game, $\mathcal{V}_{\mathrm{INT}}$ of the verification text is sampled uniformly and independently from $\mathbb{G}_T$. Therefore, the forgery in the final game will be invalid. If v_1 and v_2 are respectively the number of key queries and signature queries made by $\mathscr{A}$, then the reduction cost is $\mathcal{O}(v_1 + v_2)$. We use the abbreviations "vText" and "sf-type", respectively, for verification text and semi-functional type. For all the games defined in Theorem 4.2, the following algorithms will be used to define normal verification text, and semi-functional verification text, keys and signatures:

- $\mathsf{SFSetup}(1^\kappa, \boldsymbol{j})$ runs $(\mathcal{PP}, \mathcal{MSK}) \leftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$ and, in addition, returns the semi-functional parameters $g_2 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_2}$, $\hat{\theta}_1, \hat{\theta}_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and $\hat{\boldsymbol{h}} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n$. We set $\hat{\boldsymbol{h}}_{\mathrm{M}} := (\hat{\theta}_1, \hat{\theta}_2, \hat{\boldsymbol{h}})$.
- $\mathsf{SFKeyGen}(\mathcal{PP}, \mathcal{MSK}, x, g_2, \mathrm{type}, \hat{\alpha}, \hat{\boldsymbol{h}})$ runs $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x, N)$ with $|\boldsymbol{k}_x| = m_1$, chooses $\boldsymbol{r}, \hat{\boldsymbol{r}} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$ and outputs the semi-functional key $\mathcal{SK}_x := (x, \boldsymbol{K}_x)$, where $\boldsymbol{K}_x$ is given by

$$\boldsymbol{K}_x := \begin{cases} g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})} \cdot g_2^{\boldsymbol{k}_x(0, \hat{\boldsymbol{r}}, \hat{\boldsymbol{h}})} \cdot \boldsymbol{R}_3 & \text{if type} = 1, \\ g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})} \cdot g_2^{\boldsymbol{k}_x(\hat{\alpha}, \hat{\boldsymbol{r}}, \hat{\boldsymbol{h}})} \cdot \boldsymbol{R}_3 & \text{if type} = 2, \\ g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})} \cdot g_2^{\boldsymbol{k}_x(\hat{\alpha}, \boldsymbol{0}, \boldsymbol{0})} \cdot \boldsymbol{R}_3 & \text{if type} = 3. \end{cases}$$

- $\mathsf{SFSign}(\mathcal{PP}, m, \mathcal{SK}_x, y, g_2, \mathrm{type})$ returns $\perp$ if $x \not\sim y$. It runs $\boldsymbol{\delta}_y \leftarrow \mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_x, y)$. Note that

$$\boldsymbol{\delta}_y = g^{\boldsymbol{v} + \boldsymbol{v}_{\mathrm{sp}}} \cdot \boldsymbol{R}_3 \quad \text{with } \boldsymbol{R}_3 \in \mathbb{G}_{p_3}^{\omega_1 + 1}.$$

  It picks $b, \iota \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and returns the semi-functional signature $\boldsymbol{\delta}_y \cdot g_2^{\hat{\boldsymbol{v}}}$, where $\hat{\boldsymbol{v}} \in \mathbb{Z}_N^{\omega_1 + 1}$ is given by

$$\hat{\boldsymbol{v}} := \begin{cases} (b, \iota, 0, \ldots, 0) & \text{if type} = 1, \\ (0, \iota, 0, \ldots, 0) & \text{if type} = 2. \end{cases}$$

- $\mathsf{VText}(\mathcal{PP}, m, y)$ runs $(\boldsymbol{c}_y, \omega_2) \leftarrow \mathsf{Enc2}(y, N)$, picks $\boldsymbol{s} := (s_0, \ldots, s_{\omega_2})$ and $\hat{\boldsymbol{s}} := (\hat{s}_0, \ldots, \hat{s}_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2 + 1}$ and computes $\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}}) := (c_0(s_0, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{G}^{\omega_1 + 1}$, where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, $\hbar := H(m, y)$ and $c_0(s_0, \boldsymbol{\theta}) := s_0(\theta_1 \hbar + \theta_2)$. It returns the verification text

$$\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := g_T^{\alpha s_0}, \mathcal{V}_y := g^{\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})}).$$

- $\mathsf{SFVText}(\mathcal{PP}, m, y, g_2, \mathrm{type}, \hat{\boldsymbol{h}}_{\mathrm{M}})$ is similar to VText, except it additionally computes

$$\boldsymbol{c}_y^{\mathrm{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathrm{M}}) := (c_0(\hat{s}_0, \hat{\boldsymbol{\theta}}), \boldsymbol{c}_y(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}})) \in \mathbb{G}^{\omega_1 + 1},$$

  where $\hat{\boldsymbol{\theta}} := (\hat{\theta}_1, \hat{\theta}_2, \hbar)$ and $c_0(\hat{s}_0, \hat{\boldsymbol{\theta}}) := \hat{s}_0(\hat{\theta}_1 \hbar + \hat{\theta}_2)$. It returns the semi-functional verification text

$$\mathcal{V} := \begin{cases} (\mathcal{V}_{\mathrm{INT}} := g_T^{\alpha s_0},\ \mathcal{V}_y := g^{\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})} \cdot g_2^{\boldsymbol{c}_y^{\mathrm{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathrm{M}})}) & \text{if type} = 1, \\ (\mathcal{V}_{\mathrm{INT}} \xleftarrow{\mathrm{U}} \mathbb{G}_T,\ \mathcal{V}_y := g^{\boldsymbol{c}_y^{\mathrm{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})} \cdot g_2^{\boldsymbol{c}_y^{\mathrm{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathrm{M}})}) & \text{if type} = 2. \end{cases}$$

Theorem 4.2. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies Conditions 3.1, where* $\sim$ *is domain-transferable. Suppose* P *has CMH-security, the assumptions DSG1, DSG2 and DSG3 hold in* $\mathcal{J}$ *and* H *is a collision-resistant hash function. Then the proposed predicate signature scheme* PS *in Section 3.3 for the predicate* $\sim$ *is adaptively existential unforgeable (Definition 2.9).*

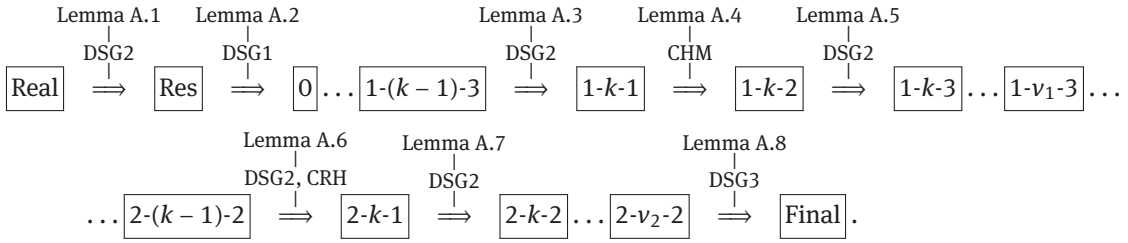
Proof. Suppose an adversary $\mathscr{A}$ makes at most v_1 key queries and v_2 signature queries. Then the security proof consists of a hybrid argument over a sequence of $3v_1 + 2v_2 + 4$ games. Let $\mathrm{Game}_{\mathrm{Real}}$ be the original Ad-EUF-CMA game of the predicate signature scheme. By applying hybrid arguments on $\mathrm{Game}_{\mathrm{Real}}$ through the sequence of intermediate games $\mathrm{Game}_{\mathrm{Res}}$, Game_0, $\{\mathrm{Game}_{1\text{-}k\text{-}1}, \mathrm{Game}_{1\text{-}k\text{-}2}, \mathrm{Game}_{1\text{-}k\text{-}3}\}_{k \in [v_1]}$ and

