On the Power of Pair Encodings: Frameworks for Predicate Cryptographic Primitives

Mridul Nandi and Tapas Pandit
Indian Statistical Institute, Kolkata
mridul@isical.ac.in,tapasgmmath@gmail.com

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 authors 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 for predicate signature (PS) from the pair encoding schemes. Our construction provides the signer privacy, and unforgeability in the adaptive-predicate model. Thereafter, we instantiate many PS schemes with new results, e.g., the first PS schemes for regular languages, the first attribute-based signature (ABS) scheme with constant-size signature in adaptive-predicate model, the unbounded ABS with large universes in key-policy flavor etc.

Following the CCA conversions of Yamada et al. (PKC 2011, 2012) and Nandi et al. (ePrint Archive: 2015/457), one can have CCA secure PE from CPA-secure PE if the primitive PE has either verifiability or delegation. We show that the fully secure CPA-construction of Attrapadung holds the verifiability if we assume a very simple condition on the underlying pair encoding scheme. The aforesaid approach degrades the performance of the resultant CCA-secure PE scheme. As an alternative, we provide a direct fully secure CCA-construction for PE from the pair encoding schemes. This costs an extra computation of group element in encryption and an extra pairing computation in decryption as compared to CPA-construction of Attrapadung.

The predicate signcryption (PSC) is a super class of the existing class, attribute-based signcryption (ABSC), where the confidentiality, unforgeability and signer privacy are well preserved. By combining the proposed frameworks for PS and PE, we provide a generic construction for PSC from the pair encodings. It achieves the perfect privacy, and the strong unforgeability and CCA security in the adaptive-predicates model. The construction has the support of "combined-setup", where the distribution of public parameters and keys in the (implicit) signature and encryption schemes are identical. The instantiations of the proposed PSC, provide many new schemes, e.g., the first PSC schemes for regular languages, the first ABSC with either constant-size signatures or constant-size keys, the unbounded ABSC with large universes in adaptive-predicates model etc.

1 Introduction

Predicate signature (PS) [4] is a signature analogue of predicate encryption (PE) [7], 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 ruled out by a binary relation $\sim$, called predicate relation defined over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are respectively called key space and associated data space. The attribute-based signature (ABS) [26] is a larger subclass of PS. Like ABS, the 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 $\mathcal{X}$ are more expressive 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, the unforgeability and signer privacy. The former ensures that the signatures are generated by a valid user and later protects from revealing the signer key index.

Attribute-based signcryption (ABSC) [17] is a natural extension of attribute-based encryption (ABE) and attribute-based signature (ABS) such that the confidentiality, unforgeability and signer privacy are well maintained. Like ABS, if the key is labeled with the set of attributes and the policies (signer policy and receiver policy) are associated with the signcryption, then the ABSC is known to be the key-policy attribute-based signcryption (KP-ABSC) and its dual form is called the signcryption-policy attribute-based signcryption (SCP-ABSC). In this paper, we start with predicate signcryption (PSC) which is a larger class of signcryptions containing the subclass, ABSC. Similar to ABSC, the predicate signcryptions are of two forms, the key-policy predicate signcryption (KP-PSC) and signcryption-policy predicate signcryption (SCP-PSC).

The concept of signcryption was introduced by Zheng [37]. Since, then many signcryption schemes [1, 28, 25] have been proposed. Among the three well known paradigms of [1], the paradigm "Commit then Encrypt and Sign ($\mathcal{C}t\mathcal{E}\&\mathcal{S}$)" runs faster as the implicit subroutines execute in parallel in signcrypt and unsigncrypt algorithms. The "combined-setup" in ABSC [31] and combining public-key schemes [20] allows to keep the distribution of public parameters and keys in the (implicit) signature and encryption schemes identical. Therefore, the schemes with "combined-setup" will be privileged as compared to the schemes with independent setup.

The dual system methodology of Waters [33] is a well known tool for constructing the predicate encryption scheme. But, for some predicates, e.g., regular languages, the adaptively secure predicate encryption were not known, even though their selectively-secure version was available. Therefore, for those class of predicates, the dual system technique of Waters [33] was unreachable. Recently, Attrapadung [2] introduced a new primitive, called pair encoding schemes which are sitting inside many PE schemes. They proposed a generic framework for adaptively secure predicate encryption, which captures the core idea of dual system technique [33]. They showed that by applying the generic approach on the pair encoding, the 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 authors constructed the first fully secure predicate encryption schemes for which only selectively secure schemes were known. They 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 [34] proposed the notion of predicate encodings which is exactly identical to the perfectly secure pair encodings of [2]. Some of the instantiations in [34] are similar to [2], viz., the KP-ABE, CP-ABE for small universe with improved efficiency and doubly-spatial encryption.

Our Contribution. In this paper, we provide the generic constructions for predicate signature, (CCA-secure) predicate encryption and predicate signcryption schemes from the pair encoding schemes. If the underlying pair encoding scheme with a least security[1], fulfills some (natural) conditions, then the PS and PSC schemes will achieve the perfect signer privacy, and the unforgeability in adaptive-predicate(s) model. But to ensure the adaptive-predicate(s) IND-CCA security of the PE and PSC schemes, we assume either both the computational security, CMH and SMH or the PMH security of the underlying pair encoding scheme. All the constructions are given in the setting of composite order bilinear groups. The unforgeability

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

(applicable to PS and PSC) and IND-CCA security (applicable to PE and PSC) are proven under the three subgroup assumptions, DSG1, DSG2, DSG3 and the extra hardness assumption(s) required for the CMH (and SMH)-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 predicate scheme is *cost free*. Through these generic constrictions what we achieved are summarized below:

- **Predicate Signature.** Since, all the pair encoding schemes of [2, 5] maintain the least security and satisfy the natural conditions therefore, the resultant predicate signature schemes are adaptive-predicate unforgeable and perfectly private.
  - (PS for Regular Languages.) The predicate signature schemes for regular languages in both the flavors, Key-policy and signature-policy are provided in this paper. Both the schemes support the large universe alphabet. To the best of our knowledge, these are the first non-trivial 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*. A dual version, unbounded CP-ABS with large universes is also proposed in this paper.
  - (Constant-size Signatures and Constant-size Keys.) Till to date, the only available ABS scheme with constant-size signature is known to be unforgeable in the selective-predicate model. We propose the first KP-ABS with constant-size signature, where the unforgeability is proven in adaptive-predicate model. A dual version, CP-ABS with constant-size keys is also provided in this paper.
  - (Policy over Doubly-Spatial Signature.) Similar to key-policy over doubly spatial encryption (KP-DSE) [2] and ciphertext-policy over doubly spatial encryption (CP-DSE) [5] , the new predicate signatures, key-policy over doubly-spatial signature (KP-DSS) and signature-policy over doubly-spatial signature (SP-DSS) are proposed in this paper. The new classes, KP-DSS and SP-DSS generalize the existing classes, KP-ABS and SP-ABS respectively.
  - (Cost Free ABS with Small Universe.) We present the small universes KP-ABS and SP-ABS schemes, where a restriction is imposed only on the polices. Since, the primitive pair encoding schemes are PMH secure, so the ABS schemes are *cost free*.
  - (Cost Free ABS with Large Universe.) Again analogous to new large universe ABE [2], the new *cost free* KP-ABS and CP-ABS schemes with large universes are presented. Unlike small universes construction, the bounds on both, the size of attribute set and size of access structure are imposed.
- **CCA Secure Predicate Encryption.** We obtain the adaptive-predicate IND-CCA predicate encryption schemes from the pair encoding schemes in two approaches:
  - (Traditional Approach.) We first show that if the underlying pair encoding scheme fulfills the condition (1) in **Conditions 3.2**, then the fully secure construction in sec.4.3 of [2] satisfies the *verifiability*. Then, by applying the CCA conversion technique [36, 29, 35], we obtain adaptive-predicate IND-CCA predicate encryption schemes.
  - (New Approach.) We first point out some drawbacks of the traditional approach:

1. The aforementioned approach may increases the length of key index and data index and the size of universe.
2. The checking in verifiability degrades the performance of decryption.
3. For new predicate scheme (in future), we may not know the concrete index-transformer [29, 36].

All these drawbacks keeping in mind, we provide a direct adaptive-predicate IND-CCA secure construction from the pair encoding schemes with having the fulfillment of the conditions (1) and (3) in **Conditions 3.2**. It has one extra group element in ciphertext and one extra pairing computation in decryption as compared to the CPA construction of [2].

Since, all the underlying pair encodings [2, 5] satisfy the conditions (1) and (3), therefore, using these approaches, we achieve CCA security of all the predicate encryptions found in [2, 5].

– **Predicate Signcryption.** Since, all the pair encoding schemes of [2, 5] either have both the computational security, CMH and SMH or the PMH security, and satisfy the natural conditions therefore, the resultant predicate signcryption schemes are adaptive-predicates strong unforgeable and perfectly private. All the predicate signcryption schemes have the combined-setup, non-repudiation and follow the new paradigm, "Commit then Encrypt and Sign then Sign $\mathcal{C}t\mathcal{E}\&\mathcal{S}t\mathcal{S}$" of [31]. To the best of our knowledge, all the results describes below are new except the SCP-ABSC with small universes construction of [31].

- (PSC for Regular Languages.) We present the predicate signcryptions for regular languages in both policies, key-policy (KP) and signcryption-policy (SCP) which support the large universe alphabet.
- (Unbounded ABSC.) An unbounded ABSC schemes with large universes in both flavors, KP and SCP are provided in this paper.
- (Constant-size Signcryptions and Constant-size keys.) A KP-ABSC scheme with constant-size signcryptions and an SCP-ABSC with constant-size keys are proposed in this paper.
- (Policy over Doubly-Spatial Signcryption.) In this paper, we present the new predicate signcryptions, a key-policy over doubly-spatial signcryption (KP-DSSC) and signcryption-policy over doubly-spatial signcryption (SCP-DSSC) which respectively generalize the existing classes, KP-ABSC and SCP-ABSC.
- (Cost Free ABSC with Small Universe.) Similar to signature, we propose the cost-free KP-ABSC and SCP-ABSC schemes with the support of small universes.
- (Cost Free ABSC with Large Universe.) Similar to signature, we propose the cost-free KP-ABSC and SCP-ABSC schemes with large universes.

Our Approach. In brief, the pair encoding scheme [2] consists of four deterministic algorithm, $\mathsf{Param}, \mathsf{Enc1}, \mathsf{Enc2}$ and Pair. Let $N \in \mathbb{N}$. $\mathsf{Param} \to n$ which 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 polynomial over $\mathbb{Z}_N$ with $|\boldsymbol{k}_x| = m_1$ and m_2 is 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 polynomial over $\mathbb{Z}_N$ with $|\boldsymbol{c}_y| = \omega_1$ and $\omega_2 + 1$ is length of the random coin $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}$. $\mathsf{Pair}(x, y) \to \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1}$. The correctness says 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 $\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}\boldsymbol{c}_y^\top(\boldsymbol{s}, \boldsymbol{h}) = \alpha s$.

– **Approach for PS.**

Fact 1.1. Before describing the central idea, we state the following two facts:

1. A signature in nothing but a diluted key for a policy y computed from an actual (strong) key $\mathcal{SK}_x$, where the message is binded.
2. To maintain the signer privacy, the signature is labeled with policy y, at least not labeled with the key index x.

Using the Fact 1.1 and the power of pair encoding, we develop the central idea as follow. Let $(N, \mathbb{G}, \mathbb{G}_T, e)$ be a bilinear groups and $g_T := e(g, g)$, where $g \in \mathbb{G}$. Let $\mathcal{SK}_x := g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})}$ be a key.

– $\mathsf{Signature\ Generation}$. For $x \sim y$, the diluted key $\boldsymbol{\delta}_y$ for a policy y is computed as $\boldsymbol{\delta}_y := g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}}$, where $\boldsymbol{E} \leftarrow \mathsf{Pair}(x, y)$. Now to ensure the signer privacy, we compose $\boldsymbol{\delta}_y$ by $g^{\boldsymbol{v}}$, i.e $\boldsymbol{\delta}_y = g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}+\boldsymbol{v}}$, where $\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1}$ is a random coin for the signer privacy such that $\boldsymbol{v}\boldsymbol{c}_y^\top(\boldsymbol{s}, \boldsymbol{h}) = 0$ for all $\boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}$. The computational feasibility of such $\boldsymbol{v}$ is ensured by imposing some natural conditions on the pair encoding schemes and the hardness of factorization problem.

– $\mathsf{Signature\ Verification}$. The verification process is probabilistic as it can be thought as a combination of encryption and decryption. Since, a signature is a poor or diluted key, so verifying a signature is nothing but the 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 (it is almost like ciphertext, where some randomness involved) $\mathcal{V} := (\mathcal{V}_{\mathsf{INT}} := g_T^{\alpha s}, \boldsymbol{\mathcal{V}}_y := g^{\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h})})$. The signature is accepted if $e(\boldsymbol{\delta}_y, \boldsymbol{\mathcal{V}}_y) = \mathcal{V}_{\mathsf{INT}}$ else rejected.

– $\mathsf{Correctness}$. For $x \sim y$, $e(\boldsymbol{\delta}_y, \boldsymbol{\mathcal{V}}_y) = e(g, g)^{(\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}+\boldsymbol{v})\boldsymbol{c}_y^\top(\boldsymbol{s},\boldsymbol{h})} = e(g, g)^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}\boldsymbol{c}_y^\top(\boldsymbol{s},\boldsymbol{h})} = g_T^{\alpha s}$, where the last equality is obtained from the correctness of pair encoding scheme.

The signature $\boldsymbol{\delta}_y$ described here is a core part of the actual signature proposed in sec.3.6, where some other components are to be added. E.g., for binding the message (m, y) to the signature, a component $g^{\tau(\theta_1\hbar+\theta_2)}$ to be added to $g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}}$ while generating the signature, where τ is a randomness, $\hbar := H(m, y)$ and H is a collision resistant hash. Accordingly the verification text is to be changed (for details, refer to sec.3.6). However, this prevents to alter a given signature for (m, y) to an another signature for (m', y') unless $(m, y) = (m', y')$.