Game	Verification text	Key	Signature
Real $(x_i \nprec_N y^*)$	$\mathsf{VText}(m^*, y^*)$	$\mathsf{KeyGen}(x_i)$	$\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$
Res $\boxed{(x_i \nprec_{p_2} y^*)}$	$\mathsf{VText}(m^*, y^*)$	$\mathsf{KeyGen}(x_i)$	$\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$
0	$\boxed{\mathsf{SFVText}(m^*, y^*, g_2, 1, \hat{\boldsymbol{h}}_{\mathsf{M}})}$	$\mathsf{KeyGen}(x_i)$	$\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$
1-k-1 $(1 \le k \le v_1)$	$\mathsf{SFVText}(m^*, y^*, g_2, 1, \hat{\boldsymbol{h}}_{\mathsf{M}})$	$\hat{\alpha}_i \xleftarrow{\cup} \mathbb{Z}_N$ for all $i \in [k-1]$; $\mathsf{SFKeyGen}(x_i, g_2, 3, \hat{\alpha}_i, \mathbf{0})$ if $i < k$ $\boxed{\mathsf{SFKeyGen}(x_i, g_2, 1, 0, \hat{\boldsymbol{h}})}$ if $i = k$ $\mathsf{KeyGen}(x_i)$ if $i > k$	$\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$
1-k-2 $(1 \le k \le v_1)$	$\mathsf{SFVText}(m^*, y^*, g_2, 1, \hat{\boldsymbol{h}}_{\mathsf{M}})$	$\hat{\alpha}_i \xleftarrow{\cup} \mathbb{Z}_N$ for all $i \in [k]$; $\mathsf{SFKeyGen}(x_i, g_2, 3, \hat{\alpha}_i, \mathbf{0})$ if $i < k$ $\boxed{\mathsf{SFKeyGen}(x_i, g_2, 2, \hat{\alpha}_i, \hat{\boldsymbol{h}})}$ if $i = k$ $\mathsf{KeyGen}(x_i)$ if $i > k$	$\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$
1-k-3 $(1 \le k \le v_1)$	$\mathsf{SFVText}(m^*, y^*, g_2, 1, \hat{\boldsymbol{h}}_{\mathsf{M}})$	$\hat{\alpha}_i \xleftarrow{\cup} \mathbb{Z}_N$ for all $i \in [k]$; $\mathsf{SFKeyGen}(x_i, g_2, 3, \hat{\alpha}_i, \mathbf{0})$ if $i < k$ $\boxed{\mathsf{SFKeyGen}(x_i, g_2, 3, \hat{\alpha}_i, \mathbf{0})}$ if $i = k$ $\mathsf{KeyGen}(x_i)$ if $i > k$	$\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$
2-k-1 $(1 \le k \le v_2)$	$\mathsf{SFVText}(m^*, y^*, g_2, 1, \hat{\boldsymbol{h}}_{\mathsf{M}})$	$\hat{\alpha}_i \xleftarrow{\cup} \mathbb{Z}_N$ for all $i \in [v_1]$; $\mathsf{SFKeyGen}(x_i, g_2, 3, \hat{\alpha}_i, \mathbf{0})$	$\mathsf{SFSign}(m_j, \mathcal{SK}_{x_j}, y_j, g_2, 2)$ if $j < k$ $\boxed{\mathsf{SFSign}(m_j, \mathcal{SK}_{x_j}, y_j, g_2, 1)}$ if $j = k$ $\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$ if $j > k$
2-k-2 $(1 \le k \le v_2)$	$\mathsf{SFVText}(m^*, y^*, g_2, 1, \hat{\boldsymbol{h}}_{\mathsf{M}})$	$\hat{\alpha}_i \xleftarrow{\cup} \mathbb{Z}_N$ for all $i \in [v_1]$; $\mathsf{SFKeyGen}(x_i, g_2, 3, \hat{\alpha}_i, \mathbf{0})$	$\mathsf{SFSign}(m_j, \mathcal{SK}_{x_j}, y_j, g_2, 2)$ if $j < k$ $\boxed{\mathsf{SFSign}(m_j, \mathcal{SK}_{x_j}, y_j, g_2, 2)}$ if $j = k$ $\mathsf{Sign}(m_j, \mathcal{SK}_{x_j}, y_j)$ if $j > k$
Final	$\boxed{\mathsf{SFVText}(m^*, y^*, g_2, 2, \hat{\boldsymbol{h}}_{\mathsf{M}})}$	$\hat{\alpha}_i \xleftarrow{\cup} \mathbb{Z}_N$ for all $i \in [v_1]$; $\mathsf{SFKeyGen}(x_i, g_2, 3, \hat{\alpha}_i, \mathbf{0})$	$\mathsf{SFSign}(m_j, \mathcal{SK}_{x_j}, y_j, g_2, 2)$

Table 1: The description of hybrid games used in the security proof.

$\{\mathrm{Game}_{2\text{-}k\text{-}1}, \mathrm{Game}_{2\text{-}k\text{-}2}\}_{k \in [v_2]}$, we finally reach $\mathrm{Game}_{\mathrm{Final}}$. $\mathrm{Game}_{\mathrm{Res}}$ is the same as $\mathrm{Game}_{\mathrm{Real}}$, except the natural restriction $x \nprec_N y^*$ is replaced by $x \nprec_{p_2} y^*$ for each key query x made by $\mathscr{A}$. Game_0 is just like $\mathrm{Game}_{\mathrm{Res}}$, except the vText is of sf-type 1. In $\mathrm{Game}_{1\text{-}k\text{-}\iota}$ $(1 \le \iota \le 3)$, the verification text is of sf-type 1, the first $(k-1)$ keys are of sf-type 3, the k-th one is of sf-type ι and the remaining keys are normal, and all the signatures are normal. In $\mathrm{Game}_{2\text{-}k\text{-}\iota}$ $(1 \le \iota \le 2)$, the verification text is of sf-type 1, all the keys are of sf-type 3 and the first $(k-1)$ signatures are of sf-type 2, the k-th signature is of sf-type ι and the remaining signatures are normal. $\mathrm{Game}_{\mathrm{Final}}$ is the same as $\mathrm{Game}_{2\text{-}v_2\text{-}2}$, except the vText is of sf-type 2. A concrete description of the games is given in Table 1, where we mention the exact distribution of verification text, keys and signatures. The expression in the box indicates the modification from the previous game. For simplicity, $\mathcal{PP}$ and $\mathcal{MSK}$ are omitted from the respective algorithms appearing in the table.

In $\mathrm{Game}_{\mathrm{Final}}$, the part $\mathcal{V}_{\mathrm{INT}}$ is chosen independently and uniformly at random from $\mathbb{G}_T$. This implies that the forgery will be invalid with respect to the vText. Therefore, the adversary $\mathscr{A}$ has no advantage in $\mathrm{Game}_{\mathrm{Final}}$. The outline of the hybrid arguments over the games is given below:



Using the lemmas referred to above (for details, see Appendix A), we have the reduction

$$\mathrm{Adv}^{\text{Ad-EUF-CMA}}_{\mathscr{A},\mathsf{PS}}(\kappa) \le \mathrm{Adv}^{\mathrm{DSG1}}_{\mathscr{B}_1}(\kappa) + (2\nu_1 + 2\nu_2 + 1)\mathrm{Adv}^{\mathrm{DSG2}}_{\mathscr{B}_2}(\kappa) + \nu_1\mathrm{Adv}^{\mathrm{CMH}}_{\mathscr{B}_3,\mathsf{P}}(\kappa) + \nu_2\mathrm{Adv}^{\mathrm{CRH}}_{\mathscr{B}_4}(\kappa) + \mathrm{Adv}^{\mathrm{DSG3}}_{\mathscr{B}_5}(\kappa),$$

where $\mathrm{Adv}^{\mathrm{CRH}}_{\mathscr{B}_4}(\kappa)$ is the advantage of $\mathscr{B}_4$ in breaking the collision-resistant property of H and $\mathscr{B}_1$, $\mathscr{B}_2$, $\mathscr{B}_3$, $\mathscr{B}_4$, $\mathscr{B}_5$ are PPT algorithms whose running times are the same as that of $\mathscr{A}$. □

Theorem 4.3. *Let* P *be a pair encoding scheme for a predicate ~ which satisfies Conditions 3.1, where ~ is domain-transferable. Suppose* P *has PMH-security, the assumptions DSG1, DSG2 and DSG3 hold in* $\mathfrak{J}$ *and H is a collision-resistant hash function. Then the proposed predicate signature scheme* PS *in Section 3.3 for the predicate ~ is adaptively existential unforgeable.*

Proof. The proof is similar to that of Theorem 4.2. The reduction of the proof is given by

$$\mathrm{Adv}^{\text{Ad-EUF-CMA}}_{\mathscr{A},\mathsf{PS}}(\kappa) \le \mathrm{Adv}^{\mathrm{DSG1}}_{\mathscr{B}_1}(\kappa) + (2\nu_1 + 2\nu_2 + 1)\mathrm{Adv}^{\mathrm{DSG2}}_{\mathscr{B}_2}(\kappa) + \nu_2\mathrm{Adv}^{\mathrm{CRH}}_{\mathscr{B}_3}(\kappa) + \mathrm{Adv}^{\mathrm{DSG3}}_{\mathscr{B}_4}(\kappa),$$

where $\mathscr{B}_1$, $\mathscr{B}_2$, $\mathscr{B}_3$ and $\mathscr{B}_4$ are PPT algorithms whose running times are the same as that of $\mathscr{A}$. □

5 Instantiations of predicate signature

In this section, we instantiate different predicate signature schemes from various pair encoding schemes. The different variants of PS with many new features which did not exist earlier in the literature are presented here. Also we show that some existing PS schemes can be obtained by applying our framework. If the underlying pair encoding scheme with either PMH or CMH-security satisfies the sufficient conditions (Conditions 3.1), then our construction of predicate signature in Section 3.3 guarantees signer privacy and adaptive unforgeability. For instantiations, we consider only the pair encoding schemes[6] presented in [1, 5, 39] as they have either PMH or CMH-security and satisfy the aforementioned conditions. Other reasons for considering the pair encoding schemes mainly from [1, 5, 39] are that they are available in ready-made forms, and many PS schemes with new features can be derived from them. In the following, we briefly describe the instantiations of predicate signature using the pair encodings of [1, 5, 39].

Our framework provides a predicate signature scheme for regular languages in key-policy and signature-policy forms. The KP-PS and SP-PS for regular languages are instantiated from [1, Pair Encoding Schemes 3 and 7], respectively. These are the first non-trivial practical schemes beyond ABS.

We can derive an unbounded KP-ABS with large universes from [1, Pair Encoding Scheme 4]. Here unbounded means there is no restriction on the sizes of policies and attribute sets and the repetition of attributes in a policy. An ABS with large universes will have a super-polynomial size attribute universe. The universe of attributes is considered to be $\mathbb{Z}_N$, and the size of the public parameters is constant. The only known adaptively unforgeable ABSs with large universes available in the literature are the construction of [29, 31]; among them, only the ABS of [29] has the feature *unbounded*. However, these constructions are known to have signature-policy form. Therefore, the proposed ABS scheme is the first unbounded KP-ABS with large universes which is unforgeable in the adaptive model. We can also instantiate an unbounded SP-ABS with large universes from the dual [5] of [1, Pair Encoding Scheme 4], but it is less efficient than the SP-ABS of [29].

We can achieve a KP-ABS with constant-size signatures using [1, Pair Encoding Scheme 5]. The unforgeability of the only known constant-size signature [3] for non-monotone access structures was proven in the selective model. Therefore, the proposed ABS scheme is the first ABS with constant-size signature which is existential unforgeable in the adaptive model. Similarly, by applying our framework on the dual [5] of [1, Pair Encoding Scheme 5], we achieve an adaptively unforgeable SP-ABS with constant-size keys.

6 Since the predicate encodings of [39] have a structure similar to the pair encodings of [1], w.l.o.g., we refer to the predicate encodings of [39] as pair encodings in the paper. All the pair encoding schemes of [39] are perfectly master-key hiding.

The authors of [1, 5] proposed new encryption schemes for policy over doubly spatial relation (see Section 2.5) in key-policy and ciphertext-policy forms. These predicate encryption schemes are called key policy over doubly spatial encryption (KP-DSE) and ciphertext policy over doubly spatial encryption (CP-DSE), respectively. These predicate encryption schemes work in a similar manner to ABE, except the equality relation is replaced by a doubly spatial relation [22]. The signature analogues of KP-DSE and CP-DSE are called key policy over doubly spatial signature (KP-DSS) and signature policy over doubly spatial signature (SP-DSS), respectively. If we apply our framework on Pair Encoding Scheme 6 and its dual, we can obtain KP-DSS and SP-DSS, respectively. Similar to KP-DSE (resp. CP-DSE), KP-DSS (resp. SP-DSS) generalizes the existing class KP-ABS (resp. SP-ABS).

By applying our framework on [1, Pair Encoding Schemes 8 and 9], we can obtain KP-ABS and SP-ABS with small universes, respectively, where a restriction is imposed only on the polices. Since the underlying pair encodings are perfectly master-key hiding, both the ABS schemes are cost free. The SP-ABS of [33] can be viewed by the proposed SP-ABS.

Attrapadung [1] constructed new cost-free ABE schemes with large universes in key-policy and ciphertext-policy forms. The KP-ABE and CP-ABE were constructed from Pair Encoding Schemes 12 and 13, respectively. These pair encoding schemes were constructed based on cover-free families [18, 24]. Analogously, by applying our framework on Pair Encoding Schemes 12 and 13, we obtain cost-free KP-ABS and SP-ABS with large universes. Unlike ABS with small universes, bounds on both the sizes of attribute sets and the sizes of access structures are imposed.

We also instantiate many other cost-free predicate signatures as follows. A doubly spatial signature (DSS) scheme (as a signature analogue of DSE [22]) can be derived using [1, Pair Encoding Scheme 14]. The signature analogue of negated spatial encryption [4] is called negated spatial signature (NSS). An NSS can be instantiated from [1, Pair Encoding Scheme 15]. Using the pair encodings of [39] for the inner-product predicate, we can obtain inner-product signature (IPS) schemes with constant-size keys and constant-size signatures, respectively. We can also instantiate non-zero inner-product signature (NIPS) schemes with constant-size keys and constant-size signatures, respectively, using the pair encodings of [39] for the non-zero inner-product predicate. We note that a non-zero inner-product predicate is a special case of a negated spatial predicate. We can also obtain a spatial signature scheme with constant-size signatures using the pair encoding of [39].

PS	Form	Feature	Pair encoding	SPES
PS	KP	Regular languages	PES 3 [1]	CMH
PS	SP	Regular languages	PES 7 [1]	CMH
ABS	KP	Unbounded, large universes	PES 4 [1]	CMH
ABS	SP	Unbounded, large universes	Dual [5] of PES 4 [1]	CMH
ABS	KP	Constant-size signatures	PES 5 [1]	CMH
ABS	SP	Constant-size keys	Dual [5] of PES 5 [1]	CMH
KP-DSS	KP	It generalizes KP-ABS	PES 6 [1]	CMH
SP-DSS	SP	It generalizes SP-ABS	Dual [5] of PES 6 [1]	CMH
ABS	KP	Cost free	PES 8 [1]	PMH
ABS	SP	Cost free	PES 10 [1]	PMH
ABS	KP	Cost free, large universes	PES 12 [1]	PMH
ABS	SP	Cost free, large universes	PES 13 [1]	PMH
IPS	NA	Cost free, constant-size signatures	PES [39]	PMH
IPS	NA	Cost free, constant-size keys	PES [39]	PMH
NIPS	NA	Cost free, constant-size signatures	PES [39]	PMH
NIPS	NA	Cost free, constant-size keys	PES [39]	PMH
SS	KP	Cost free, constant-size signatures	PES [39]	PMH
SS	SP	Cost free, constant-size keys	Dual [1] of PES [39]	PMH
DSS	NA	Cost free	PES 14 [1]	PMH
NSS	NA	Cost free	PES 15 [1]	PMH

Table 2: Instantiations of predicate signature using existing pair encodings.

A summary of the instantiations of the predicate signature using the pair encodings of [1, 5, 39] is provided in Table 2. The abbreviations NA, KP, SP, PES and SPES stand for not applicable, key policy, signature policy, pair encoding scheme and security of pair encoding scheme, respectively. All the pair encodings shown in Table 2 are either perfectly (PMH) secure or computationally (both SMH and CMH) secure. The rightmost column stands for the security of the corresponding pair encoding scheme. The security given in Table 2 is used for unforgeability of the predicate signatures. The notations DSS, KP-DSS, SP-DSS, IPS, NIPS, SS and NSS respectively denote doubly spatial signature, key policy over DSS, signature policy over DSS, inner-product signature, non-zero IPS, spatial signature and negated spatial signature.

6 Conclusion

In this paper, for the first time, we showed that pair encodings provide adaptively unforgeable predicate signatures with prefect privacy. We have instantiated many schemes with new features using the existing pair encoding schemes, e.g., the first practical construction of PS schemes for regular languages, the first attribute-based signature scheme with constant-size signatures in the adaptive model, unbounded ABS with large universes in key-policy flavor, etc.

A Lemmas used in the proof of Theorem 4.2

Lemma A.1. $\mathrm{Game}_{\mathrm{Real}}$ *and* $\mathrm{Game}_{\mathrm{Res}}$ *are indistinguishable under DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{\mathrm{Real}}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{\mathrm{Res}}_{\mathscr{A},\mathsf{PS}}(\kappa)| \le \mathrm{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa).$$

Proof. Suppose an adversary can distinguish the games with a non-negligible probability. Then we will establish a PPT simulator $\mathscr{B}$ for breaking the DSG2 assumption with the same probability. An instance of DSG2, $(\mathbb{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0,1\}$, is given to $\mathscr{B}$. The only difference between the games $\mathrm{Game}_{\mathrm{Real}}$ and $\mathrm{Game}_{\mathrm{Res}}$ is that if x is a queried key index and y^* is a challenge associated data index, then $x \sim_{p_2} y^*$, but $x \not\sim_N y^*$. We show that the above scenario will not happen. In fact, from the soundness of domain transferability of $\sim$, we can find a factor F such that $p_2 \mid F \mid N$. There are three possibilities of F: (1) $F = p_2$, (2) $F = p_1p_2$ and (3) $F = p_2p_3$. We remark that the aforesaid cases are recognized using the parameters of the given instance of DSG2. Suppose $F = p_2$. Let $B := N/F = p_1p_3$, and then, by checking $T_\beta^B \stackrel{?}{=} 1_{\mathbb{G}}$, $\mathscr{B}$ can break the DSG2 assumption. Now suppose $F = p_1p_2$ or $F = p_2p_3$. Let $B := N/F$. If $B = p_3$, it computes $Y_2 := (W_2W_3)^B = W_2^{p_3}$, else $Y_2 := (Z_1Z_2)^B = Z_2^{p_1}$. In both case, we have $Y_2 \in \mathbb{G}_{p_2}$. Then, by checking $e(T_\beta, Y_2) \stackrel{?}{=} 1$, $\mathscr{B}$ can break the DSG2 assumption. □

Lemma A.2. $\mathrm{Game}_{\mathrm{Res}}$ *and* Game_0 *are indistinguishable under DSG1 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{\mathrm{Res}}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{0}_{\mathscr{A},\mathsf{PS}}(\kappa)| \le \mathrm{Adv}^{\mathrm{DSG1}}_{\mathscr{B}}(\kappa).$$

Proof. We establish a PPT simulator $\mathscr{B}$ which receives an instance of DSG1, $(\mathbb{J}, g, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0,1\}$, and depending on the distribution of β, it simulates either $\mathrm{Game}_{\mathrm{Res}}$ or Game_0.

Setup. $\mathscr{B}$ chooses $\alpha, \theta_1, \theta_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n$ and sets $\boldsymbol{h}_{\mathrm{M}} := (\theta_1, \theta_2, \boldsymbol{h})$. Let $H\colon \{0,1\}^* \to \mathbb{Z}_N$ be a hash function. Then it provides $\mathcal{PP} := (\mathbb{J}, g, g^{\boldsymbol{h}_{\mathrm{M}}}, g_T^\alpha := e(g,g)^\alpha, Z_3, H)$ to $\mathscr{A}$ and keeps $\mathcal{MSK} := (\alpha)$ to itself. It implicitly sets $\hat{\boldsymbol{h}}_{\mathrm{M}} := \boldsymbol{h}_{\mathrm{M}} \bmod p_2$. By the Chinese remainder theorem (CRT), $\hat{\boldsymbol{h}}_{\mathrm{M}}$ is independent from $\boldsymbol{h}_{\mathrm{M}} \bmod p_1$, so $\hat{\boldsymbol{h}}_{\mathrm{M}}$ is perfectly distributed.