To program the above central idea in our framework for PS with the assurance of adaptive security, we used the composite order bilinear groups. In this setup, our framework captures the dual system methodology of Waters [33] as a signature analogue of [2], where the hybrid arguments over the games follow the style of [30, 31]. In this style, we consider the semi-functional (mimic) forms of the original stuffs, the verification text, signatures and keys. Through the hybrid arguments, we finally reach to a game, where the $\mathcal{V}_{\mathsf{INT}}$ is chosen independently and uniformly random from $\mathbb{G}_T$ which implies that the forgery will be invalid with respect to the verification text $\mathcal{V}$. To simulate all stuffs perfectly, we assume some conditions on the underlying pair encoding schemes. To the best of our knowledge, most of the pair encodings (in fact, all the pair encoding of [2, 5]) satisfy these conditions.

– **Approach for CCA secure PE.** The above message binding idea gives the abstraction for (direct construction) CCA secure PE. Suppose C_{cpa} be the ciphertext of the CPA-construction of [2]. Similar to message binder above, we compute an additional components $g^{s(\theta_1\hbar+\theta_2)}$, where $\hbar := H(C_{\mathsf{cpa}})$ and s is a randomness involved in C_{cpa}. The new component, $g^{s(\theta_1\hbar+\theta_2)}$ is natural replacement for one-time signature in traditional approach [29, 35], where s plays the role of a signing key of the one-time signature. The adaptive security are obtained by following the dual system style of [2] which incorporates the dual system methodology of [33].

– **Approach for PSC.** This combines the approaches for PS and PE (for detail, refer to sec.6).

Related Work.
Attribute-based signature. Maji, Prabhakaran, and Rosulek [26] presented efficient ABS scheme for monotone span programs, but the unforgeability was proven in the generic group model. Later, the authors [27] proposed a general framework for ABS using the credential bundle and NIWI scheme as primitives. The unforgeability of construction was proven in the standard model using the security of credential bundle and soundness of NIWI scheme. The perfect witness hiding of the NIWI scheme provides the signer privacy in information-theoretic sense. As pointed out in [30] that it is much less efficient as compared to [26], since former uses the Groth-Sahai NIZK protocols [19] as building blocks. Though, the performance of the ABS construction [30] defeats the same of [27] but, the size of the public parameters is linear to the size of sub-universe and a bound is set on the number of times a attribute could appear in a policy. All the aforementioned ABS schemes have only the flavor of signature-policy, support large universe and the unforgeability is proven in adaptive-predicate model.

Functional Signature. Bellare and Fuchsbauer [6] proposed the notion of policy-based signatures which unifies the exists signatures, e.g., group signatures [11], mess signatures [8], attribute-based signatures [27] etc. For a policy-based signature (PBS) scheme, the authors defined the policy language, $\mathcal{L}$ to be any member of the complexity class, $\mathbf{NP}$. 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 be come from the complexity class, $\mathbf{P} \subseteq \mathbf{NP}$, then what we have is nothing but the predicate signatures, where the witness is computed in polynomial time. At the same time, Boyle et al. [9] introduced the concept of functional signatures, where a key is associated with a function f and that key 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.

Attribute-based signcryption. In recent years, many ABSC schemes [32, 31, 15, 12, 14] have been proposed to deal with various aspects, e.g., efficiency, expressibility, security feature, generality, model etc. Among them only the scheme of [31] has the support of combined setup, signer privacy, and confidentiality and unforgeability in the adaptive-predicates model. However, we show that this can be instantiated from our framework.

CCA secure Predicate Encryption. The techniques [10, 18, 35] available in the literature for converting CPA to CCA secure PE (even including CPA-secure IBE to CCA-secure PKE) in the standard model is the use of one-time signature (OTS). In this approach, first a pair of verification key and signing key, $(\mathsf{vk}, \mathsf{signk})$ is generated, then vk is embedded into the ciphertext C_{cpa} generated using the CPA-secure primitive PE scheme. Then, C_{cpa} is signed by signk to form the one-time signature, δ. So, the ciphertext for the CCA-secure PE scheme is of the form, $\mathsf{CT} := (C_{\mathsf{cpa}}, \delta, \mathsf{vk})$. The whole process makes sure that a new ciphertext (possibly well-formed or ill-formed but up to a certain extent) can not be constructed from a given ciphertext unless the signk is known. The generic technique of [35] is applicable to only ABE. Later, Attrapadung et al. [36] went beyond ABE to capture many other predicates for which no well known technique was known. Still they missed an important class, the predicate encryption for regular languages. Their technique is based on the verifiability of the primitive CPA secure public index scheme. Recently Nandi et al. [29] proposed a similar approach based on both the delegation and verifiability of the primitive CPA secure PE and instantiated many missing classes including PE for regular languages. The generic approach in [29] basically performs the index transformation for all the predicates including hidden index, but the instantiations that actually transform the CPA to CCA-secure PE are customized w.r.t the predicates.

Pair Encodings. In addition to the full PE construction [2] from the pair encodings, the authors showed a dual conversion for the pair encodings. If the source pair encoding, P is perfectly secure, then the dual of P, $\mathbb{D}(\mathsf{P})$ is also perfectly secure encoding. Using this conversion the full security of the dual of a PE, denoted by $\mathbb{D}(\mathsf{PE})$ is guaranteed if the underlying pair encoding, P has the perfect security. However, there are many PE schemes for which the perfectly secure encodings was not known, so the fully secure construction of their dual form was unsolved. Recently, Attrapadung et al. [5] showed that the same dual conversion of [2] actually works for the computationally secure encodings. More specifically, they proved (Theorem 4 and 5 of [5]) that if a pair encoding P for a predicate is normal and has (1,1)-CMH security, then the $\mathbb{D}(\mathsf{P})$ for the dual predicate is (1,1)-SMH secure and vice versa. By applying this conversion on the underlying pair encoding of previously proposed KP-ABE of [2], the authors achieved the first fully secure unbounded CP-ABE with short keys for Boolean formulae. They also provided a direct construction of pair encoding scheme for a certain dual predicate and show that it is (1,1)-CMH-secure and (1, poly)-SMH-secure. Therefore, the resulting ABE enjoys tighter reduction of $\mathcal{O}(q_1)$, where q_1 is the number of key queries in phases 1. What they considered is the CP-DSE, which is the dual of KP-DSE. Very recently, Chen, Gay, and Wee [13] and Attrapadung [3] proposed the new generic frameworks for achieving adaptively secure ABE in the prime order bilinear groups which are nothing but the prime order version of [34] and [2] respectively. The main difference between the frameworks of [13] and [3] is that the former deals with only the perfectly secure encodings, whereas the later can deal with the computationally secure encodings.

Organization. This paper is organized as follows. The basic notation, composite order bilinear groups, hardness assumptions and the syntax of commitment scheme and predicate family are given in sec.2. The syntax and security definition of predicate signature scheme and pair encoding scheme, the construction of predicate signature, and security of the construction are provided in sec.3. The instantiations of predicate signature, and the framework for predicate encryption and predicate signcryption are respectively given in sec.4, sec. 5 and sec.6. The syntax and security definition of predicate encryption and predicate signcryption are provided respectively in sec.B and sec. C.

2 Preliminaries

2.1 Notation

For a set X, $x \xleftarrow{\text{R}} X$ denotes that x is randomly picked from X according to the distribution R. Likewise, $x \xleftarrow{\text{U}} X$ indicates x is uniformly selected from X. For $a, b \in \mathbb{N}$, let $[a, b] := \{i \in \mathbb{N} : a \leq i \leq b\}$ and $[b] := [1, b]$. Through out this paper, ***bold*** marks indicate the vector notations. For $\boldsymbol{h} \in \mathbb{Z}_N^n$ and $p|N$, we define $\boldsymbol{h} \mod p := (h_1 \mod p, \ldots, h_n \mod p)$. For a matrix $\boldsymbol{M}$, the notations $\boldsymbol{M}^\top$ and M_{ij} respectively denote the transpose of M and entry of M at (i, j)-position. Let $\mathsf{Null}(\boldsymbol{M})$ represent the nullity of the matrix, $\boldsymbol{M}$. For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}_N^n$, we define $< \boldsymbol{x}, \boldsymbol{y} >:= \boldsymbol{x}\boldsymbol{y}^\top := \sum_{i=1}^n x_i.y_i$. Let $\mathbb{G}$ be a cyclic group of order N, then for $g \in \mathbb{G}$ and $\boldsymbol{h} \in \mathbb{Z}_N^n$, let $g^{\boldsymbol{h}} := (g^{h_1}, \ldots, g^{h_n})$. For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{G}^n$, the notation $\boldsymbol{x}.\boldsymbol{y}$ stands for component wise group operations and so, $\boldsymbol{x}.\boldsymbol{y} \in \mathbb{G}^n$. 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 $z_i := W_1^{E_{1i}}.W_2^{E_{2i}}.\ldots.W_n^{E_{ni}}$ and '.' is the group operation. If $\boldsymbol{W} = g^{\boldsymbol{w}}$, for $g \in \mathbb{G}$ and $\boldsymbol{w} \in \mathbb{Z}_N^n$, then we can write $\boldsymbol{W}^{\boldsymbol{E}} = g^{\boldsymbol{w}\boldsymbol{E}}$. For a bilinear groups $(N, \mathbb{G}, \mathbb{G}_T, e)$, let $g_T := e(g, g)$, where $g \in \mathbb{G}$ and Θ be the zero (identity) element of $\mathbb{G}$. For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{G}^n$, let $e(\boldsymbol{x}, \boldsymbol{y}) := \prod_{i=1}^n e(x_i, y_i)$. The abbreviation CBG stands for composite order bilinear groups. For three distinct primes, p_1, p_2 and p_3, a cyclic group $\mathbb{G}$ of order $N = p_1p_2p_3$, can be written as $\mathbb{G} = \mathbb{G}_{p_1}\mathbb{G}_{p_2}\mathbb{G}_{p_3}$, where $\mathbb{G}_{p_i}$'s are subgroups of $\mathbb{G}$. So, each element $x \in \mathbb{G}$ can be expressed as $x = x_1x_2x_3$, where $x_i \in \mathbb{G}_{p_i}$. For $x \in \mathbb{G}$, the notation $x\big|_{\mathbb{G}_{p_i}}$ means the projection of x over $\mathbb{G}_{p_i}$, i.e., $x_i = x\big|_{\mathbb{G}_{p_i}}$. For $\boldsymbol{x} \in \mathbb{G}^n$, let $\boldsymbol{x}\big|_{\mathbb{G}_{p_i}}$ denote $(x_1\big|_{\mathbb{G}_{p_i}}, \ldots, x_n\big|_{\mathbb{G}_{p_i}})$. The notation, $\mathbf{0}_{m\times n}$ stands for an $m \times n$ matrix with all the entries are 0. Let $str_1||\ldots||str_n$ denote the concatenation of the strings, $str_1, \ldots, str_n \in \{0,1\}^*$. $Alg_1 \parallel \ldots \parallel Alg_n$ stands for the parallel execution of the algorithms, $Alg_1, \ldots, Alg_n$.

2.2 Composite Order Bilinear Groups

Let $\mathcal{G}$ be an algorithm which takes 1^κ as a security parameter and returns a description of a composite order bilinear groups, $\mathscr{J} := (N := p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e)$, 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 $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a map such that

1. (Bilinear) $\forall g, h \in \mathbb{G}, a, b \in \mathbb{Z}_N, e(g^a, h^b) = e(g, h)^{ab}$
2. (Non-degenerate) $\exists g \in \mathbb{G}$ such that $e(g, g)$ has order N in $\mathbb{G}_T$

Let $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}$ and $\mathbb{G}_{p_3}$ respectively denote the subgroups of $\mathbb{G}$ of order p_1, p_2 and p_3. Let $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$ be arbitrary elements with $i \neq j$, then $e(h_i, h_j) = 1$. This property is called orthogonal property of $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}$.

2.3 Hardness Assumptions

We describe here three Decisional SubGroup (DSG) assumptions [24] for 3 primes, DSG1, DSG2 and DSS3 in composite order bilinear groups. Let $\mathscr{J} := (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \xleftarrow{\text{U}} \mathcal{G}(1^\kappa)$ be the common parameters for each assumptions.

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

[DSG2]. Let $g, Z_1 \xleftarrow{\text{U}} \mathbb{G}_{p_1},\ Z_2, W_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2},\ W_3, Z_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}, T_0 \xleftarrow{\text{U}} \mathbb{G}_{p_1p_3},\ T_1 \xleftarrow{\text{U}} \mathbb{G}$. Then set $\mathcal{D} := (\mathscr{J}, g, Z_1Z_2, W_2W_3, Z_3)$

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

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

$$\mathsf{Adv}_{\mathscr{A}}^{\text{DSGi}}(\kappa) = |Pr[\mathscr{A}(\mathcal{D}, T_0) = 1] - Pr[\mathscr{A}(\mathcal{D}, T_1) = 1]|$$

We say that the DSGi assumption holds if for every PPT algorithm $\mathscr{A}$, the advantage $\mathsf{Adv}_{\mathscr{A}}^{\text{DSGi}}(\kappa)$ is at most negligible in security parameter κ.

2.4 Commitment scheme

A non-interactive commitment scheme consists of three PPT algorithms - Setup, Commit and Open.

- Setup: It takes a security parameter κ and outputs a public commitment key $\mathcal{CK}$.
- Commit: It takes as input a message m, the public commitment key $\mathcal{CK}$ and returns a pair $(\mathsf{com}, \mathsf{decom})$, where com is a commitment of the message m and decom is the decommitment.
- Open: takes a pair $(\mathsf{com}, \mathsf{decom})$, the public commitment key $\mathcal{CK}$ as input and outputs m or $\perp$.

For correctness, it is required that[2] $\mathsf{Open}(\mathsf{Commit}(m)) = m$ for all message $m \in \mathcal{M}$, where $\mathcal{M}$ is the message space.