Query phase. This consists of the following queries in adaptive manner:

- $\mathsf{KeyGen}(x)$ is a query for a normal key. The algorithm $\mathscr{B}$ can handle the key query of $\mathscr{A}$ since $\mathcal{MSK}$ is known to it.

- $\mathsf{Sign}(m, x, y)$ returns $\perp$ if $x \nrightarrow y$. This is a query for a normal signature. The algorithm $\mathscr{B}$ can answer the query of $\mathscr{A}$ since it can construct $\mathcal{SK}_x$ using the $\mathcal{MSK}$ known to it.

Forgery. $\mathscr{A}$ outputs a signature $\boldsymbol{\delta}_{y^*}$ for (m^*, y^*). Then $\mathscr{B}$ prepares a vText for (m^*, y^*) as follows: It computes $\hbar^* := H(m^*, y^*)$, runs $(\boldsymbol{c}_{y^*}, \omega_2) \leftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}^{\mathrm{M}}_{y^*} := (c_0, \boldsymbol{c}_{y^*})$, then picks $\boldsymbol{s}' := (s'_0, \ldots, s'_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$. Finally, it computes the vText as

$$\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := e(g^{\alpha}, T_{\beta})^{s'_0},\ \boldsymbol{\mathcal{V}}_{y^*} := T_{\beta}^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})}).$$

It returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathrm{INT}}$, else 0.

Analysis. We will show that all the objects are perfectly distributed as required. The algorithm $\mathscr{B}$ implicitly sets $g^{t_1} := T_{\beta}|_{\mathbb{G}_{p_1}}$ and, for $\beta = 1$, $g_2^{t_2} := T_{\beta}|_{\mathbb{G}_{p_2}}$. Then, by linearity of P, we have

$$g^{t_1 \boldsymbol{c}^{\mathrm{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})} = g^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(t_1 \boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})} \quad \text{and} \quad g_2^{t_2 \boldsymbol{c}^{\mathrm{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})} = g_2^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(t_2 \boldsymbol{s}', \hat{\boldsymbol{h}}_{\mathrm{M}})}.$$

It implicitly sets $\boldsymbol{s} := t_1 \boldsymbol{s}' \bmod p_1$ and, for $\beta = 1$, $\hat{\boldsymbol{s}} := t_2 \boldsymbol{s}' \bmod p_2$. By CRT, $\boldsymbol{s}' \bmod p_1$ is independent from $\boldsymbol{s}' \bmod p_2$, and therefore $\boldsymbol{s}$ and $\hat{\boldsymbol{s}}$ are perfectly distributed as required. Altogether, we have that the joint distribution of all the objects simulated by $\mathscr{B}$ is identical to that of $\mathrm{Game}_{\mathrm{Res}}$ if $\beta = 0$, else Game_0. □

Lemma A.3. $\mathrm{Game}_{1\text{-}(k-1)\text{-}3}$ *and* $\mathrm{Game}_{1\text{-}k\text{-}1}$ *are indistinguishable under DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{1\text{-}(k-1)\text{-}3}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{1\text{-}k\text{-}1}_{\mathscr{A},\mathsf{PS}}(\kappa)| \le \mathrm{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) \quad \textit{for } 1 \le k \le \nu_1.$$

Proof. We establish a PPT simulator $\mathscr{B}$ which receives an instance of DSG2, $(\mathcal{J}, g, Z_1 Z_2, W_2 W_3, Z_3, T_{\beta})$ with $\beta \xleftarrow{\mathrm{U}} \{0, 1\}$, and depending on the distribution of β, it simulates either $\mathrm{Game}_{1\text{-}(k-1)\text{-}3}$ or $\mathrm{Game}_{1\text{-}k\text{-}1}$.

Setup. $\mathscr{B}$ chooses $\alpha, \theta_1, \theta_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n$ and sets $\boldsymbol{h}_{\mathrm{M}} := (\theta_1, \theta_2, \boldsymbol{h})$. Let $H\colon \{0,1\}^* \to \mathbb{Z}_N$ be a hash function. Then it provides $\mathcal{PP} := (\mathcal{J}, g, g^{\boldsymbol{h}_{\mathrm{M}}}, g_T^{\alpha} := e(g, g)^{\alpha}, Z_3, H)$ to $\mathscr{A}$ and keeps $\mathcal{MSK} := (\alpha)$ to itself. It implicitly sets $\hat{\boldsymbol{h}}_{\mathrm{M}} := \boldsymbol{h}_{\mathrm{M}} \bmod p_2$. By CRT, $\hat{\boldsymbol{h}}_{\mathrm{M}}$ is independent from $\boldsymbol{h}_{\mathrm{M}} \bmod p_1$, so $\hat{\boldsymbol{h}}_{\mathrm{M}}$ is perfectly distributed.

Query phase. This consists of the following queries in adaptive manner:

- $\mathsf{KeyGen}(x)$: Let x_j be the j-th query key index. The algorithm $\mathscr{B}$ answers the key $\mathcal{SK}_{x_j}$ as follows:
  - If $j > k$, then $\mathscr{B}$ runs the KeyGen algorithm and gives the normal key to $\mathscr{A}$.
  - If $j < k$, then it is an sf-type 3 key, and $\mathscr{B}$ runs $(\boldsymbol{k}_{x_j}, m_2) \leftarrow \mathsf{Enc1}(x_j, N)$ with $|\boldsymbol{k}_{x_j}| = m_1$ and picks $\alpha'_j \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{r}_j \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$. It computes the sf-type 3 key as

$$\mathcal{SK}_{x_j} := g^{\boldsymbol{k}_{x_j}(\alpha, \boldsymbol{r}_j, \boldsymbol{h})} \cdot (W_2 W_3)^{\boldsymbol{k}_{x_j}(\alpha'_j, \boldsymbol{0}, \boldsymbol{0})} \cdot \boldsymbol{R}_3.$$

    It implicitly sets $\hat{\alpha}_j := w_2 \alpha'_j$, where $W_2 W_3 = g_2^{w_2} g_3^{w_3}$. So $\mathcal{SK}_{x_j}$ is a properly distributed sf-type 3 key.
  - If $j = k$, then it is either a normal or an sf-type 1 key, and $\mathscr{B}$ runs $(\boldsymbol{k}_{x_k}, m_2) \leftarrow \mathsf{Enc1}(x_k, N)$ with $|\boldsymbol{k}_{x_k}| = m_1$ and picks $\boldsymbol{r}'_k, \hat{\boldsymbol{r}}'_k \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$. It generates the following $\mathcal{SK}_{x_j}$ using T_{β} of the instance of DSG2:

$$\mathcal{SK}_{x_k} := g^{\boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}'_k, \boldsymbol{h})} \cdot T_{\beta}^{\boldsymbol{k}_{x_k}(0, \hat{\boldsymbol{r}}'_k, \boldsymbol{h})} \cdot \boldsymbol{R}_3.$$

    It implicitly sets $g^{t_1} := T_{\beta}|_{\mathbb{G}_{p_1}}$ and, for $\beta = 1$, $g_2^{t_2} := T_{\beta}|_{\mathbb{G}_{p_2}}$. Then, by linearity of P, we have

$$g^{\boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}'_k, \boldsymbol{h})} \cdot g^{t_1 \boldsymbol{k}_{x_k}(0, \hat{\boldsymbol{r}}'_k, \boldsymbol{h})} = g^{\boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}'_k + t_1 \hat{\boldsymbol{r}}'_k, \boldsymbol{h})} \quad \text{and} \quad g_2^{t_2 \boldsymbol{k}_{x_k}(0, \hat{\boldsymbol{r}}'_k, \boldsymbol{h})} = g_2^{\boldsymbol{k}_{x_k}(0, t_2 \hat{\boldsymbol{r}}'_k, \hat{\boldsymbol{h}})}.$$

    It implicitly sets $\boldsymbol{r}_k := \boldsymbol{r}'_k + t_1 \hat{\boldsymbol{r}}'_k$ and $\hat{\boldsymbol{r}}_k := t_2 \hat{\boldsymbol{r}}'_k$. Since $\boldsymbol{r}'_k$ and $\hat{\boldsymbol{r}}'_k$ are chosen uniformly and independently from $\mathbb{Z}_N^{m_2}$, then so are $\boldsymbol{r}_k$ and $\hat{\boldsymbol{r}}_k$. Therefore, $\mathcal{SK}_{x_k}$ is a perfectly distributed normal (resp. sf-type 1) key if $\beta = 0$ (resp. $\beta = 1$).
- $\mathsf{Sign}(m, x, y)$ returns $\perp$ if $x \nrightarrow y$. This is a query for a normal signature, and $\mathscr{B}$ can answer the query of $\mathscr{A}$ as $\mathcal{MSK}$ is known to it.

Forgery. $\mathscr{A}$ outputs a signature $\boldsymbol{\delta}_{y^*}$ for (m^*, y^*). Then $\mathscr{B}$ prepares a vText for (m^*, y^*) as follows: It computes $\hbar^* := H(m^*, y^*)$, runs $(\boldsymbol{c}_{y^*}, \omega_2) \leftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}^{\mathrm{M}}_{y^*} := (c_0, \boldsymbol{c}_{y^*})$, then picks

$\boldsymbol{s}' := (s'_0, \ldots, s'_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$. Finally, it computes

$$\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := e(g^{\alpha}, Z_1Z_2)^{s'_0},\ \boldsymbol{\mathcal{V}}_{y^*} := (Z_1Z_2)^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})}).$$

It returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathrm{INT}}$, else 0.