2.5 Security of Commitment

A commitment scheme is said to have hiding, binding and relaxed-binding properties if it satisfies the following respectively:

Hiding: For all PPT $\mathscr{A}$ the following is negligible:

$$\left| \Pr \begin{bmatrix} \mathcal{CK} \longleftarrow \mathsf{C.Setup}(1^\kappa),\ (m_0, m_1, st) \longleftarrow \mathscr{A}(\mathcal{CK}), & \\ b \xleftarrow{\mathrm{U}} \{0,1\}, (\mathsf{com}_b, \mathsf{decom}_b) \longleftarrow \mathsf{Commit}(\mathcal{CK}, m_b), & \end{bmatrix} : \mathscr{A}(\mathcal{CK}, st, \mathsf{com}_b) = b \right] - \frac{1}{2} \right|.$$

Binding: For all PPT $\mathscr{A}$ the following is negligible:

$$\Pr \begin{bmatrix} \mathcal{CK} \longleftarrow \mathsf{C.Setup}(1^\kappa),\ (\mathsf{com}, \mathsf{decom}, \mathsf{decom}') \longleftarrow \mathscr{A}(\mathcal{CK}), & \\ m \longleftarrow \mathsf{Open}(\mathsf{com}, \mathsf{decom}),\ m' \longleftarrow \mathsf{Open}(\mathsf{com}, \mathsf{decom}'), & : (m \neq m') \wedge (m, m' \neq \perp) \end{bmatrix}.$$

Relaxed-Binding: For all PPT $\mathscr{A}$ the following is negligible:

$$\Pr \begin{bmatrix} \mathcal{CK} \longleftarrow \mathsf{C.Setup}(1^\kappa),\ (m, st) \longleftarrow \mathscr{A}(\mathcal{CK}), (\mathsf{com}, \mathsf{decom}) \longleftarrow \mathsf{Commit}(m), & \\ \mathsf{decom}' \longleftarrow \mathscr{A}(\mathcal{CK}, st, \mathsf{com}, \mathsf{decom}),\ m' \longleftarrow \mathsf{Open}(\mathsf{com}, \mathsf{decom}'), & : (m \neq m') \wedge (m' \neq \perp) \end{bmatrix}.$$

Remark 2.1. It is immediate that the relaxed-binding property is weaker than the binding property.

2.6 Predicate Family

Let $\sim := \{(\sim_{\boldsymbol{j}}, \mathcal{X}_{\boldsymbol{j}}, \mathcal{Y}_{\boldsymbol{j}})\}_{\boldsymbol{j} \in \mathbb{N}^c}$ for some constant $c \in \mathbb{N}$ be the family of predicate tuples, $(\sim_{\boldsymbol{j}}, \mathcal{X}_{\boldsymbol{j}}, \mathcal{Y}_{\boldsymbol{j}})$, where $\mathcal{X}_{\boldsymbol{j}}$ and $\mathcal{Y}_{\boldsymbol{j}}$ are respectively key space and associative data space and $\sim_{\boldsymbol{j}}$: $\mathcal{X}_{\boldsymbol{j}} \times \mathcal{Y}_{\boldsymbol{j}} \to \{0,1\}$ is a predicate[3] function. 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 $x \not\sim_{\boldsymbol{j}} y$. The index of the family, $\boldsymbol{j}$ is called system parameter index which defines predicate tuple, $(\sim_{\boldsymbol{j}}, \mathcal{X}_{\boldsymbol{j}}, \mathcal{Y}_{\boldsymbol{j}})$. Here we are interested to design the predicate signature, predicate encryption and predicate signcryption over composite order bilinear groups (CBG) and let N be the order of the groups. This N basically 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, i.e., $j_1 = N$. For notational simplicity, we omit $\boldsymbol{j}$, simply write $(\sim_N, \mathcal{X}_N, \mathcal{Y}_N)$.

Definition 2.1. (Domain-transferable [2]). We say that $\sim$ is domain-transferable if for p divides N, the projection map $f_1 : \mathcal{X}_N \to \mathcal{X}_p$ and $f_2 : \mathcal{Y}_N \to \mathcal{Y}_p$ 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 $p|F|N$.

Remark 2.2. Attrapadung [2] showed that the equality predicate (for IBE) is domain-transferable. Since, all other predicates are defined through the equality predicate, all the predicates of [2] are domain-transferable.

[2]For brevity, we just omit $\mathcal{CK}$ in Open and Commit algorithm throughout this paper

[3]This predicate function $\sim_{\boldsymbol{j}}$ also called binary relation or predicate relation over $\mathcal{X}_{\boldsymbol{j}} \times \mathcal{Y}_{\boldsymbol{j}}$.

3 Framework for Predicate Signature

3.1 Definition of Predicate Signature

A predicate signature scheme for a predicate tuple family, $\sim$ consists of four PPT algorithms - Setup, KeyGen, Sign and Ver.

- Setup: It takes a security parameter κ and a system parameter index $\boldsymbol{j}$ as input, outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$. Let $(\sim_N, \mathcal{X}_N, \mathcal{Y}_N)$ be the predicate tuple corresponding to the index, $\boldsymbol{j} = (N, \ldots,)$. From now onwards we ignore N, just write $(\sim, \mathcal{X}, \mathcal{Y})$ and N will be understood from the context.
- KeyGen: It takes as input public parameters $\mathcal{PP}$, master secret $\mathcal{MSK}$ and a key index $x \in \mathcal{X}$ and outputs a secret key $\mathcal{SK}_x$ corresponding to x.
- Sign: It takes public parameters $\mathcal{PP}$, 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: It receives public parameters $\mathcal{PP}$, a message $m \in \mathcal{M}$, a signature δ and a claim associated index y as input. It returns a boolean value 1 for accept or 0 for reject.

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

3.2 Security of Predicate Signature

Definition 3.1 (Signer Privacy)**.** A PS scheme is said to be perfectly private if for all $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}$, $x_1, x_2 \in \mathcal{X}$, $\mathcal{SK}_{x_1} \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x_1)$, $\mathcal{SK}_{x_2} \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x_2)$, $m \in \mathcal{M}$, and $y \in \mathcal{Y}$ with $x_1 \sim y$ and $x_2 \sim y$, the distributions of $\mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_{x_1}, y)$ and $\mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_{x_2}, y)$ are identical, where the random coins of the distributions are only the random coins involved in Sign algorithm.

Definition 3.2 (Adaptive-Predicate Unforgeability)**.** A PS scheme is said to be *adaptive-predicate existential unforgeable* (AP-UF-CMA) if for all PPT adversary $\mathscr{A}$, the advantage $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PS-UF}}(\kappa)$ is at most negligible function in κ, where $\mathscr{A}$ is provided the access to keyGen oracle, $\mathcal{O}_K$ and sign oracle, $\mathcal{O}_{Sg}$ and NRn is the natural restriction that (m^*, y^*) was never queried to $\mathcal{O}_{Sg}$ oracle and for each key index x queried to $\mathcal{O}_K$, $x \not\sim y^*$.

$$\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PS-UF}}(\kappa) := \Pr\left[\begin{array}{l}(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j}), \\ (\delta^*, m^*, y^*) \longleftarrow \mathscr{A}^{\{\mathcal{O}_K,\, \mathcal{O}_{Sg}\}}(\mathcal{PP})\end{array} : \mathsf{Ver}(\mathcal{PP}, m^*, \delta^*, y^*) = 1 \wedge \mathsf{NRn}\right].$$

Remark 3.1. There is an another variant of unforgeability, called *selective-predicate unforgeability*, where $\mathscr{A}$ submits a challenge index $y^* \in \mathcal{Y}$ (later on which it will forge) before obtaining the $\mathcal{PP}$ of ABS.

3.3 Pair Encoding Scheme ([2])

A Pair Encoding Scheme, P for a predicate family, $\sim$ consists of four deterministic algorithms, Param, $\mathsf{Enc1}$, $\mathsf{Enc2}$ and Pair.

– $\mathsf{Param}(\boldsymbol{j}) \longrightarrow n \in \mathbb{N}$. n describes the number of common variables involved in $\mathsf{Enc1}$ and $\mathsf{Enc2}$. Let $\boldsymbol{h} := (h_1, \ldots, h_n) \in \mathbb{Z}_N^n$ denotes the common variables in $\mathsf{Enc1}$ and $\mathsf{Enc2}$.

– $\mathsf{Enc1}(x \in \mathcal{X}, N) \longrightarrow (\boldsymbol{k}_x := (k_1, \ldots, k_{m_1}), m_2)$, where k_ι's 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 word, 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 $\Big(k_\iota(\alpha, \boldsymbol{r}, \boldsymbol{h}) := b_\iota \alpha + (\sum_{j\in[m_2]} b_{\iota,j} r_j) + (\sum_{\substack{j\in[m_2]\\ i\in[n]}} b_{\iota,j,i} h_i r_j)\Big)_{\iota\in[m_1]}$, where $\boldsymbol{r} := (r_1, \ldots, r_{m_2})$.

– $\mathsf{Enc2}(y \in \mathcal{Y}, N) \longrightarrow (\boldsymbol{c}_y := (c_1, \ldots, c_{\omega_1}), \omega_2)$, where c_ι's 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, $s, s_j, h_i s, h_i s_j$, where $s, s_1, \ldots, s_{\omega_2}, h_1, \ldots, h_n$ are variables. In other word, it outputs a set of coefficients $\{a_\iota, a_{\iota,j}, a'_{\iota,i}, a_{\iota,j,i}\}_{\iota\in[\omega_1],j\in[\omega_2],i\in[n]}$ which define the sequence of polynomials

$$\Big(c_\iota(\boldsymbol{s} = (s, s_1, \ldots, s_{\omega_2}), \boldsymbol{h}) := a_\iota s + (\sum_{j\in[\omega_2]} a_{\iota,j} s_j) + (\sum_{i\in[n]} a'_{\iota,i} h_i s) + (\sum_{\substack{j\in[\omega_2]\\ i\in[n]}} a_{\iota,j,i} h_i s_j)\Big)$$

– $\mathsf{Pair}(x, y, N) \longrightarrow \boldsymbol{E} \in \mathbb{Z}_N^{m_1\times\omega_1}$

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

Properties of Pair Encoding Scheme. We define two properties of pair encoding scheme as follows

- (Param-Vanishing): $\boldsymbol{k}(\alpha, \boldsymbol{0}, \boldsymbol{h}) = \boldsymbol{k}(\alpha, \boldsymbol{0}, \boldsymbol{0})$.
- (Linearity):

$$\boldsymbol{k}(\alpha_1, \boldsymbol{r}_1, \boldsymbol{h}) + \boldsymbol{k}(\alpha_2, \boldsymbol{r}_2, \boldsymbol{h}) = \boldsymbol{k}(\alpha_1 + \alpha_2, \boldsymbol{r}_1 + \boldsymbol{r}_2, \boldsymbol{h})$$
$$\boldsymbol{c}(\boldsymbol{s}_1, \boldsymbol{h}) + \boldsymbol{c}(\boldsymbol{s}_2, \boldsymbol{h}) = \boldsymbol{c}(\boldsymbol{s}_1 + \boldsymbol{s}_2, \boldsymbol{h})$$

Conditions 3.2. (Sufficient) Our objectives are to design the predicate signature, CCA-secure predicate encryption and predicate signcryption schemes from the pair encoding schemes. For assuring the correctness and security of the constructions, we impose some conditions defined below on the primitive pair encoding schemes. To best of our knowledge, most of the pair encoding schemes satisfy these conditions:

(1) $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = s$ for some $\iota \in [\omega_1]$ (w.l.g we assume $c_1(\boldsymbol{s}, \boldsymbol{h}) = s$)
(2) For $j \in [\omega_2]$, either (a) there is a $\iota \in [\omega_1]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j} s_j$ or (b) first the case-(a) is not happened, then if $a_{\iota,j,i'} \neq 0$ 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 $i \neq i'$, $a_{\iota,j,i} = 0$ and $h_{i'}$ is co-prime to N.
(3) For $(x, y) \in \mathcal{X} \times \mathcal{Y}$ with $x \sim y$, let $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x, N)$ and $\boldsymbol{E} \longleftarrow \mathsf{Pair}(x, y, N)$. Suppose there are $\iota_1, \ldots, \iota_\ell \in [m_1]$ such that $b_{\iota_i} \neq 0$ for $i \in [\ell]$. W.l.g, we assume that $\iota_i = i$, i.e., $b_i \neq 0$ for $i \in [\ell]$. Then, we require that $E_{ij} = 0$ for $i \in [\ell]$ and $j \in [2, \omega_1]$.

The 1st and 2nd conditions are put on $\mathsf{Enc2}$ and 3rd condition is imposed on $\mathsf{Enc1}$ and Pair. However, all these conditions are discuss in their respective contexts. A pair encoding which satisfies the 1st condition is referred as normal in [5]. In Appendix A, we worked out some pair encodings to understand the conditions.

3.4 Security of Pair Encoding Scheme

Below, we consider two forms of security, viz., perfect security and computational security as defined in [2].

– Perfect Security: A pair encoding scheme is said to be *perfectly master-key hiding* (PMH) if for $N \in \mathbb{N}$, $x \not\sim_N y$, $n \longleftarrow \mathsf{Param}(\boldsymbol{j})$, $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x, N)$ and $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$, the following two distributions are identical:

$$\{\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}), \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\} \text{ and } \{\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}), \boldsymbol{k}_x(0, \boldsymbol{r}, \boldsymbol{h})\}$$

where the random coins of the distributions are $\alpha \xleftarrow{\mathrm{U}} \mathbb{Z}_N, \boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n, \boldsymbol{s} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}, \boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$.

– 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 $G \in \{\text{SMH,CMH}\}$ if for $b \xleftarrow{\mathrm{U}} \{0,1\}$, all PPT adversary $\mathscr{A} := (\mathscr{A}_1, \mathscr{A}_2)$, the advantage $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{P-G}}(\kappa) := \left|\Pr[\mathsf{Exp}_{\mathscr{A},0}^{G}(\kappa) = 1] - \Pr[\mathsf{Exp}_{\mathscr{A},1}^{G}(\kappa) = 1]\right|$ in the experiment $\mathsf{Exp}_{\mathscr{A},b}^{G}(\kappa)$ defined below is at most a negligible function in security parameter κ:

$$\mathsf{Exp}_{\mathscr{A},b}^{G}(\kappa) := \begin{pmatrix} (N := p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \longleftarrow \mathcal{G}(1^\kappa) \\ (g, g_2, g_3) \xleftarrow{\mathrm{u}} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3} \\ \alpha \xleftarrow{\mathrm{U}} \mathbb{Z}_N,\ n \longleftarrow \mathsf{Param}(\boldsymbol{j}),\ \boldsymbol{h} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^n \\ st \longleftarrow \mathscr{A}_1^{\mathcal{O}^1_{G,b,\alpha,\boldsymbol{h}}(.)}(g, g_2, g_3) \\ b' \longleftarrow \mathscr{A}_2^{\mathcal{O}^2_{G,b,\alpha,\boldsymbol{h}}(.)}(st) \end{pmatrix}$$

where $\mathscr{A}$ is provided the access to two oracles, $\mathcal{O}^1_{G,b,\alpha,\boldsymbol{h}}(.)$ and $\mathcal{O}^2_{G,b,\alpha,\boldsymbol{h}}(.)$ defined below:

- For Selective Security: $\mathcal{O}^1$ is allowed only once, while $\mathcal{O}^2$ is allowed to query polynomially many times
  - $\mathcal{O}^1_{\mathsf{SMH},b,\alpha,\boldsymbol{h}}(y^*)$: Run $(\boldsymbol{c}_{y^*}, \omega_2) \longleftarrow \mathsf{Enc2}(y^*, p_2)$, pick $\boldsymbol{s} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$ and return $\boldsymbol{C}_{y^*} := g_2^{\boldsymbol{c}_{y^*}(\boldsymbol{s},\boldsymbol{h})}$.
  - $\mathcal{O}^2_{\mathsf{SMH},b,\alpha,\boldsymbol{h}}(x)$: If $x \not\sim_{p_2} y^*$, return $\perp$. Run $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x, p_2)$, pick $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and return
$$\boldsymbol{K}_x := \begin{cases} g_2^{\boldsymbol{k}_x(0,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 0 \\ g_2^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 1 \end{cases}$$
- For Co-selective Security: Both the oracles, $\mathcal{O}^1$ and $\mathcal{O}^2$ are allowed to query only once.
  - $\mathcal{O}^1_{\mathsf{CMH},b,\alpha,\boldsymbol{h}}(x^*)$: Run $(\boldsymbol{k}_{x^*}, m_2) \longleftarrow \mathsf{Enc1}(x^*, p_2)$, pick $\boldsymbol{r} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and then return
$$\boldsymbol{K}_{x^*} := \begin{cases} g_2^{\boldsymbol{k}_{x^*}(0,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 0 \\ g_2^{\boldsymbol{k}_{x^*}(\alpha,\boldsymbol{r},\boldsymbol{h})} & \text{if } b = 1 \end{cases}$$
  - $\mathcal{O}^2_{\mathsf{CMH},b,\alpha,\boldsymbol{h}}(y)$: If $x^* \not\sim_{p_2} y$, return $\perp$. Run $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, p_2)$, pick $\boldsymbol{s} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$ and then return $\boldsymbol{C}_y := g_2^{\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h})}$.

Remark 3.3. 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 as (t_1, t_2)-SMH (resp. (t_1, t_2)-CMH) security. What considered in [2], are actually the $(1, poly)$-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.

3.5 Dual conversion of pair encodings([2, 5])

For a predicate tuple $(\sim, \mathcal{X}, \mathcal{Y})$, its dual predicate tuple, $(\bar{\sim}, \bar{\mathcal{X}}, \bar{\mathcal{Y}})$ is defined by $\bar{\mathcal{X}} := \mathcal{Y}$, $\bar{\mathcal{Y}} := \mathcal{X}$ and for $x \in \bar{\mathcal{X}}$ and $y \in \bar{\mathcal{Y}}$, $x \bar{\sim} y := y \sim x$. We illustrate the dual conversion technique [2, 5] for converting a pair encoding for $\sim$ to a another pair encoding for the dual predicate, $\bar{\sim}$.

Let P be a given pair encoding scheme for the predicate $\sim$. We construct a pair encoding scheme $\mathbb{D}(\mathsf{P})$ for the predicate $\bar{\sim}$ 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.

- $\overline{\mathsf{Enc1}}(x, N)$: It runs $(\boldsymbol{c}'_x(\boldsymbol{s}', \boldsymbol{h}), \omega_2) \leftarrow \mathsf{Enc2}(x, N)$, where $\boldsymbol{s}' := (s', s'_1, \ldots, s'_{\omega_2})$. Then sets, $\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \bar{\boldsymbol{h}}) := (\boldsymbol{c}'_x(\boldsymbol{s}', \boldsymbol{h}), \alpha + \phi.s')$, $\boldsymbol{r} := \boldsymbol{s}'$ and outputs $(\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \bar{\boldsymbol{h}}), \omega_2)$, where α is 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, $\boldsymbol{c}_y(\boldsymbol{s}, \bar{\boldsymbol{h}}) := (\boldsymbol{k}'_y(\phi.s, \boldsymbol{s}, \bar{\boldsymbol{h}}), s)$, $\boldsymbol{s} := (s, \boldsymbol{r}')$ and returns $(\boldsymbol{c}_y(\boldsymbol{s}, \bar{\boldsymbol{h}}), m_2)$, where s is a new variable.

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

Proposition 3.1. *([2]) If a pair encoding* P *for* $\sim$ *is perfectly master-key hiding, then the pair encoding* $\mathbb{D}(\mathsf{P})$ *for* $\bar{\sim}$ *is also perfectly master-key hiding.*

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

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

Observation 3.4. We first note that the pair encoding scheme, $\mathbb{D}(\mathsf{P})$ satisfies the condition (1) in **Conditions 3.2** due to newly added variable s. Let's examine the 3rd condition. W.l.g, we set $c_{y,1} = s$ and $k_{x,1} = \alpha + \phi.s'$. The correctness of $\mathbb{D}(\mathsf{P})$ says that $\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}\boldsymbol{c}^{\top}_y(\boldsymbol{s}, \boldsymbol{h}) = k_{x,1}.c_{y,1} - \boldsymbol{k}'_y(\alpha', \boldsymbol{r}', \boldsymbol{h})\boldsymbol{E}'\boldsymbol{c}'^{\top}_x(\boldsymbol{s}', \boldsymbol{h}) = \alpha s$. 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 of 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 the condition (3) in **Conditions 3.2**. We note that the condition (2) in **Conditions 3.2** is imposed on the $\mathsf{Enc2}$, similarly it could be defined over $\mathsf{Enc1}$ and let's call it condition $(\bar{2})$. One can verify that if a pair encoding scheme P for predicate, $\sim$ fulfills the condition $(\bar{2})$, then its dual, $\mathbb{D}(\mathsf{P})$ for $\bar{\sim}$ satisfies the condition (2). We remark that all the pair encodings [2] and its dual so far satisfy the **Conditions 3.2**.

3.6 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 $\boldsymbol{h}_{\mathsf{M}} := (\theta_1, \theta_2, \boldsymbol{h})$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$ and $c_0(z, \boldsymbol{\theta}) := z(\theta_1\hbar + \theta_2)$, where z is the independent variable. Note that $\theta_1, \theta_2, \hbar$ and $\boldsymbol{h}$ will be understood from the context. For $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$, we define $\boldsymbol{c}_y^{\mathsf{M}} := (c_0, \boldsymbol{c}_y)$, so $|\boldsymbol{c}_y^{\mathsf{M}}| = \omega_1 + 1$ if $|\boldsymbol{c}_y| = \omega_1$. Therefore, we can write $\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) := (c_0(s, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h}))$ for $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}$. We define $\mathbf{V} := \{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) \in \mathbb{Z}_N^{\omega_1+1} \mid \boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}\}$. Now, we define a orthogonal set to be $\mathbf{V}^{\perp} := \{\boldsymbol{v}_{\mathsf{sp}} \in \mathbb{Z}_N^{\omega_1+1} \mid <\boldsymbol{v}_{\mathsf{sp}}, \boldsymbol{u}>= 0 \ \forall \ \boldsymbol{u} \in \mathbf{V}\}$. The process of sampling from $\mathbf{V}^{\perp}$ are given in sec.3.7.

Let $\mathsf{P} := (\mathsf{Param}, \mathsf{Enc1}, \mathsf{Enc2}, \mathsf{Pair})$ be a primitive pair encoding scheme with the following condition (already defined in **Conditions 3.2**)

Here we assume that for some $\iota \in [\omega_1]$, $c_{y,\iota}(\boldsymbol{s}, \boldsymbol{h}) = s$. W.l.g, we assume that $c_{y,1}(\boldsymbol{s}, \boldsymbol{h}) = s$

– $\mathsf{Setup}(1^\kappa, \boldsymbol{j})$: It executes $\mathcal{J} := (N := p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, e) \longleftarrow \mathcal{G}(1^\kappa)$. It chooses $g \xleftarrow{\text{U}} \mathbb{G}_{p_1}, Z_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}$. It runs $n \longleftarrow \mathsf{Param}(\boldsymbol{j})$ and picks $\boldsymbol{h} \xleftarrow{\text{U}} \mathbb{Z}_N^n$. Again it picks $\alpha, \theta_1, \theta_2 \xleftarrow{\text{U}} \mathbb{Z}_N$. It sets $\boldsymbol{h}_{\mathsf{M}} := (\theta_1, \theta_2, \boldsymbol{h}) \in \mathbb{Z}_N^{n+2}$. Let $H : \{0,1\}^* \longrightarrow \mathbb{Z}_N$ be a hash function. The public parameters and master secret are given by

$$\mathcal{PP} := [\mathcal{J}, g, g^{\boldsymbol{h}_{\mathsf{M}}}, g_T^{\alpha} := e(g,g)^{\alpha}, Z_3, H], \quad \mathcal{MSK} := [\alpha]$$

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

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

– $\mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_x, y)$: If $x \not\sim y$, returns $\perp$. Let $\mathcal{SK}_x = [x, \ \boldsymbol{K}_x]$. It runs[4] $\boldsymbol{K}_x := g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})}.\boldsymbol{R}_3 \longleftarrow \mathsf{Re\text{-}Randomize}(\boldsymbol{K}_x)$ and $\mathsf{Pair}(x, y) \longrightarrow \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1}$. Then, it computes $\hbar := H(m, y)$. It picks $\tau \xleftarrow{\text{U}} \mathbb{Z}_N$, $\boldsymbol{v}_{\mathsf{sp}} \xleftarrow{\text{U}} \mathbf{V}^{\perp}$ and $\boldsymbol{R}'_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{\omega_1+1}$. It sets $\boldsymbol{u} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})\boldsymbol{E}) \in \mathbb{Z}_N^{\omega_1+1}$, where $\boldsymbol{\psi} := (\tau(\theta_1\hbar + \theta_2), 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$. The signature is given by

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

We note that $\boldsymbol{\delta}_y$ can be easily computed from $\mathcal{SK}_x$, $g^{\boldsymbol{h}_{\mathsf{M}}}$, $\boldsymbol{E}$ and the random coins involved in the sign algorithm. In fact,

$$\boldsymbol{\delta}_y = (g^{-\tau}, \Theta, \ldots, \Theta).(\Theta, (g^{\theta_1})^{\tau\hbar}.(g^{\theta_2})^{\tau}, \Theta, \ldots, \Theta).(\Theta, \boldsymbol{K}_x^{\boldsymbol{E}}).g^{\boldsymbol{v}_{\mathsf{sp}}}.\boldsymbol{R}'_3$$

– $\mathsf{Ver}(\mathcal{PP}, m, \boldsymbol{\delta}_y, y)$: It runs $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$ and picks $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. It computes $\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) := (c_0(s, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{Z}_N^{\omega_1+1}$, where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, $\hbar := H(m, y)$ and $c_0(s, \boldsymbol{\theta}) := s(\theta_1\hbar + \theta_2)$. It computes a verification text, $\mathcal{V} := (\mathcal{V}_{\mathsf{INT}} := g_T^{\alpha s}, \boldsymbol{\mathcal{V}}_y := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}})})$. It returns 1 if $e(\boldsymbol{\delta}_y, \boldsymbol{\mathcal{V}}_y) = \mathcal{V}_{\mathsf{INT}}$ else 0.

[4]The linear property of pair encoding scheme guarantees the re-randomization.

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

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

Remark 3.5. In the Sign algorithm, two random coins, τ and $\boldsymbol{v}_{\mathsf{sp}}$ are used, among them $\boldsymbol{v}_{\mathsf{sp}}$ is assigned only for signer privacy and τ is the only coin that provides the randomness in unforgeability. If signer privacy is not our interest, then we can ignore $\boldsymbol{v}_{\mathsf{sp}}$.

Fact 3.6. We note that size of the signature for a message (m, y) is $\omega_1 + 1$, where $|\boldsymbol{c}_y| = \omega_1$ and 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 constant-size and the number of pairings in verification will be constant-size**.