Analysis. We will show that all the objects are perfectly distributed as required. Let $Z_1Z_2 = g^{z_1}g_2^{z_2}$. Then, by linearity of P, we have

$$g^{z_1\boldsymbol{c}^{\mathrm{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})} = g^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(z_1\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})} \quad \text{and} \quad g_2^{z_2\boldsymbol{c}^{\mathrm{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})} = g_2^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(z_2\boldsymbol{s}', \hat{\boldsymbol{h}}_{\mathrm{M}})}.$$

The algorithm $\mathscr{B}$ implicitly sets $\boldsymbol{s} := z_1\boldsymbol{s}' \bmod p_1$ and $\hat{\boldsymbol{s}} := z_2\boldsymbol{s}' \bmod p_2$. By CRT, $\boldsymbol{s}' \bmod p_1$ is independent from $\boldsymbol{s}' \bmod p_2$, and therefore $\boldsymbol{s}$ and $\hat{\boldsymbol{s}}$ are perfectly distributed as required. Altogether, we have that the joint distribution of all the objects simulated by $\mathscr{B}$ is identical to that of $\mathrm{Game}_{1\text{-}(k-1)\text{-}3}$ if $\beta = 0$, else $\mathrm{Game}_{1\text{-}k\text{-}1}$. □

Lemma A.4. $\mathrm{Game}_{1\text{-}k\text{-}1}$ *and* $\mathrm{Game}_{1\text{-}k\text{-}2}$ *are indistinguishable under the CMH security of primitive pair encoding scheme* P. *That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{1\text{-}k\text{-}1}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{1\text{-}k\text{-}2}_{\mathscr{A},\mathsf{PS}}(\kappa)| \le \mathrm{Adv}^{\mathrm{CMH}}_{\mathscr{B},\mathsf{P}}(\kappa) \quad \textit{for } 1 \le k \le \nu_1.$$

Proof. Suppose $\mathscr{A}$ can distinguish $\mathrm{Game}_{1\text{-}k\text{-}1}$ and $\mathrm{Game}_{1\text{-}k\text{-}2}$ with non-negligible probability. Then we will construct a PPT simulator $\mathscr{B}$ for breaking the CMH security of P with the same probability.

Setup. The challenger $\mathcal{CH}$ of P gives $(g, g_2, g_3) \in \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3}$ to $\mathscr{B}$. Then $\mathscr{B}$ chooses $\alpha, \theta_1, \theta_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n$ and sets $\boldsymbol{h}_{\mathrm{M}} := (\theta_1, \theta_2, \boldsymbol{h})$. Let $H\colon \{0,1\}^* \to \mathbb{Z}_N$ be a hash function. Then it provides

$$\mathcal{PP} := (\mathcal{J}, g, g^{\boldsymbol{h}_{\mathrm{M}}},\ g_T^{\alpha} := e(g,g)^{\alpha},\ Z_3 := g_3, H)$$

to $\mathscr{A}$ and keeps $\mathcal{MSK} := (\alpha)$ and g_2 to itself.

Query phase. This consists of the following queries in adaptive manner:

- KeyGen(x): Let x_j be the j-th query key index. The algorithm $\mathscr{B}$ answers the key $\mathcal{SK}_{x_j}$ as follows:
  - If $j > k$, then $\mathscr{B}$ runs the KeyGen algorithm and gives the normal key to $\mathscr{A}$.
  - If $j < k$, then it is an sf-type 3 key. Using $\mathcal{PP}$, $\mathcal{MSK}$ and g_2, $\mathscr{B}$ can generate the required key.
  - If $j = k$, then it is either an sf-type 1 or an sf-type 2 key, and $\mathscr{B}$ runs $(\boldsymbol{k}_{x_k}, m_2) \leftarrow \mathsf{Enc1}(x_k, N)$ with $|\boldsymbol{k}_{x_k}| = m_1$ and picks $\boldsymbol{r}_k \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$. It makes a query with x_k to $\mathcal{CH}$. Let $\boldsymbol{T} := g_2^{\boldsymbol{k}_{x_k}(\beta, \hat{\boldsymbol{r}}_k, \hat{\boldsymbol{h}})}$ be the reply, where $\beta = 0$ or a random element from $\mathbb{Z}_N$. Then $\mathscr{B}$ returns the following key to $\mathscr{A}$:

    $$\mathcal{SK}_{x_k} := g^{\boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}_k, \boldsymbol{h})} \cdot \boldsymbol{T} \cdot \boldsymbol{R}_3.$$

    Therefore, $\mathcal{SK}_{x_j}$ is a perfectly distributed sf-type 1 key if $\beta = 0$, else sf-type 2.
- Sign(m, x, y) returns $\perp$ if $x \nrightarrow y$. This is a query for a normal signature, and $\mathscr{B}$ can answer the query of $\mathscr{A}$ as $\mathcal{MSK}$ is known to it.

Forgery. $\mathscr{A}$ outputs a signature $\boldsymbol{\delta}_{y^*}$ for (m^*, y^*). Then $\mathscr{B}$ prepares a vText for (m^*, y^*) as follows: It computes $\hbar^* := H(m^*, y^*)$, runs $(\boldsymbol{c}_{y^*}, \omega_2) \leftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}^{\mathrm{M}}_{y^*} := (c_0, \boldsymbol{c}_{y^*})$, then picks $\boldsymbol{s} := (s_0, \ldots, s_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$. Then it makes a query with y^* to $\mathcal{CH}$. Let $D := g_2^{\boldsymbol{c}_{y^*}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}})}$ be the reply. Finally, it computes a vText as

$$\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := e(g,g)^{\alpha s_0},\ \boldsymbol{\mathcal{V}}_{y^*} := g^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(\boldsymbol{s}, \boldsymbol{h}_{\mathrm{M}})} \cdot g_2^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathrm{M}})}), \quad \text{where } g_2^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathrm{M}})} := (g_2^{\hat{s}(\theta_1\hbar^* + \theta_2)}, D).$$

It returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathrm{INT}}$, else 0.

Analysis.

- *Correctness.* $\mathscr{B}$ follows the restriction of the CMH security game (while interacting with $\mathcal{CH}$) as long as $\mathscr{A}$ does so in the unforgeability game with $\mathscr{B}$. In fact, by natural restriction, for all key queries x made by $\mathscr{A}$, we have $x \nrightarrow_{p_2} y^*$, in particular, for the k-th query, $x_k \nrightarrow_{p_2} y^*$. Therefore, $\mathscr{B}$ does not violate the restriction of the CMH security game with $\mathcal{CH}$.

- *Perfectness.* By the assumption $c_{y^*,1}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}) = \hat{s}_0$, the first component of D is $g_2^{\hat{s}_0}$. So the first component of $g_2^{\boldsymbol{c}^{\mathrm{M}}_{y^*}(\hat{\boldsymbol{s}},\hat{\boldsymbol{h}}_{\mathrm{M}})}$ can be computed as $g_2^{\hat{s}_0(\theta_1 \hbar^* + \theta_2)} = (g_2^{\hat{s}_0})^{\theta_1 \hbar^* + \theta_2}$. The algorithm $\mathscr{B}$ implicitly sets

$$(\hat{\theta}_1, \hat{\theta}_2) := (\theta_1, \theta_2) \bmod p_2.$$

  By CRT, $(\hat{\theta}_1, \hat{\theta}_2)$ is independent from $(\theta_1, \theta_2) \bmod p_1$, and therefore $\mathcal{V}$ is a perfectly distributed sf-type 1 vText. Altogether, we have that the joint distribution of all the objects simulated by $\mathscr{B}$ is identical to that of $\mathrm{Game}_{1\text{-}k\text{-}1}$ if $\beta = 0$, else $\mathrm{Game}_{1\text{-}k\text{-}2}$. □

Lemma A.5. $\mathrm{Game}_{1\text{-}k\text{-}2}$ *and* $\mathrm{Game}_{1\text{-}k\text{-}3}$ *are indistinguishable under DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{1\text{-}k\text{-}2}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{1\text{-}k\text{-}3}_{\mathscr{A},\mathsf{PS}}(\kappa)| \le \mathrm{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) \quad \textit{for } 1 \le k \le \nu_1.$$

Proof. We establish a PPT simulator $\mathscr{B}$ which receives an instance of DSG2, $(\mathcal{J}, g, Z_1 Z_2, W_2 W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0,1\}$, and depending on the distribution of β, it simulates either $\mathrm{Game}_{1\text{-}k\text{-}2}$ or $\mathrm{Game}_{1\text{-}k\text{-}3}$. The description of the simulation is the same as that of Lemma A.3 except for answering the k-th key query. Below, we only describe the simulation of the k-th query.