3.7 How to uniformly sample from $\mathbf{V}^{\perp}$

Let $\mathbf{V}' := \{\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h}) \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}) \in \mathbb{Z}_N^{\omega_2+1}\}$ and $\mathbf{V}'^{\perp} := \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid\ < \boldsymbol{v}, \boldsymbol{u} >= 0\ \forall\ \boldsymbol{u} \in \mathbf{V}'\}$. We don't know how to sample uniformly from $\mathbf{V}'^{\perp}$ for arbitrary pair encoding schemes, however if we put some conditions on Enc2 of P, then we can sample. Of course, these restrictions narrow down our scope, but to best of our knowledge most of the existing pair encoding schemes satisfy the conditions. If we write $\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h}) = \boldsymbol{c}(\boldsymbol{s},\boldsymbol{h}) := (c_1(\boldsymbol{s},\boldsymbol{h}), c_2(\boldsymbol{s},\boldsymbol{h}), \ldots, c_{\omega_1}(\boldsymbol{s},\boldsymbol{h}))$, where $c_\iota(\boldsymbol{s},\boldsymbol{h}) := a_\iota s + \big(\sum_{j\in[\omega_2]} a_{\iota,j}s_j\big) + \big(\sum_{i\in[n]} a'_{\iota,i}h_i s\big) + \big(\sum_{\substack{j\in[\omega_2]\\ i\in[n]}} a_{\iota,j,i}h_i s_j\big)$, $\boldsymbol{s} = (s, s_1, \ldots, s_{\omega_2})$ and $\boldsymbol{h} = (h_1, \ldots, h_n)$, 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 $\boldsymbol{A} \in \mathbb{Z}_N^{\omega_1\times(\omega_2+1)}$. In fact, the ι^{th} row of $\boldsymbol{A}$ is given by $(a_\iota + \sum_{i\in[n]} a'_{\iota,i}h_i,\ a_{\iota,1} + \sum_{i\in[n]} a_{\iota,1,i}h_i,\ \ldots,\ a_{\iota,\omega_2} + \sum_{i\in[n]} a_{\iota,\omega_2,i}h_i)$. By the definition of $\mathbf{V}'^{\perp}$, we have following identities

$$\begin{aligned}
\mathbf{V}'^{\perp} &= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid\ < \boldsymbol{v}, \boldsymbol{u} >= 0\ \forall\ \boldsymbol{u} \in \mathbf{V}'\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{v}\boldsymbol{c}_y^{\top}(\boldsymbol{s},\boldsymbol{h}) = 0\ \forall\ \boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{v}\boldsymbol{A}\boldsymbol{s}^{\top} = 0\ \forall\ \boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{v}\boldsymbol{A} = \mathbf{0}\}\\
&= \{\boldsymbol{v} \in \mathbb{Z}_N^{\omega_1} \mid \boldsymbol{A}^{\top}\boldsymbol{v}^{\top} = \mathbf{0}\}
\end{aligned}$$

Now we are solely interested in solving the homogeneous system, $\boldsymbol{A}^{\top}\boldsymbol{X} = \mathbf{0}$, with $\boldsymbol{X}^{\top} := (x_1, x_2, \ldots, x_{\omega_1})$. **Before proceeding further we note that the sampling of $\mathbf{V}'^{\perp}$ gives rise the sampling of $\mathbf{V}^{\perp}$ if** $c_1(\boldsymbol{s},\boldsymbol{h}) = s$. It is assured using the Theorem A.3, where $\boldsymbol{A}_{\mathsf{M}}^{\top}$ is defined from $\boldsymbol{A}^{\top}$ and $t := \theta_1\hbar + \theta_2$.

Our goal is to compute $g^{\boldsymbol{v}}$, where $\boldsymbol{v} \xleftarrow{\text{U}} \mathbf{V}'^{\perp}$. Note that $g^{\boldsymbol{h}}$ is given but not $\boldsymbol{h}$. If each component v_j of $\boldsymbol{v}$ are linear combination of h_i's, then we will be able to compute $g^{\boldsymbol{v}}$. Since h_i's are not known, we are not able to compute h_i^{-1} required for the elementary operations (for details of the elementary operations, we refer to Appendix A). Even, it may happen that the h_i's are not invertible in $\mathbb{Z}_N$. So the only information of $\boldsymbol{A}$ available in the process of elementary operations are a_ι's, $a'_{\iota,i}$'s, $a_{\iota,j}$'s and $a_{\iota,j,i}$'s. Therefore, through out the elementary operations, we treat h_i's as symbols, where the symbols h_i^{-1}'s are not known. But, if we find some row of $\boldsymbol{A}^\top$ is a multiple of h_i, then we can multiple the row by h_i^{-1} (provided it exists in $\mathbb{Z}_N$) to make the row h_i free as solutions of the systems before and after the multiplication, remain unchanged. Suppose $\boldsymbol{M}$ is obtained by applying say n elementary column operations on $\boldsymbol{A}^\top$, then we have $\boldsymbol{A}^\top\boldsymbol{\mathcal{E}}_1^\top\boldsymbol{\mathcal{E}}_2^\top\cdots\boldsymbol{\mathcal{E}}_n^\top = \boldsymbol{M}$, where $\boldsymbol{\mathcal{E}}_i$ is an elementary matrix. If the column operations are other than the type-1, then there is a chance that h_i may appear in the elementary matrix $\boldsymbol{\mathcal{E}}_j^\top$. Since for each solution $\boldsymbol{v} := (v_1, \ldots, v_{\omega_1})^\top$ of $\boldsymbol{MX} = \boldsymbol{0}$, $\boldsymbol{\mathcal{E}}_1^\top\boldsymbol{\mathcal{E}}_2^\top\cdots\boldsymbol{\mathcal{E}}_n^\top\boldsymbol{v}$ is a solution of $\boldsymbol{A}^\top\boldsymbol{X} = \boldsymbol{0}$ and v_ι's are linear combination of h_i's, the terms like $h_{i_1}h_{i_2}\cdots h_{i_k}$ may appear in $\boldsymbol{v}$ to hamper our life.

Definition 3.3. A ι^{th} column of $\boldsymbol{A}^\top$ is said to be "leading h-free" column for j^{th} row of $\boldsymbol{A}^\top$ if for $i \in [\omega_2{+}1]$, $\boldsymbol{A}^\top_{i\iota} = a_{\iota,i}\delta_{i,j}$, where $\delta_{i,j}$ is Kronecker delta function (in fact, $\delta_{i,j} = 1$ if $i = j$, else 0).

Definition 3.4. A coin s_j is called "h-free" if a leading h-free column exists for the j^{th} row of $\boldsymbol{A}^\top$, otherwise it is called "non h-free" coin.

Conditions 3.7. (For Signer Privacy) Now all the technicalities keeping in mind, we define the sufficient conditions (already defined in **Conditions 3.2**) for sampling as follows:

(1) $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = s$ for some $\iota \in [\omega_1]$ (w.l.g we assume $c_1(\boldsymbol{s}, \boldsymbol{h}) = s$)
(2) For $j \in [\omega_2]$, either (a) [case - s_j **is h-free**]: there is a $\iota \in [\omega_1]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j}s_j$ or (b) [case - s_j **is non h-free**]: first the case-(a) is not happened, then if $a_{\iota,j,i'} \neq 0$ 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 $i \neq i'$, $a_{\iota,j,i} = 0$ and $h_{i'}$ is co-prime to N.

We set $s_0 := s$. We define $S_{\mathsf{hf}} := \{\iota \in [\omega_1] \mid \exists!\ j \in [0, \omega_2]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j}s_j\}$ and $T_{\mathsf{hf}} := \{j \in [0, \omega_2] \mid \exists\iota \in [\omega_1]$ such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j}s_j\}$. By assumption $c_1(\boldsymbol{s}, \boldsymbol{h}) = s = s_0$, we have $1 \in S_{\mathsf{hf}}$ and $0 \in T_{\mathsf{hf}}$ which imply S_{hf} and T_{hf} are non empty. Let $S_{\mathsf{non\text{-}hf}} := [\omega_1] \setminus S_{\mathsf{hf}}$. and $T_{\mathsf{non\text{-}hf}} := [0, \omega_2] \setminus T_{\mathsf{hf}}$. Now define $S_{\mathsf{phf}} := \{(\iota, j) \in [\omega_1] \times [0, \omega_2] \mid c_\iota(\boldsymbol{s}, \boldsymbol{h}) = a_{\iota,j}s_j\}$ and for each $j \in T_{\mathsf{hf}}$, $S_{\mathsf{non\text{-}fv},j} := \{\iota \in S_{\mathsf{hf}} \mid \iota = \min\{\iota \mid (\iota, j) \in S_{\mathsf{phf}}\}\}$.

Remark 3.8. Since, the factorization problem is intractable (as discussed in sec.A), all $a_{\iota,j}$'s appeared in condition (2) are invertible in $\mathbb{Z}_N$. For most of the pair coding schemes, we have $a_{\iota,j} = 1$ when $(\iota, j) \in S_{\mathsf{phf}}$ and so, $S_{\mathsf{non\text{-}fv},j}$ is singleton set for $j \in T_{\mathsf{hf}}$. When all coins are h-free, then $T_{\mathsf{non\text{-}hf}} = \emptyset$.

The main task is to find which variables are free and which are not among $x_1, x_2, \ldots, x_{\omega_1}$ with $\boldsymbol{X} := (x_1, \ldots, x_{\omega_1})^\top$. We will handle two cases separately for simplicity.

– **All s_j's are h-free.** Let $S_{\mathsf{non\text{-}fv}} := \cup_{j\in T_{\mathsf{hf}}} S_{\mathsf{non\text{-}fv},j}$ and $S_{\mathsf{fv}} := [\omega_1] \setminus S_{\mathsf{non\text{-}fv}}$. Note that S_{fv} and $S_{\mathsf{non\text{-}fv}}$ respectively represent the indices for free variable and non free variable. For $i \in S_{\mathsf{fv}}$, we assign $x_i := b_i \xleftarrow{\text{U}} \mathbb{Z}_N$ and for $\iota \in S_{\mathsf{non\text{-}fv}}$, we compute[5] $x_\iota := -a_{\iota,j}^{-1} \sum_{i\in S_{\mathsf{fv}}} a_{i,j}b_i$. The condition 2(a) guarantees that non non-free variable contributes during the computation of others and therefore $(x_1, x_2, \ldots, x_{\omega_1})^\top$ is a solution of the system $\boldsymbol{A}^\top\boldsymbol{X} = \boldsymbol{0}$. For this case, we do not require any elementary operation. In this case, $\mathsf{Null}(\boldsymbol{A}^\top) = \omega_1 - (\omega_2 + 1)$. For better understandable, we refer to Example A.2.

[5]for each $\iota \in S_{\mathsf{non\text{-}fv}}$, there is unique j such that $(\iota, j) \in S_{\mathsf{phf}}$

– **Not all s_j's are h-free.** For $j \in T_{\textsf{non-hf}}$, the j^{th} row of $\boldsymbol{A}^\top$ is multiplied[6] by $h_{j_i}^{-1}$ (symbolically) to have each element free from h-term. Under these changes the h-free coins remain h-free as the corresponding leading h-free columns are unaffected. Since, h_{j_i} is invertible (by condition 2(b)), the solutions of the system $\boldsymbol{A}^\top \boldsymbol{X} = \boldsymbol{0}$ remain unaltered. Now, we apply the elementary row operations of type-2 and type-3 as described below until each row $j \in T_{\textsf{non-hf}}$ becomes row reduced :

- Choose first non-zero (leading) element, say k of that row and then find the $gcd(k, N)$. If $gcd(k, N) > 1$, we solve the factorization problem in polynomial time in κ, else we apply the elementary row operation of type-2 to make the leading element to 1.
- Then, apply the elementary row operations of type-3 to reduce all other elements of the column containing leading 1 to 0.

Again under above elementary row operations, the h-free coins remain h-free as the corresponding leading h-free columns are unaffected but some non h-free coins become h-free. These new h-free coins make the free variables to non free variables. Let $S_{\textsf{new}} := \{\iota \in S_{\textsf{non-hf}} \mid \exists j \in T_{\textsf{non-hf}} \text{ such that } \boldsymbol{A}^\top_{i\iota} = \delta_{i,j}\}$ be the set of those new non free variables. Let $S_{\textsf{non-fv}} := \cup_{j \in T_{\textsf{hf}}} S_{\textsf{non-fv},j} \cup S_{\textsf{new}}$ and $S_{\textsf{fv}} := [\omega_1] \setminus S_{\textsf{non-fv}}$. The rest of the part are same as above. In this case, $\textsf{Null}(\boldsymbol{A}^\top) \leq \omega_1 - (\omega_2 + 1)$. More specifically, if $\boldsymbol{B} \in \mathbb{Z}_N^{|T_{\textsf{non-hf}}| \times \omega_1}$ be a sub-matrix of $\boldsymbol{A}^\top$ with $B_{ij} := A^\top_{ij}$ for $(i, j) \in [\omega_1] \times T_{\textsf{non-hf}}$, then $\textsf{Null}(\boldsymbol{A}^\top) = \omega_1 - (\omega_2 + 1) - \textsf{rank}(\boldsymbol{B})$. For better understandable, we refer to Example A.3.

3.8 Security Proof of Proposed Predicate Signature

3.8.1 Signer Privacy

Theorem 3.4. *Our proposed PS scheme in sec.3.6 is perfectly private.*

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

$$\boldsymbol{\delta}_y := g^{\boldsymbol{u}+\boldsymbol{v}_{\textsf{sp}}}.\boldsymbol{R}_3 \in \mathbb{G}^{\omega_1+1} \text{ and } \boldsymbol{u} \in \mathbf{V}_{\alpha \boldsymbol{s}} \text{ for } \boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}$$

so, it is sufficient to that $\boldsymbol{u} + \boldsymbol{v}_{\textsf{sp}}$ is uniformly distributed over $\mathbf{V}_{\alpha \boldsymbol{s}}$ for each $\boldsymbol{s} \in \mathbb{Z}_N^{\omega_2+1}$. Since, $\boldsymbol{v}_{\textsf{sp}}$ is chosen uniformly and independently from $\mathbf{V}^\perp$ and $\boldsymbol{u} + \mathbf{V}^\perp = \mathbf{V}_{\alpha \boldsymbol{s}}$, so we are done. □

3.8.2 The Proof of Unforgeability

To prove the unforgeability of the proposed construction in sec.3.6, we apply the signature variant [30, 31] of the dual system style of [2] (which abstracts the dual system methodology of [33]). In this variant, the original unforgeability game is changed to the final game through some intermediate games under three subgroup decision problems and CMH or PMH-security of the underlying pair encoding scheme. In the final game, $\mathcal{V}_{\textsf{INT}}$ of the verification text is sampled uniformly and independently from $\mathbb{G}_T$. Therefore, the forgery in the final game will be invalid. If ν_1 and ν_2 are respectively the number of key query and signature query made by $\mathscr{A}$, then the reduction cost is $\mathcal{O}(\nu_1 + \nu_2)$. For all the games, we define the semi-functional keys, signatures and verification texts of various type. We use the abbreviations 'vText' and 'sf-type' respectability for verification text and semi-functional type.