The k-th key is either sf-type 2 or sf-type 3. The algorithm $\mathscr{B}$ runs $(\boldsymbol{k}_{x_k}, m_2) \leftarrow \mathsf{Enc1}(x_k, N)$ with $|\boldsymbol{k}_{x_k}| = m_1$ and picks $\boldsymbol{r}'_k, \hat{\boldsymbol{r}}'_k \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$. It generates the following $\mathcal{SK}_{x_k}$ using T_β of the instance of DSG2:

$$\mathcal{SK}_{x_k} := g^{\boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}'_k, \boldsymbol{h})} \cdot (W_2 W_3)^{\boldsymbol{k}_{x_k}(\alpha'_k, \boldsymbol{0}, \boldsymbol{0})} \cdot T_\beta^{\boldsymbol{k}_{x_j}(0, \hat{\boldsymbol{r}}'_k, \boldsymbol{h})} \cdot \boldsymbol{R}_3.$$

If $W_2 W_3 = g_2^{w_2} g_3^{w_3}$ and $T_\beta = g^{t_1} g_2^{t_2} g_3^{t_3}$ (for $\beta = 1$), then $\mathscr{B}$ implicitly sets $\hat{\alpha}_k := w_2 \alpha'_k$, $\boldsymbol{r}_k := \boldsymbol{r}'_k + t_1 \hat{\boldsymbol{r}}'_k$ and $\hat{\boldsymbol{r}}_k := t_2 \hat{\boldsymbol{r}}'_k$. Note that here we use the linearity and param-vanishing properties of the pair encoding P. Since $\boldsymbol{r}'_k$ and $\hat{\boldsymbol{r}}'_k$ are chosen uniformly and independently from $\mathbb{Z}_N^{m_2}$, then so are $\boldsymbol{r}_k$ and $\hat{\boldsymbol{r}}_k$. Therefore, $\mathcal{SK}_{x_k}$ is a perfectly distributed sf-type 2 (resp. sf-type 3) key if $\beta = 1$ (resp. $\beta = 0$). □

Lemma A.6. $\mathrm{Game}_{2\text{-}(k-1)\text{-}2}$ *and* $\mathrm{Game}_{2\text{-}k\text{-}1}$ *are indistinguishable under DSG2 assumption and the collision-resistant property of* H*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{2\text{-}(k-1)\text{-}2}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{2\text{-}k\text{-}1}_{\mathscr{A},\mathsf{PS}}(\kappa)| \le \mathrm{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathrm{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa) \quad \textit{for } 1 \le k \le \nu_2.$$

Proof. We establish a PPT simulator $\mathscr{B}$ which receives an instance of DSG2, $(\mathcal{J}, g, Z_1 Z_2, W_2 W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0,1\}$, and depending on the distribution of β, it simulates either $\mathrm{Game}_{2\text{-}(k-1)\text{-}2}$ or $\mathrm{Game}_{2\text{-}k\text{-}1}$.

Setup. This is the same as for Lemma A.3.

Query phase. This consists of the following queries in adaptive manner:

- $\mathsf{KeyGen}(x)$: Here all the keys are of sf-type 3, and the simulation of the keys is the same as that of the sf-type 3 keys of Lemma A.3.
- $\mathsf{Sign}(m, x, y)$ returns $\perp$ if $x \nvdash y$. Let (m_j, x_j, y_j) be the j-th signature query made by $\mathscr{A}$. Then $\mathscr{B}$ answers the signature $\boldsymbol{\delta}_{y_j}$ as follows:
  - If $j > k$, it is a normal signature, and $\mathscr{B}$ can answer the queries of $\mathscr{A}$ as $\mathcal{MSK}$ is known to it.
  - If $j < k$, it is an sf-type 2 signature, and $\mathscr{B}$ first computes the normal signature $\boldsymbol{\delta}_{y_j}$, picks $\iota'_j \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and then returns

$$\tilde{\boldsymbol{\delta}}_{y_j} := \boldsymbol{\delta}_{y_j} \cdot (W_2 W_3)^{(0, \iota'_j, 0, \ldots, 0)}.$$

    If $W_2 W_3 = g_2^{w_2} g_3^{w_3}$, then $\mathscr{B}$ implicitly sets $\iota_j := w_2 \iota'_j$. So $\tilde{\boldsymbol{\delta}}_{y_j}$ is a properly distributed sf-type 2 signature.
  - If $j = k$, it is either a normal signature or an sf-type 1 signature. Then $\mathscr{B}$ runs $(\boldsymbol{k}_{x_k}, m_2) \leftarrow \mathsf{Enc1}(x_k, N)$ and $\mathsf{Pair}(x_k, y_k) \to \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1}$ and picks $\boldsymbol{v}_{\mathrm{sp}} \xleftarrow{\mathrm{U}} (\mathbf{V}_{\mathrm{M}})^\perp$, $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{\omega_1 + 1}$. It computes $\hbar_k := H(m_k, y_k)$ and then returns the signature

$$\boldsymbol{\delta}_{y_k} := g^{(0, \boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E})} \cdot g^{\boldsymbol{v}_{\mathrm{sp}}} \cdot T_\beta^{(-1, 0, \ldots, 0)} \cdot T_\beta^{(0, \theta_1 \hbar_k + \theta_2, \ldots, 0)} \cdot \boldsymbol{R}_3.$$

    Let $g^\tau := T_\beta|_{\mathbb{G}_{p_1}}$ and, for $\beta = 1$, $g_2^{t_2} := T_\beta|_{\mathbb{G}_{p_2}}$. Then the $\mathbb{G}_{p_1}$ component of $\boldsymbol{\delta}_{y_k}$ can be written as $g^{\boldsymbol{\nu} + \boldsymbol{\nu}_{\mathrm{sp}}}$, where $\boldsymbol{\nu} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E})$ and $\boldsymbol{\psi} := (\tau(\theta_1 \hbar_k + \theta_2), 0, \ldots, 0)$. If $\beta = 1$, the $\mathbb{G}_{p_2}$ component

of $\boldsymbol{\delta}_{y_k}$ is expressed as $g_2^{\hat{\boldsymbol{v}}}$, where $\mathscr{B}$ implicitly sets $b := -t_2 \bmod p_2$ and $\iota := t_2(\theta_1\hbar_k + \theta_2) \bmod p_2$. Since $\theta_1\hbar_k + \theta_2 \bmod p_1$ is independent from $\theta_1\hbar_k + \theta_2 \bmod p_2$ by CRT, therefore $\boldsymbol{\delta}_{y_k}$ is a perfectly distributed signature unless some correlation with vText is found later.

Forgery. $\mathscr{A}$ outputs a signature $\boldsymbol{\delta}_{y^*}$ for (m^*, y^*). Then $\mathscr{B}$ prepares a vText for (m^*, y^*) as follows: It computes $\hbar^* := H(m^*, y^*)$, runs $(\boldsymbol{c}_{y^*}, \omega_2) \leftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}_{y^*}^{\mathrm{M}} := (c_0, \boldsymbol{c}_{y^*})$, then picks $\boldsymbol{s}' := (s'_0, \ldots, s'_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$. It computes a vText as

$$\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := e(g^{\alpha}, Z_1Z_2)^{s'_0},\ \boldsymbol{\mathcal{V}}_{y^*} := (Z_1Z_2)^{\boldsymbol{c}_{y^*}^{\mathrm{M}}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})}).$$

It returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathrm{INT}}$, else 0.

Analysis. Now we mainly concentrate on the joint distribution of the k-th signature and vText as there may be a correlation between them. More precisely, we observe the distributional relation between

$$c_0^*(\hat{s}_0, \hat{\boldsymbol{\theta}}) := \hat{s}_0(\hat{\theta}_1\hbar^* + \hat{\theta}_2) := \tilde{s}_0(\theta_1\hbar^* + \theta_2) \bmod p_2 \quad \text{and} \quad c_{y^*,1}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}) := \hat{s}_0 := \tilde{s}_0 \bmod p_2$$

with $\tilde{s}_0 := z_1 s'_0$ involved in $\boldsymbol{c}_{y^*}^{\mathrm{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathrm{M}})$ of vText. Unfortunately, a similar kind of relation is found in $\hat{\boldsymbol{v}}$, viz., between $b := -t_2 \bmod p_2$ and $\iota := t_2(\theta_1\hbar_j + \theta_2) \bmod p_2$. But that correlation does not hamper our life: since H has collision resistant property and $(m_j, y_j) \neq (m^*, y^*)$, we have $\hbar_j \neq \hbar^*$. By applying the argument of [26], we have that $\theta_1\hbar_j + \theta_2$ and $\theta_1\hbar^* + \theta_2$ are independently and uniformly distributed[7] over $\mathbb{Z}_{p_2}$. Therefore, $(\tilde{s}_0, \tilde{s}_0(\theta_1\hbar^* + \theta_2)) \bmod p_2$ is uncorrelated from (b, ι). Altogether, we have that the joint distribution of all the objects simulated by $\mathscr{B}$ is identical to that of $\mathrm{Game}_{2\text{-}(k-1)\text{-}2}$ if $\beta = 0$ else $\mathrm{Game}_{2\text{-}k\text{-}1}$. □

Lemma A.7. $\mathrm{Game}_{2\text{-}k\text{-}1}$ *and* $\mathrm{Game}_{2\text{-}k\text{-}2}$ *are indistinguishable under DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{2\text{-}k\text{-}1}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{2\text{-}k\text{-}2}_{\mathscr{A},\mathsf{PS}}(\kappa)| \leq \mathrm{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) \quad \textit{for } 1 \leq k \leq \nu_2.$$

Proof. We establish a PPT simulator $\mathscr{B}$ which receives an instance of DSG2, $(\mathbb{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0, 1\}$, and depending on the distribution of β, it simulates either $\mathrm{Game}_{2\text{-}k\text{-}1}$ or $\mathrm{Game}_{2\text{-}k\text{-}2}$. The simulation is similar to that of Lemma A.6 except for answering k-th signature query. Note that, in this case, we do not require the collision-resistant property of H. We only illustrate here the k-th signature. The k-th signature is either of sf-type 1 or sf-type 2. The algorithm $\mathscr{B}$ runs $(\boldsymbol{k}_{x_k}, m_2) \leftarrow \mathsf{Enc1}(x_k, N)$ and $\mathsf{Pair}(x_k, y_k) \to \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1}$, picks $\iota'_k \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{v}_{\mathrm{sp}} \xleftarrow{\mathrm{U}} (\mathbf{V}_{\mathrm{M}})^{\perp}$, $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{\omega_1+1}$. It computes $\hbar_k := H(m_k, y_k)$ and then returns the signature as

$$\boldsymbol{\delta}_{y_k} := g^{(0, \boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E})} \cdot g^{\boldsymbol{v}_{\mathrm{sp}}} \cdot T_\beta^{(-1,0,\ldots,0)} \cdot T_\beta^{(0,\theta_1\hbar_k+\theta_2,\ldots,0)} \cdot (W_2W_3)^{(0,\iota'_k,0,\ldots,0)} \cdot \boldsymbol{R}_3.$$

Let $W_2W_3 = g_2^{w_2}g_3^{w_3}$. Let $g^{\tau} := T_\beta|_{\mathbb{G}_{p_1}}$ and, for $\beta = 1$, $g_2^{t_2} := T_\beta|_{\mathbb{G}_{p_2}}$. Then the $\mathbb{G}_{p_1}$ component of $\boldsymbol{\delta}_{y_k}$ can be written as $g^{\boldsymbol{v}+\boldsymbol{v}_{\mathrm{sp}}}$, where $\boldsymbol{v} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E})$ and $\boldsymbol{\psi} := (\tau(\theta_1\hbar_k + \theta_2), 0, \ldots, 0)$. If $\beta = 1$ (resp. $\beta = 0$), the $\mathbb{G}_{p_2}$ component of $\boldsymbol{\delta}_{y_k}$ is expressed as $g_2^{\hat{\boldsymbol{v}}}$ with $\hat{\boldsymbol{v}} := (b, \iota, 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1+1}$, where $\mathscr{B}$ implicitly sets $b := -t_2 \bmod p_2$ (resp. $b := 0 \bmod p_2$) and $\iota := t_2(\theta_1\hbar_k + \theta_2) + w_2\iota'_k \bmod p_2$ (resp. $\iota := w_2\iota'_k \bmod p_2$). Therefore, $\boldsymbol{\delta}_{y_k}$ is a perfectly distributed sf-type 1 (resp. sf-type 2) signature if $\beta = 1$ (resp. $\beta = 0$). □

Lemma A.8. $\mathrm{Game}_{2\text{-}\nu_2\text{-}2}$ *and* $\mathrm{Game}_{\mathrm{Final}}$ *are indistinguishable under DSG3 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that*

$$|\mathrm{Adv}^{2\text{-}\nu_2\text{-}2}_{\mathscr{A},\mathsf{PS}}(\kappa) - \mathrm{Adv}^{\mathrm{Final}}_{\mathscr{A},\mathsf{PS}}(\kappa)| \leq \mathrm{Adv}^{\mathrm{DSG3}}_{\mathscr{B}}(\kappa).$$

Proof. We establish a PPT simulator $\mathscr{B}$ which receives an instance of DSG1, $(\mathbb{J}, g, g^{\alpha}Y_2, g^{s_0}W_2, g_2, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0, 1\}$, and depending on the distribution of β, it simulates either $\mathrm{Game}_{2\text{-}\nu_2\text{-}2}$ or $\mathrm{Game}_{\mathrm{Final}}$.

7 To apply [26], we require that $\hbar_j - \hbar^* \neq 0 \bmod p_2$. From $\hbar_j - \hbar^* \neq 0 \bmod N$, we have $\hbar_j - \hbar^* \neq 0 \bmod p$ for at least one p such that $p \in \{p_1, p_2, p_3\}$. One can show that $\hbar_j - \hbar^* \neq 0 \bmod p$ for all p with $p \in \{p_1, p_2, p_3\}$ assuming the factorization problem is hard. However, if $\hbar_j - \hbar^* = 0 \bmod p_2$, we can find a factor F of N with $p_2 \mid F$ and which leads to breaking the DSG2 assumption, a contradiction.

Setup. $\mathscr{B}$ chooses $\theta_1, \theta_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n$ and sets $\boldsymbol{h}_{\mathrm{M}} := (\theta_1, \theta_2, \boldsymbol{h})$. Let $H\colon \{0,1\}^* \to \mathbb{Z}_N$ be a hash function. Then it provides $\mathcal{PP} := (\mathcal{J}, g, g^{\boldsymbol{h}_{\mathrm{M}}}, g_T^{\alpha} := e(g, g^{\alpha} Y_2), Z_3, H)$ to $\mathscr{A}$ and implicitly sets $\hat{\boldsymbol{h}}_{\mathrm{M}} := \boldsymbol{h}_{\mathrm{M}} \bmod p_2$. By CRT, $\hat{\boldsymbol{h}}_{\mathrm{M}}$ is independent from $\boldsymbol{h}_{\mathrm{M}} \bmod p_1$, so $\hat{\boldsymbol{h}}_{\mathrm{M}}$ is perfectly distributed.

Query phase. This consists of the following queries in adaptive manner:

- $\mathsf{KeyGen}(x)$: It is an sf-type 3 key. The algorithm $\mathscr{B}$ runs $(\boldsymbol{k}_x, m_2) \leftarrow \mathsf{Enc1}(x)$, then picks $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$, $\hat{\alpha}' \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$. Finally, it returns
$$\mathcal{SK}_x := (g^{\alpha} Y_2)^{\boldsymbol{k}_x(1,\boldsymbol{0},\boldsymbol{0})} \cdot g^{\boldsymbol{k}_x(0,\boldsymbol{r},\boldsymbol{h})} \cdot g_2^{\boldsymbol{k}_x(\hat{\alpha}',\boldsymbol{0},\boldsymbol{0})} \cdot \boldsymbol{R}_3.$$
If $Y_2 = g_2^{y_2}$, then $\mathscr{B}$ implicitly sets $\hat{\alpha} := y_2 + \hat{\alpha}' \bmod p_2$, so $\mathcal{SK}_x$ is a perfectly distributed sf-type 3 key.
- $\mathsf{Sign}(m, x, y)$ returns $\perp$ if $x \not\sim y$. This is a query for an sf-type 2 signature. As above, $\mathscr{B}$ first creates the sf-type 3 key $\mathcal{SK}_x := (x, \boldsymbol{K}_x := g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})} \cdot g_2^{\boldsymbol{k}_x(\hat{\alpha},\boldsymbol{0},\boldsymbol{0})} \cdot \boldsymbol{R}_3)$, and then, using $\mathcal{SK}_x$, it can compute the sf-type 2 signature $\boldsymbol{\delta}_y$ as follows: It computes $\boldsymbol{\delta}_y := (g^{-\tau}, \boldsymbol{\Psi} \cdot \boldsymbol{K}_x^{\boldsymbol{E}}) \cdot g^{\boldsymbol{v}_{\mathrm{sp}}} \cdot g_2^{(0,\iota',0,\ldots,0)} \cdot \widetilde{\boldsymbol{R}}_3 \in \mathbb{G}^{\omega_1+1}$, where $\tau, \iota' \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\widetilde{\boldsymbol{R}}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{\omega_1+1}$, $\boldsymbol{\Psi} := g^{\boldsymbol{\psi}}$ with $\boldsymbol{\psi} := (\tau(\theta_1 \hbar + \theta_2), 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$, $\hbar := H(C_{\mathrm{cpa}})$, $\boldsymbol{v}_{\mathrm{sp}} \xleftarrow{\mathrm{U}} (\mathbf{V}_{\mathrm{M}})^{\perp}$ and $\boldsymbol{E} \leftarrow \mathsf{Pair}(x, y)$.
$$\begin{aligned}
\boldsymbol{\delta}_y|_{\mathbb{G}_{p_2}} &= (g_2^0, g_2^{\boldsymbol{k}_x(\hat{\alpha},\boldsymbol{0},\boldsymbol{0})\boldsymbol{E}}) \cdot g_2^{(0,\iota',0,\ldots,0)} \\
&= (g_2^0, g_2^{(*,0,\ldots,0)}) \cdot g_2^{(0,\iota',0,\ldots,0)} && \text{(by Conditions 3.1 (2))} \\
&= g_2^{(0,\iota,0,\ldots,0)} && \text{(where } \iota := * \cdot \iota').
\end{aligned}$$
This shows that $\boldsymbol{\delta}_y$ is a perfectly distributed sf-type 2 signature.

Forgery. $\mathscr{A}$ outputs a signature $\boldsymbol{\delta}_{y^*}$ for (m^*, y^*). Then $\mathscr{B}$ prepares a vText for (m^*, y^*) as follows: It computes $\hbar^* := H(m^*, y^*)$, runs $(\boldsymbol{c}_{y^*}, \omega_2) \leftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}_{y^*}^{\mathrm{M}} := (c_0, \boldsymbol{c}_{y^*})$, then picks $(s_1', \ldots, s_{\omega_2}') \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2}$ and sets $\boldsymbol{s}' := (1, s_1', \ldots, s_{\omega_2}') \in \mathbb{Z}_N^{\omega_2+1}$. Finally, it computes a vText as
$$\mathcal{V} := (\mathcal{V}_{\mathrm{INT}} := T_{\beta},\ \mathcal{V}_{y^*} := (g^{s_0} W_2)^{\boldsymbol{c}_{y^*}^{\mathrm{M}}(\boldsymbol{s}', \boldsymbol{h}_{\mathrm{M}})}).$$
It returns 1 if $e(\boldsymbol{\delta}_{y^*}, \mathcal{V}_{y^*}) = \mathcal{V}_{\mathrm{INT}}$, else 0.

The algorithm $\mathscr{B}$ implicitly sets $\boldsymbol{s} := s_0 \boldsymbol{s}' \bmod p_1$ and $\hat{\boldsymbol{s}} := s_0 \boldsymbol{s}' \bmod p_2$. By CRT, $\boldsymbol{s}' \bmod p_1$ is independent from $\boldsymbol{s}' \bmod p_2$, so $\boldsymbol{s}$ and $\hat{\boldsymbol{s}}$ are perfectly distributed as required. Therefore, $\mathcal{V}$ is a perfectly distributed sf-type 1 vText if $\beta = 0$, else sf-type 2.