[6]for each $j \in T_{\textsf{non-hf}}$, there is a unique i' by condition 2(b) and let $j_i := i'$

– $\mathsf{SFSetup}(1^\kappa, \boldsymbol{j})$: It runs $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$ and in addition it returns semi-functional parameters, $g_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2}$, $\hat{\theta}_1, \hat{\theta}_2 \xleftarrow{\text{U}} \mathbb{Z}_N$ and $\hat{\boldsymbol{h}} \xleftarrow{\text{U}} \mathbb{Z}_N^n$. We set $\hat{\boldsymbol{h}}_{\mathsf{M}} := (\hat{\theta}_1, \hat{\theta}_2, \hat{\boldsymbol{h}})$.

– $\mathsf{SFKeyGen}(\mathcal{PP}, \mathcal{MSK}, x, g_2, \mathsf{type}, \hat{\boldsymbol{h}})$: It runs $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x, N)$ with $|\boldsymbol{k}_x| = m_1$. It chooses $\hat{\alpha} \xleftarrow{\text{U}} \mathbb{Z}_N$, $\boldsymbol{r}, \hat{\boldsymbol{r}} \xleftarrow{\text{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{m_1}$. It outputs the semi-functional key $\mathcal{SK}_x := (x, \boldsymbol{K}_x)$, where $\boldsymbol{K}_x$ is given by

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

– $\mathsf{SFSign}(\mathcal{PP}, m, \mathcal{SK}_x, y, g_2, \mathsf{type})$: If $x \not\sim y$, returns $\perp$. It runs $\boldsymbol{\delta}_y \longleftarrow \mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_x, y)$. Note that $\boldsymbol{\delta}_y = g^{\boldsymbol{u}+\boldsymbol{v}_{\mathsf{sp}}} . \boldsymbol{R}_3$ with $\boldsymbol{R}_3 \in \mathbb{G}_{p_3}^{\omega_1+1}$. It picks $b, \iota \xleftarrow{\text{U}} \mathbb{Z}_N$ and returns the semi-functional signature $\boldsymbol{\delta}_y . g_2^{\hat{\boldsymbol{u}}}$, where $\hat{\boldsymbol{u}} \in \mathbb{Z}_N^{\omega_1+1}$ is given by

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

– $\mathsf{SFVText}(\mathcal{PP}, m, y, g_2, \mathsf{type}, \hat{\boldsymbol{h}}_{\mathsf{M}})$: It runs $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$ and picks $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}), \hat{\boldsymbol{s}} := (\hat{s}, \hat{s}_1, \ldots, \hat{s}_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. It computes $\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) := (c_0(s, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{G}^{\omega_1+1}$ and $\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}}) := (c_0(\hat{s}, \hat{\boldsymbol{\theta}}), \boldsymbol{c}_y(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}})) \in \mathbb{G}^{\omega_1+1}$, where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, $\hat{\boldsymbol{\theta}} := (\hat{\theta}_1, \hat{\theta}_2, \hbar)$, $\hbar := H(m, y)$, $c_0(s, \boldsymbol{\theta}) := s(\theta_1 \hbar + \theta_2)$ and $c_0(\hat{s}, \hat{\boldsymbol{\theta}}) := \hat{s}(\hat{\theta}_1 \hbar + \hat{\theta}_2)$. It returns the semi-function verification text as

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

Condition 3.9. We assume a condition on the underlying pair encoding scheme, viz, $\boldsymbol{k}_x$ and $\boldsymbol{E}$ to go through the proof of unforgeability:

For $(x, y) \in \mathcal{X} \times \mathcal{Y}$ with $x \sim y$, $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x, N)$ and $\boldsymbol{E} \longleftarrow \mathsf{Pair}(x, y, N)$, we require that $\boldsymbol{k}_x(\alpha, \boldsymbol{0}, \boldsymbol{0})\boldsymbol{E} := (*, 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$, where $*$ is any entry from $\mathbb{Z}_N$.

Remark 3.10. The above condition is straightforwardly implied by the condition (3) in **Conditions 3.2**.

Remark 3.11. (Construction of sf-type 2 signature from sf-type 3 key.) We apply this construction (to Lemma D.8) for reaching to Game$_{Final}$ using the problem, DSG3. We see later that the sf-type 3 key is easily computed from the instance of the DSG3 problem. But, the computation of sf-type 2 signature is not possible unless we assume the **Condition 3.9**. Although one can directly compute the sf-type 2 signature, for simplicity of the simulation we show the construction of sf-type 2 signature from sf-type 3 key. If we apply the Sign algorithm to the sf-type 3 key, $\mathcal{SK}_x$, we have the following distribution (viz., the $\mathbb{G}_{p_2}$ part):

$$\begin{aligned} \boldsymbol{\delta}_y\big|_{\mathbb{G}_{p_2}} &= (g_2^0, g_2^{\boldsymbol{k}_x(\hat{\alpha}, \boldsymbol{0}, \boldsymbol{0})\boldsymbol{E}}) \\ &= (g_2^0, g_2^{(*, 0, \ldots, 0)}) && \text{(by condition 3.9)} \\ &= g_2^{(0, *, 0, \ldots, 0)} \end{aligned}$$

Then, randomize it by composing $g_2^{(0,\iota',0,\ldots,0)} \in \mathbb{G}_{p_2}^{\omega_1+1}$ for $\iota' \xleftarrow{\text{U}} \mathbb{Z}_N$ and finally what we get is the sf-type 2 signature.

Theorem 3.5. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies the* **Conditions 3.2** *and* $\sim$ *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 sec.3.6 for the predicate* $\sim$ *is adaptive-predicate existential unforgeable.*

Proof. Suppose there are at most ν_1 (resp. ν_2) key (resp. signature) queries made by an adversary $\mathscr{A}$, then the security proof consists of hybrid argument over a sequence of $3\nu_1 + 2\nu_2 + 4$ games. The games are defined below:

– Game$_{Real}$:= The original unforgeability game of the predicate signature scheme.

– Game$_{Res}$:= This is same as Game$_{Real}$ except $x \not\sim_N y^*$ is replaced by $x \not\sim_{p_2} y^*$ for each key query x made by $\mathscr{A}$.

– Game$_0$ (= Game$_{1-0-3}$) is just like Game$_{Res}$ except that the vText is of sf-type 1.

– In Game$_{1-k-1}$ (for $1 \leq k \leq \nu_1$) is same as Game$_{1-(k-1)-3}$ except the k^{th} key is sf-type 1.

– Game$_{1-k-2}$ (for $1 \leq k \leq \nu_1$) is same as Game$_{1-k-1}$ except the k^{th} key is sf-type 2.

– Game$_{1-k-3}$ (for $1 \leq k \leq \nu_1$) is same as Game$_{1-k-2}$ except the k^{th} key is sf-type 3.

– In Game$_{2-k-1}$ (for $1 \leq k \leq \nu_2$) is same as Game$_{2-(k-1)-2}$ except the k^{th} signature is of sf-type 1. (In this sequel, we define Game$_{2-0-2}$ = Game$_{1-\nu_1-3}$)

– Game$_{2-k-2}$ (for $1 \leq k \leq \nu_2$) is same as Game$_{2-k-1}$ except the k^{th} signature is of sf-type 2.

– Game$_{Final}$ is similar to Game$_{2-\nu_2-2}$ except that the vText is of sf-type 2.

In Game$_{Final}$, the part, $\mathcal{V}_{\mathsf{INT}}$ is chosen independently and uniformly random from $\mathbb{G}_T$ which implies that the forgery will be invalid with respect to the vText. Therefore, the adversary $\mathscr{A}$ has no advantage in Game$_{Final}$. The outline of the hybrid arguments over the games are structured in the box (for details of the lemmas, refer to Appendix D):

$$\begin{tikzcd}
\boxed{Real} \arrow[r, "\substack{\boxed{\text{Lem D.1}}\\ \boxed{\text{DSG2}}}"] & \boxed{Res} \arrow[r, "\substack{\boxed{\text{Lem D.2}}\\ \boxed{\text{DSG1}}}"] & \boxed{0} \arrow[r, "\substack{\boxed{\text{Lem D.3}}\\ \boxed{\text{DSG2}}}"] & \boxed{1-1-1} \ldots \boxed{1-(k-1)-3} \arrow[r, "\substack{\boxed{\text{Lem D.3}}\\ \boxed{\text{DSG2}}}"] & \boxed{1-k-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{1-k-1} \arrow[r, "\substack{\boxed{\text{Lem D.4}}\\ \boxed{\text{CHM}}}"] & \boxed{1-k-2} \arrow[r, "\substack{\boxed{\text{Lem D.5}}\\ \boxed{\text{DSG2}}}"] & \boxed{1-k-3} \ldots \boxed{1-\nu_1-3} \arrow[r, "\substack{\boxed{\text{Lem D.6}}\\ \boxed{\text{DSG2, CRH}}}"] & \boxed{2-1-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{2-1-1} \ldots \boxed{2-(k-1)-2} \arrow[r, "\substack{\boxed{\text{Lem D.6}}\\ \boxed{\text{DSG2, CRH}}}"] & \boxed{2-k-1} \arrow[r, "\substack{\boxed{\text{Lem D.7}}\\ \boxed{\text{DSG2}}}"] & \boxed{2-k-2} \ldots \boxed{2-\nu_2-2} \arrow[r, "\substack{\boxed{\text{Lem D.8}}\\ \boxed{\text{DSG3}}}"] & \boxed{Final}
\end{tikzcd}$$

Using the above structure, we have the following reduction:

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

where $\mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}_4}(\kappa)$ is the advantage of $\mathscr{B}_4$ in breaking 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 same as that of $\mathscr{A}$. This completes the theorem. □

Theorem 3.6. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies the* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has* PMH*-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 sec.3.6 for the predicate* $\sim$ *is adaptive-predicate existential unforgeable.*

Proof. Similar to the proof of the Theorem 3.5. The reduction of the proof is given by

$$\mathsf{Adv}^{\mathrm{PS-UF}}_{\mathscr{A}}(\kappa) \leq \mathsf{Adv}^{\mathrm{DSG1}}_{\mathscr{B}_1}(\kappa) + (2\nu_1 + 2\nu_2 + 1)\mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}_2}(\kappa) + \nu_2 \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}_3}(\kappa) + \mathsf{Adv}^{\mathrm{DSG3}}_{\mathscr{B}_4}(\kappa)$$

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

4 Instantiations of Predicate Signature

In this section, we instantiate (see Table 1) the different predicate signatures from various pair encoding schemes. The different variants of PS with many new features which have not been achieved earlier are presented here. If the underlying pair encoding scheme with either PMH or CMH-security satisfies the sufficient **Conditions 3.2**, then our construction for predicate signature in sec.3.6 guarantees the signer privacy and adaptive-predicate unforgeability of the predicate signature scheme. We consider in this section only the pair encoding schemes presented in [2] as they are having at least either PMH or CMH-security. Well, the other reasons to consider the pair encoding schemes mainly from [2] are that they are available in ready-made form and many new PS schemes with new features can be constructed from them. However, we are not strict to the pair encoding schemes of [2], it is applicable to all pair encoding scheme provided they are eligible. Now, the requirement remains to show that the pair encoding schemes of [2] satisfy the three conditions defined in sec.3.3. The first two conditions are imposed on $\boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h})$, whereas the third condition is on $\boldsymbol{k}(\alpha, \boldsymbol{r}, \boldsymbol{h})$ and $\boldsymbol{E}$. One can easily check that every pair encoding scheme of [2] fulfills the 1st and 3rd conditions.

Now come to the 2nd condition, which is required to ensure the signer-privacy: all the pair encoding schemes of [2] satisfy this condition. In fact, all the schemes except Scheme 10, Scheme 11 and Scheme 13 satisfy condition 2(a) not 2(b), i.e., all the coins involved in each scheme are h-free and whereas some of the coins in Scheme 10, Scheme 11 and Scheme 13 are non h-free due to the h-term, ϕ, i.e., $h_{j_i} = \phi$. This shows that all the pair encoding schemes in [2] fulfills the 3rd condition.

The authors in [5] used the pair encodings of [2] to construct dual of the previous proposed predicate encryption of [2]. Following the Observation 3.4, we have that the $\mathbb{D}(\mathsf{P})$ of any pair encoding P satisfy the 1st and 3rd condition. Since, the pair encodings of [2] satisfy the condition $(\bar{2})$ (as mentioned in Observation 3.4), its dual will fulfill the condition (2). Therefore, all the pair encoding schemes in [2, 5] are eligible for conversion to predicate signature schemes.

IBS We start with an Adaptive-ID unforgeable identity-based signature scheme constructed from a simple pair encoding scheme, Scheme 1 of [2]. Both, the key space and associated data space are $\mathbb{Z}_N$. The

Table 1: In this table, we summarize different instantiations of the predicate signature constructed from the pair encodings of [2, 5]. The abbreviations NA, KP, SP and PES stand for not-applicable, key-policy, signature-policy and pair encoding scheme respectively. All the pair encodings shown in the table are either perfectly (PMH) secure or computationally (both, SMH and CMH) secure. However, the security given in table are used for unforgeability of the predicate signatures.