Analysis. All the components simulated above are perfectly distributed as required. Therefore, the joint distribution of all the objects simulated by $\mathscr{B}$ is identical to that of $\mathrm{Game}_{2\text{-}\nu_2\text{-}2}$ if $\beta = 0$, else $\mathrm{Game}_{\mathrm{Final}}$. □

References

[1] N. Attrapadung, Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more, in: *Advances in cryptology—EUROCRYPT 2014*, Lecture Notes in Comput. Sci. 8441, Springer, Heidelberg (2014), 557–577.

[2] N. Attrapadung, Dual system encryption framework in prime-order groups, Cryptology ePrint Archive (2015), https://eprint.iacr.org/2015/390.pdf.

[3] N. Attrapadung, G. Hanaoka and S. Yamada, Conversions among several classes of predicate encryption and applications to ABE with various compactness tradeoffs, in: *Advances in Cryptology—ASIACRYPT 2015. Part I*, Lecture Notes in Comput. Sci. 9452, Springer, Heidelberg (2015), 575–601.

[4] N. Attrapadung and B. Libert, Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation, in: *Public Key Cryptography—PKC 2010*, Lecture Notes in Comput. Sci. 6056, Springer, Berlin (2010), 384–402.

[5] N. Attrapadung and S. Yamada, Duality in ABE: converting attribute based encryption for dual predicate and dual policy via computational encodings, in: *Topics in Cryptology—CT-RSA 2015*, Lecture Notes in Comput. Sci. 9048, Springer, Cham (2015), 87–105.

[6] M. Bellare and G. Fuchsbauer, Policy-based signatures, in: *Public-key Cryptography—PKC 2014*, Lecture Notes in Comput. Sci. 8383, Springer, Heidelberg (2014), 520–537.

[7] J. Blömer and G. Liske, Construction of fully CCA-secure predicate encryptions from pair encoding schemes, in: *Topics in Cryptology—CT-RSA 2016*, Lecture Notes in Comput. Sci. 9610, Springer, Cham (2016), 431–447.
[8] D. Boneh and X. Boyen, Efficient selective-ID secure identity-based encryption without random oracles, in: *Advances in Cryptology—EUROCRYPT 2004*, Lecture Notes in Comput. Sci. 3027, Springer, Berlin (2004), 223–238.
[9] D. Boneh and X. Boyen, Secure identity based encryption without random oracles, in: *Advances in Cryptology—CRYPTO 2004*, Lecture Notes in Comput. Sci. 3152, Springer, Berlin (2004), 443–459.
[10] D. Boneh, E.-J. Goh and K. Nissim, Evaluating 2-DNF formulas on ciphertexts, in: *Theory of Cryptography*, Lecture Notes in Comput. Sci. 3378, Springer, Berlin (2005), 325–341.
[11] D. Boneh, A. Sahai and B. Waters, Functional encryption: Definitions and challenges, in: *Theory of Cryptography*, Lecture Notes in Comput. Sci. 6597, Springer, Heidelberg (2011), 253–273.
[12] X. Boyen, Mesh signatures: How to leak a secret with unwitting and unwilling participants, in: *Advances in Cryptology—EUROCRYPT 2007*, Lecture Notes in Comput. Sci. 4515, Springer, Berlin (2007), 210–227.
[13] E. Boyle, S. Goldwasser and I. Ivan, Functional signatures and pseudorandom functions, in: *Public-key Cryptography—PKC 2014*, Lecture Notes in Comput. Sci. 8383, Springer, Heidelberg (2014), 501–519.
[14] S. Chatterjee, S. Mukherjee and T. Pandit, CCA-secure predicate encryption from pair encoding in prime order groups: Generic and efficient, in: *Progress in Cryptology—INDOCRYPT 2017*, Lecture Notes in Comput. Sci. 10698, Springer, Cham (2017), 85–106.
[15] D. Chaum and E. van Heyst, Group signatures, in: *Advances in Cryptology—EUROCRYPT '91volume*, Lecture Notes in Comput. Sci. 547, Springer, Berlin (1991), 257–265.
[16] J. Chen, R. Gay and H. Wee, Improved dual system ABE in prime-order groups via predicate encodings, in: *Advances in Cryptology—EUROCRYPT 2015. Part II*, Lecture Notes in Comput. Sci. 9057, Springer, Heidelberg (2015), 595–624.
[17] J. Chen and H. Wee, Doubly spatial encryption from DBDH, *Theoret. Comput. Sci.* **543** (2014), 79–89.
[18] P. Erdős, P. Frankl and Z. Füredi, Families of finite sets in which no set is covered by the union of *r* others, *Israel J. Math.* **51** (1985), no. 1–2, 79–89.
[19] A. Escala, J. Herranz and P. Morillo, Revocable attribute-based signatures with adaptive security in the standard model, in: *Progress in Cryptology—AFRICACRYPT 2011*, Lecture Notes in Comput. Sci. 6737, Springer, Heidelberg (2011), 224–241.
[20] J. Groth, R. Ostrovsky and A. Sahai, New techniques for noninteractive zero-knowledge, *J. ACM* **59** (2012), no. 3, Article ID 11.
[21] J. Groth and A. Sahai, Efficient non-interactive proof systems for bilinear groups, in: *Advances in Cryptology—EUROCRYPT 2008*, Lecture Notes in Comput. Sci. 4965, Springer, Berlin (2008), 415–432.
[22] M. Hamburg, Spatial encryption, Cryptology ePrint Archive (2011), https://eprint.iacr.org/2011/389.pdf.
[23] K. Hoffman and R. Kunze, *Linear Algebra*, Prentice-Hall Math. Ser., Prentice-Hall, Englewood Cliffs, 1961.
[24] R. Kumar, S. Rajagopalan and A. Sahai, Coding constructions for blacklisting problems without computational assumptions, in: *Advances in Cryptology—CRYPTO' 99*, Lecture Notes in Comput. Sci. 1666, Springer, Berlin (1999), 609–623.
[25] A. Lewko, T. Okamoto, A. Sahai, K. Takashima and B. Waters, Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption, in: *Advances in Cryptology—EUROCRYPT 2010*, Lecture Notes in Comput. Sci. 6110, Springer, Berlin (2010), 62–91.
[26] A. Lewko and B. Waters, New techniques for dual system encryption and fully secure HIBE with short ciphertexts, in: *Theory of Cryptography*, Lecture Notes in Comput. Sci. 5978, Springer, Berlin (2010), 455–479.
[27] J. Li, M. H. Au, W. Susilo, D. Xie and K. Ren, Attribute-based signature and its applications, in: *ACM Conference on Computer and Communications Security*, ACM, New York (2010), 60–69.
[28] H. Maji, M. Prabhakaran and M. Rosulek, Attribute-based signatures: Achieving attribute-privacy and collusion-resistance, Cryptology ePrint Archive (2008), http://eprint.iacr.org/2008/328.
[29] H. K. Maji, M. Prabhakaran and M. Rosulek, Attribute-based signatures, in: *Topics in Cryptology—CT-RSA 2011*, Lecture Notes in Comput. Sci. 6558, Springer, Heidelberg (2011), 376–392.
[30] T. Okamoto and K. Takashima, Hierarchical predicate encryption for inner-products, in: *Advances in Cryptology—ASIACRYPT 2009*, Lecture Notes in Comput. Sci. 5912, Springer, Berlin (2009), 214–231.
[31] T. Okamoto and K. Takashima, Efficient attribute-based signatures for non-monotone predicates in the standard model, in: *Public Key Cryptography—PKC 2011*, Lecture Notes in Comput. Sci. 6571, Springer, Heidelberg (2011), 35–52.
[32] T. Okamoto and K. Takashima, Decentralized attribute-based signatures, in: *Public-Key Cryptography—PKC 2013*, Lecture Notes in Comput. Sci. 7778, Springer, Berlin (2013), 125–142.
[33] T. Pandit, S. K. Pandey and R. Barua, Attribute-based signcryption: Signer privacy, strong unforgeability and IND-CCA2 security in adaptive-predicates attack, in: *Provable security*, Lecture Notes in Comput. Sci. 8782, Springer, Cham (2014), 274–290.
[34] Y. Sakai, N. Attrapadung and G. Hanaoka, Attribute-based signatures for circuits from bilinear map, in: *Public-key Cryptography—PKC 2016. Part I*, Lecture Notes in Comput. Sci. 9614, Springer, Cham (2016), 283–300.
[35] S. F. Shahandashti and R. Safavi-Naini, Threshold attribute-based signatures and their application to anonymous credential systems, in: *Progress in Cryptology—AFRICACRYPT 2009*, Lecture Notes in Comput. Sci. 5580, Springer, Berlin (2009), 198–216.

[36] G. Shaniqng and Z. Yingpei, Attribute-based signature scheme, in: *International Conference on Information Security and Assurance–ISA 2008*, IEEE Press, Piscataway (2008), 509–511.
[37] B. Waters, Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions, in: *Advances in Cryptology—CRYPTO 2009*, Lecture Notes in Comput. Sci. 5677, Springer, Berlin (2009), 619–636.
[38] B. Waters, Functional encryption for regular languages, in: *Advances in Cryptology—CRYPTO 2012*, Lecture Notes in Comput. Sci. 7417, Springer, Heidelberg (2012), 218–235.
[39] H. Wee, Dual system encryption via predicate encodings, in: *Theory of Cryptography*, Lecture Notes in Comput. Sci. 8349, Springer, Heidelberg (2014), 616–637.
[40] P. Yang, Z. Cao and X. Dong, Fuzzy identity based signature, Cryptology ePrint Archive (2008), https://eprint.iacr.org/2008/002.pdf.