Predicate Signature	**Flavor**	**Feature**	**Pair Encoding** (P)	**Security of** P (**Used**)
IBS	*NA*	*Cost Free*	PES 1 [2]	PMH
PS	*KP*	*Regular Languages*	PES 3 [2]	CMH
PS	*SP*	*Regular Languages*	PES 7 [2]	CMH
ABS	*KP*	*Unbounded, Large Universes*	PES 4 [2]	CMH
ABS	*SP*	*Unbounded, Large Universes*	Dual[5] of PES 4 [2]	CMH
ABS	*KP*	*Constant-size signatures*	PES 5 [2]	CMH
ABS	*SP*	*Constant-size keys*	Dual[5] of PES 5 [2]	CMH
KP-DSS	*KP*	*It generalizes KP-ABS*	PES 6 [2]	CMH
SP-DSS	*SP*	*It generalizes SP-ABS*	Dual[5] of PES 6 [2]	CMH
ABS	*KP*	*Cost Free*	PES 8 [2]	PMH
ABS	*SP*	*Cost Free*	PES 10 [2]	PMH
ABS	*KP*	*Cost Free, Large Universes*	PES 12 [2]	PMH
ABS	*SP*	*Cost Free, Large Universes*	PES 13 [2]	PMH

Scheme 1 of [2] is perfectly master-key hiding due to the fact that the function $f(X) := h_1 + h_2X$ is pairwise independent function (used in [23]) and hence, the IBS scheme does not require any extra hardness assumption.

PS for Regular Languages. We achieve the predicate signature schemes for regular languages which are beyond the scope of ABS. To best of our knowledge, these are the only predicate signature schemes that increase the functionality from ABS to non-trivial PS. Our construction in sec.3.6 provides two flavors of PS for regular languages, key-policy PS and signature-policy PS respectively from the pair encodings, Scheme 3 and Scheme 7 of [2]. Both the predicate signature schemes are unforgeable in the adaptive-predicate model and achieve the perfect privacy. Both the signatures support the large universe alphabet.

Unbounded ABS with Large Universes. We obtain an adaptive-predicate unforgeable unbounded KP-ABS with large universes from the pair encoding scheme, Scheme 4 of [2] in key-policy flavor. Here unbounded means there is no restriction on the size of policies, attribute sets and the repetition of attribute in a policy. An ABS with large universes will have super-polynomial size attribute universe. Here we consider the attribute universe to be $\mathbb{Z}_N$. The policies are are represented by $\Gamma := (\boldsymbol{M}, \rho)$, where $\boldsymbol{M} \in \mathbb{Z}_N^{\ell \times k}$, called LSSS matrix and $\rho : [\ell] \to \mathbb{Z}_N$ is a row labeling function. The attribute set S is any subset of $\mathbb{Z}_N$. The size of the public parameters is constant. The only known adaptive-predicate unforgeable ABS with large universes available in the literature are the construction of [30, 27], among them only ABS of [27] has the feature, *unbounded*. However, all these construction are known to have the signature-policy. Therefore, the proposed ABS scheme is the first unbounded KP-ABS with large universes which is unforgeable in adaptive-predicate model. By applying our conversion in sec.3.6 on the dual [5] of pair encoding, Scheme

4 of [2], we also obtain an unbounded ABS with large universes in signature-policy flavor.

ABS with Constant-size Signatures (or Keys). Considering the pair encoding scheme, Scheme 5 of [2] as an input to our signature framework in sec.3.6, we achieve an adaptive-predicate unforgeable ABS with constant-size signatures in KP flavor. We keep a bound on maximum size of the attribute sets S and let max be the bound for the attribute sets. The policies are are represented by $\Gamma := (\boldsymbol{M}, \rho)$, where $\boldsymbol{M} \in \mathbb{Z}_N^{\ell \times k}$, called LSSS matrix and $\rho : [\ell] \to \mathbb{Z}_N$ is a row labeling function. The attribute set S is any subset of $\mathbb{Z}_N$ but, $|S| \leq \mathsf{max}$. The unforgeability of the only known constant-size signature [4] for non-monotone access structures was proven in the selective-predicate model. Therefore, the proposed ABS scheme is the first ABS with constant-size signature which is existential unforgeable in the adaptive-predicate model. Similarly, by applying our construction in sec.3.6 on the dual [5] of pair encoding, Scheme 5 of [2], we also obtain a ABS with constant-size keys in signature-policy flavor.

Key-Policy over Doubly-Spatial Signature. Attrapadung [2] proposed a new key-policy predicate encryption which generalizes the KP-ABE. This new KP-PE is called key-policy over doubly-spatial encryption (KP-DSE) which works in similar manner as the KP-ABE except the equality relation is replaced by doubly-spatial relation in doubly-spatial encryption [21]. The authors instantiated the KP-DSE using the pair encoding scheme, Scheme 6. Later, Attrapadung et al. [5] achieved the dual of KP-DSE, CP-DSE by applying the dual conversion on Scheme 6.

Similar to KP-DSE [2] (resp. CP-DSE [5]), its signature analogue key-policy over doubly-spatial signature (KP-DSS) (resp. signature-policy over doubly-spatial signature (SP-DSS)) generalizes the KP-ABS (resp. SP-ABS). By applying our construction in sec.3.6 on Scheme 6 and its dual, we respectively obtain KP-DSS and SP-DSS schemes.

Cost free ABS with Small Universes. The ABE schemes of [24] are the first ABE schemes which were shown to be fully (adaptively) secure in the standard model under the subgroup decision problems, DSG1, DSG2 and DSG3. In the proof strategy, the authors employed the dual system technique of Waters [33], where an information-theoretic argument was used. Later, Attrapadung [2] show that there are pair encoding schemes, viz., Scheme 8 of [2] and Scheme 10 of [2] setting inside KP-ABE of [24] and CP-ABE of [24] respectively and which are basically perfectly master-key hiding. So, the aforementioned information-theoretic argument actually was supplied by the respective pair encoding scheme.

Analogously, we obtain the cost free adaptive-predicate unforgeable ABS in both the flavors, KP and SP by applying our construction in sec.3.6 on the Scheme 8 and Scheme 10 of [2] respectively. The SP-ABS of [31] can be viewed by applying our construction on the Scheme 10 of [2].

Cost Free ABS with Large Universes. Attrapadung [2] constructed new ABE with large universes in KP and SP flavors from the perfectly master-key hiding pair encodings, Scheme 12 and Scheme 13 respectively. The pair encoding schemes were constructed based on cover-free families [16, 22]. Analogously, by applying our construction on Scheme 12 and Scheme 13 of [2], we obtain the adaptive-predicates unforgeable ABS schemes with large universes in both flavors, KP and SP. Since, the underlying pair encodings are perfectly master-key hiding, therefore, both the ABS are cost free.

5 Framework for CCA Secure Predicate Encryption

The traditional technique [35, 36, 29] for CCA conversion requires that the primitive CPA-secure PE schemes (for syntax and security definition of PE, we refer to Appendix B) must have either verifiability or delegation property. One good side towards this direction is that if the underlying pair encoding scheme fulfills the condition (1) in **Conditions 3.2**, then the fully secure construction in sec. 4.3 of [2] always satisfies the verifiability as follows:

Verifiability. In the following, we define the algorithm, verify where y is the data index implicitly contained in C_{cpa}, and x and x' are key indices. Let $\boldsymbol{E} := \mathsf{Pair}(x, y)$ and $\boldsymbol{E}' := \mathsf{Pair}(x', y)$.

$$\mathsf{Verify}(\mathcal{PP}, C_{\mathsf{cpa}}, x, x') := \begin{cases} \perp & \text{if } y \not\sim x \text{ or } y \not\sim \tilde{x} \\ 1 & \text{if } \mathsf{Event} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathsf{Event} := \begin{cases} e(g^{\boldsymbol{k}_x(0,\boldsymbol{1}_i,\boldsymbol{h})\boldsymbol{E}},\ \boldsymbol{C}_y) = 1\ \forall\ i \in [m_2] & (1) \\ e(g^{\boldsymbol{k}_{x'}(0,\boldsymbol{1}_i,\boldsymbol{h})\boldsymbol{E}'},\ \boldsymbol{C}_y) = 1\ \forall\ i \in [m_2{}'] & (2) \\ e(g^{\boldsymbol{k}_x(1,\boldsymbol{0},\boldsymbol{h})\boldsymbol{E}},\ \boldsymbol{C}_y) = e(g^{\boldsymbol{k}_{x'}(1,\boldsymbol{0},\boldsymbol{h})\boldsymbol{E}'},\ \boldsymbol{C}_y) = e(g, C_{y,1}) & (3) \\ \text{For } R_3 \in \mathbb{G}_{p_3},\quad e(R_3, C_{y,\iota}) = 1\ \forall\ \iota \in [\omega_1] & (4) \end{cases}$$

where $\boldsymbol{1}_i$ is a vector whose i^{th} position is 1 and rest are 0.

Soundness of Verifiability. Suppose $\mathsf{Verify}(\mathcal{PP}, C_{\mathsf{cpa}}, x, x') = 1$, then we show that both the keys, $\mathcal{SK}_x$ and $\mathcal{SK}_{x'}$ output the same message on decryption:

$$\begin{aligned} \mathsf{Decrypt}(\mathcal{PP}, C_{\mathsf{cpa}}, \mathcal{SK}_x) &= C_{\mathsf{INT}}/e(\boldsymbol{K}_x^{\boldsymbol{E}}, \boldsymbol{C}_y) \\ &= C_{\mathsf{INT}}/e(g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}}, \boldsymbol{C}_y) && \text{(by (4))} \\ &= C_{\mathsf{INT}}/e(g^{(\boldsymbol{k}_x(0,\boldsymbol{r},\boldsymbol{h})+\boldsymbol{k}_x(\alpha,\boldsymbol{0},\boldsymbol{h}))\boldsymbol{E}}, \boldsymbol{C}_y) && \text{(by Linearity of P)} \\ &= C_{\mathsf{INT}}/e(g^{(\sum_{i\in[m_2]} r_i \boldsymbol{k}_x(0,\boldsymbol{1}_i,\boldsymbol{h})+\alpha\boldsymbol{k}_x(1,\boldsymbol{0},\boldsymbol{h}))\boldsymbol{E}}, \boldsymbol{C}_y) && \text{(by Linearity of P)} \\ &= C_{\mathsf{INT}}/\Big(\prod_{i\in[m_2]} e(g^{\boldsymbol{k}_x(0,\boldsymbol{1}_i,\boldsymbol{h})\boldsymbol{E}}, \boldsymbol{C}_y)^{r_i}.e(g^{\boldsymbol{k}_x(1,\boldsymbol{0},\boldsymbol{h})\boldsymbol{E}}, \boldsymbol{C}_y)^{\alpha}\Big) \\ &= C_{\mathsf{INT}}/\Big(\prod_{i\in[m_2]} 1^{r_i}.e(g, C_{y,1})^{\alpha}\Big) && \text{(by (1) and (3))} \\ &= C_{\mathsf{INT}}/e(g, C_{y,1})^{\alpha} \end{aligned}$$

Since x is arbitrary, similarly we have $\mathsf{Decrypt}(\mathcal{PP}, C_{\mathsf{cpa}}, \mathcal{SK}_{x'}) = C_{\mathsf{INT}}/e(g, C_{y,1})^{\alpha}$.

Completeness of Verifiability. It follows from the correctness of the pair encoding scheme, P, orthogonality of CBG and the assumption, $c_{y,1} = s$.

Theorem 5.1. *Suppose the underlying pair encoding scheme of the fully secure construction for predicate in sec.4.3 [2] satisfies the condition (1) described in* **Conditions 3.2** *and a concrete index-transformer [29] for the predicate is available, then the CCA construction using verifiability-friendly index transformer [36, 29] is adaptive-predicate IND-CCA secure predicate encryption scheme.*

Proof. The adaptive CPA-secure construction in sec.4.3 of [2] has the verifiability (described above). Rest are followed from the proof of security based on verifiability in sec.3.5 of [36, 29]. □

Remark 5.1. All the fully secure CPA constructions in [2, 5] are eligible for adaptive-predicate CCA secure predicate encryption schemes as the concrete verifiability-friendly index transformers for the said predicates are available in [29, 36, 35] and all the underlying pair encoding schemes satisfy condition (1) of **Conditions 3.2**.

Our proposed (direct) CCA construction is motivated by the following two aspects :

- Due to the use of OTS scheme, the vk is to be embedded in data index y. This may lead to increase the length of key index and data index, and the size of universe. If the CCA-construction is based on verifiability, then the checking in verify degrades the performance of the decryption algorithm, a lot.
- As stated in [29], for some flavors of predicates either the concrete constructions are not known yet or the defined constructions do not rule out the existing PE schemes.

5.1 CCA-secure Predicate Encryption from Pair Encoding Scheme

We explore the direct CCA secure predicate encryption scheme from the pair encoding scheme. Using this construction, we achieve CCA security of all the predicate encryptions found in [2, 5] directly from the pair encodings of [2, 5] at almost the same cost of CPA construction of [2]. In fact, the difference between the construction of ours and [2] is that we use an extra component $C_0 := g^{c_0}$ in ciphertext and one extra paring computation in decryption.

Terminology: For $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$, we define $\boldsymbol{c}_y^{\mathsf{M}} := (c_0, \boldsymbol{c}_y)$, so $|\boldsymbol{c}_y^{\mathsf{M}}| = \omega_1 + 1$, where $c_0(z, \boldsymbol{\theta}) := z(\theta_1 \hbar + \theta_2)$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar) \in \mathbb{Z}_N^3$, z is the independent variable and $|\boldsymbol{c}_y| = \omega_1$.

Let $\mathsf{P} := (\mathsf{Param}, \mathsf{Enc1}, \mathsf{Enc2}, \mathsf{Pair})$ be a primitive pair encoding scheme with the following condition (already defined in **Conditions 3.2**)

$$\boxed{\text{Here we assume that for some } \iota \in [\omega_1],\ c_{y,\iota}(\boldsymbol{s}, \boldsymbol{h}) = s.\ \text{W.l.g, we assume that } c_{y,1}(\boldsymbol{s}, \boldsymbol{h}) = s}$$

– $\mathsf{Setup}(1^\kappa, \boldsymbol{j})$: Same as the Setup in sec.3.6.

– $\mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$: Same as the KeyGen in sec.3.6.

– $\mathsf{Encrypt}(\mathcal{PP}, m, y)$: It runs $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$ and picks $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. It computes $C_{\mathsf{cpa}} := (y, \boldsymbol{C}_y := g^{\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h})}, C_{\mathsf{INT}} := m.g_T^{\alpha s})$ and $\hbar := H(C_{\mathsf{cpa}})$. It sets $\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) := (c_0(s, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{Z}_N^{\omega_1+1}$, where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, and $c_0(s, \boldsymbol{\theta}) := s(\theta_1 \hbar + \theta_2)$. It returns $\mathsf{CT} := (y, \boldsymbol{C}_y^{\mathsf{M}} := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}})}, C_{\mathsf{INT}})$.

– $\mathsf{Decrypt}(\mathcal{PP}, \mathsf{CT}, \mathcal{SK}_x)$: It phrases CT as $(y, \boldsymbol{C}_y^{\mathsf{M}} = (C_0, \boldsymbol{C}_y), C_{\mathsf{INT}})$ with $\boldsymbol{C}_y = (C_1, \ldots, C_{\omega_1})$, computes $\hbar := H(C)$ and picks $R \xleftarrow{\text{U}} \mathbb{G}_{p_3}$. If $x \not\sim y$ or $e(gR, C_0) \neq e(g^{\theta_1 \hbar + \theta_2}, C_1)$, it returns $\perp$. It sets $\mathcal{SK}_x^{\mathsf{M}} := (K_0, \boldsymbol{\Psi}.\boldsymbol{K}_x^{\boldsymbol{E}}) \in \mathbb{G}^{\omega_1+1}$, where $K_0 := g^{-\tau} R_0$, $\boldsymbol{\Psi} := g^{\boldsymbol{\psi}}$ with $\boldsymbol{\psi} := (\tau(\theta_1 \hbar + \theta_2), 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$, $\tau \xleftarrow{\text{U}} \mathbb{Z}_N$, $R_0 \xleftarrow{\text{U}} \mathbb{G}_{p_3}$ and $\boldsymbol{E} \leftarrow \mathsf{Pair}(x, y)$. It returns $C_{\mathsf{INT}} / e(\mathcal{SK}_x^{\mathsf{M}}, \boldsymbol{C}_y^{\mathsf{M}})$.

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

$$\begin{aligned}
e(\mathcal{SK}_x^{\mathsf{M}}, \boldsymbol{C}_y^{\mathsf{M}}) &= g_T^{<(-\tau,\ \boldsymbol{\psi}+\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}),\ \boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s},\boldsymbol{h}_{\mathsf{M}})>} && \text{(by orthogonality of CBG)}\\
&= g_T^{<(-\tau,\ 0,\ldots,0)+(0,\ \boldsymbol{\psi})+(0,\ \boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}),\ \boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s},\boldsymbol{h}_{\mathsf{M}})>} && \text{(by Linearity)}\\
&= g_T^{-\tau c_0(s,\boldsymbol{\theta})+\tau(\theta_1\hbar+\theta_2)c_{y,1}(\boldsymbol{s},\boldsymbol{h})+<\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E},\ \boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h}_{\mathsf{M}})>}\\
&= g_T^{-\tau s(\theta_1\hbar+\theta_2)+\tau s(\theta_1\hbar+\theta_2)+\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})\boldsymbol{E}\boldsymbol{c}_y^{\top}(\boldsymbol{s},\boldsymbol{h}_{\mathsf{M}})} && \text{(by assumption: } c_{y,1}(\boldsymbol{s},\boldsymbol{h}) = s)\\
&= g_T^{\alpha s} && \text{(by correctness of P)}
\end{aligned}$$

Remark 5.2. Note that the CCA secure ciphertext is also represented as $\mathsf{CT} := (C_{\mathsf{cpa}}, C_0)$, where the routine $\mathsf{Encrypt}$ can be thought as subroutine $\mathsf{Encrypt}^*$ (the $\mathsf{Encrypt}$ of CPA construction in [2]), then followed by the computation of C_0.

Remark 5.3. The key $\mathcal{SK}_x^{\mathsf{M}}$ defined in $\mathsf{Decrypt}$, we call the *alternative key* (in short alt-key) whose distribution is exactly the same as the signature $\boldsymbol{\delta}_y$ if we ignore the random coin $\boldsymbol{v}_{\mathsf{sp}}$ for signer privacy. Using this alternative key if we run $\mathsf{AltDecrypt}$ (defined later), we have the same message as in $\mathsf{Decrypt}$ using the original key, $\mathcal{SK}_x$.

Fact 5.4. We note that size of the ciphertext is $\omega_1 + 2$, where $|\boldsymbol{c}_y| = \omega_1$ and number pairings in $\mathsf{Decrypt}$ is $\omega_1 + 1$. **Therefore, if $\boldsymbol{c}_y$ of the underlying pair encoding scheme is of constant-size, then the corresponding ciphertext will be constant-size and the number of pairings in decryption will be constant-size**.

5.2 Security Proof of Proposed Predicate Encryption

The proof strategy is based on the dual system style of [33, 2]. Again to pass the argument in Lemma E.11, we assume the **Condition 3.9** (which is implied by condition (3) of **Conditions 3.2**).

– $\mathsf{SFSetup}(1^\kappa, \boldsymbol{j})$: It runs $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$ and in addition it returns 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 $\hat{\boldsymbol{h}}_{\mathsf{M}} := (\hat{\theta}_1, \hat{\theta}_2, \hat{\boldsymbol{h}})$.

– $\mathsf{SFKeyGen}(\mathcal{PP}, \mathcal{MSK}, x, g_2, \mathsf{type}, \hat{\boldsymbol{h}})$: It runs $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x, N)$ with $|\boldsymbol{k}_x| = m_1$. It chooses $\hat{\alpha} \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $\boldsymbol{r}, \hat{\boldsymbol{r}} \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}^{m_1}$. It outputs the semi-functional key $\mathcal{SK}_x := (x, \boldsymbol{K}_x)$, where $\boldsymbol{K}_x$ is given by

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

– $\mathsf{SFEncrypt}(\mathcal{PP}, m, y, g_2, \mathsf{type}, \hat{\boldsymbol{h}}_{\mathsf{M}})$: It runs $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$ and picks $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}), \hat{\boldsymbol{s}} := (\hat{s}, \hat{s}_1, \ldots, \hat{s}_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$. It computes $\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) := (c_0(s, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{G}^{\omega_1+1}$ and $\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}}) := (c_0(\hat{s}, \hat{\boldsymbol{\theta}}), \boldsymbol{c}_y(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}})) \in \mathbb{G}^{\omega_1+1}$, where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, $\hat{\boldsymbol{\theta}} := (\hat{\theta}_1, \hat{\theta}_2, \hbar)$, $\hbar := H(C_{\mathsf{cpa}})$, $C_{\mathsf{cpa}} := (y, \boldsymbol{C}_y := g^{\boldsymbol{c}_y(\boldsymbol{s},\boldsymbol{h})}, C_{\mathsf{INT}} := m.g_T^{\alpha s})$, $c_0(s, \boldsymbol{\theta}) := s(\theta_1\hbar + \theta_2)$ and $c_0(\hat{s}, \hat{\boldsymbol{\theta}}) := \hat{s}(\hat{\theta}_1\hbar + \hat{\theta}_2)$. It returns the semi-function ciphertext as $\mathsf{CT} := (y, \boldsymbol{C}_y^{\mathsf{M}} := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s},\boldsymbol{h}_{\mathsf{M}})}, C_{\mathsf{INT}})$.

$$\mathsf{CT} := \begin{cases} (y, \boldsymbol{C}_y^{\mathsf{M}} := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s},\boldsymbol{h}_{\mathsf{M}})}.g_2^{\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}},\hat{\boldsymbol{h}}_{\mathsf{M}})}, C_{\mathsf{INT}} := m.g_T^{\alpha s} & \text{if type}= 1\\ (y, \boldsymbol{C}_y^{\mathsf{M}} := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s},\boldsymbol{h}_{\mathsf{M}})}.g_2^{\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}},\hat{\boldsymbol{h}}_{\mathsf{M}})}, C_{\mathsf{INT}} := m.g_t;\ g_t \xleftarrow{\mathrm{U}} \mathbb{G}_T & \text{if type}= 2 \end{cases}$$

– $\mathsf{SFAltKeyGen}(\mathcal{PP}, \mathcal{MSK}, \mathsf{CT}, x, g_2, \mathsf{type})$: It phrases CT as (C_{cpa}, C_0), computes $\hbar := H(C_{\mathsf{cpa}})$ and picks $\tau \xleftarrow{\text{U}} \mathbb{Z}_N$, $R_0 \xleftarrow{\text{U}} \mathbb{G}_{p_3}$. It first generates the normal key, $\mathcal{SK}_x := [x,\ \boldsymbol{K}_x := g^{\boldsymbol{k}_x(\alpha,\boldsymbol{r},\boldsymbol{h})}.\boldsymbol{R}_3]$. Then, it creates the alt-key $\mathcal{SK}_x^{\mathsf{M}} := (K_0, \boldsymbol{\Psi}.\boldsymbol{K}_x^{\boldsymbol{E}}) \in \mathbb{G}^{\omega_1+1}$, where $K_0 := g^{-\tau}R_0$, $\boldsymbol{\Psi} := g^{\boldsymbol{\psi}}$ with $\boldsymbol{\psi} := (\tau(\theta_1\hbar + \theta_2), 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$ and $\boldsymbol{E} \leftarrow \mathsf{Pair}(x, y)$. It picks $b, \iota \xleftarrow{\text{U}} \mathbb{Z}_N$ and returns the semi-functional alt-key $\mathcal{SK}_x^{\mathsf{M}}.g_2^{\hat{\boldsymbol{u}}}$, where $\hat{\boldsymbol{u}} \in \mathbb{Z}_N^{\omega_1+1}$ is given by

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

– $\mathsf{AltDecrypt}(\mathcal{PP}, \mathsf{CT}, \mathcal{SK}_x^{\mathsf{M}})$: This is same as $\mathsf{Decrypt}$ algorithm, but here we do not need to compute the alt-key as it is supplied. For sake of completeness: It picks $R \xleftarrow{\text{U}} \mathbb{G}_{p_3}$. If $x \not\sim y$ or $e(gR, C_0) \neq e(g^{\theta_1\hbar+\theta_2}, C_1)$, it returns $\perp$ else $C_{\mathsf{INT}}/e(\mathcal{SK}_x^{\mathsf{M}}, \boldsymbol{C}_y^{\mathsf{M}})$.

Remark 5.5. If we identify the challenge ciphertext and alt-keys respectively with the verification text and queried signatures in the unforgeability proof of the predicate signature scheme in sec.3.6, then most of the part of CCA-security proof of the proposed predicate encryption scheme in sec.5.1 will follow the unforgeability proof in sec.3.8.2.

Theorem 5.2. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies the conditions (1) and (3) of* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has both the security,* SMH *and* CMH*, the assumptions, DSG1, DSG2 and DSG3 hold in* $\mathcal{J}$ *and H is a collision resistant hash function, then the proposed predicate encryption scheme,* PE *in sec.5.1 for the predicate* $\sim$ *is adaptive-predicate IND-CCA secure.*

Proof. Suppose there are at most q (resp. ν) key (resp. decryption) queries made by an adversary $\mathscr{A}$, then the security proof consists of hybrid argument over a sequence of $3q_1 + 2\nu + 7$ games, where among the q key queries, q_1 and q_2 respectively be the number of phase 1 and phase 2 key queries. The games are defined below:

– Game_{Real} := The original adaptive predicate CCA-security game.

– Game_{Res} := This is same as Game_{Real} except $x \not\sim_N y^*$ is replaced by $x \not\sim_{p_2} y^*$ for each key query x made by $\mathscr{A}$.

– Game_0 (= Game_{1-0-3}) is just like Game_{Res} except that the challenge ciphertext is of sf-type 1.

– In Game_{1-k-1} (for $1 \leq k \leq q_1$) same as $\text{Game}_{1-(k-1)-3}$ except the k^{th} queried key is sf-type 1.

– Game_{1-k-2} (for $1 \leq k \leq q_1$) is same as Game_{1-k-1} except the k^{th} queried key is sf-type 2.

– Game_{1-k-3} (for $1 \leq k \leq q_1$) is same as Game_{1-k-2} except the k^{th} queried key is sf-type 3.

– $\text{Game}_{1-(q_1+1)-i}$ (for $1 \leq i \leq 3$) is same as Game_{1-q_1-3} except the last q_2 queried keys are of sf-type i.

– In Game_{2-k-1} (for $1 \leq k \leq \nu$) is same as $\text{Game}_{2-(k-1)-2}$ except the k^{th} decryption query is answered by sf-type 1 alt-key. (In this sequel, we define $\text{Game}_{2-0-2} = \text{Game}_{1-(q_1+1)-3}$)

– Game_{2-k-2} (for $1 \leq k \leq \nu$) is same as Game_{2-k-1} except the k^{th} decryption query is answered by sf-type 2 alt-key.

– Game_{Final} is similar to $\text{Game}_{2-\nu-2}$ except that the challenge ciphertext is of sf-type 2.

In Game$_{Final}$, the challenge message m_b is masked with an independently and uniformly chosen element from $\mathbb{G}_T$ implying the component C_{INT} does not leak any information about the challenge message m_b. Therefore, the adversary $\mathscr{A}$ has no advantage in Game$_{Final}$. The outline of the hybrid arguments over the games are structured in the box (for details of the lemmas, refer to Appendix E):

$$\boxed{Real} \xRightarrow{\substack{\boxed{\text{Lem E.1}}\\ \boxed{\text{DSG2}}}} \boxed{Res} \xRightarrow{\substack{\boxed{\text{Lem E.2}}\\ \boxed{\text{DSG1}}}} \boxed{0} \xRightarrow{\substack{\boxed{\text{Lem E.3}}\\ \boxed{\text{DSG2}}}} \boxed{1-1-1}\ldots\boxed{1-(k-1)-3} \xRightarrow{\substack{\boxed{\text{Lem E.3}}\\ \boxed{\text{DSG2}}}} \boxed{1-k-1}$$

$$\boxed{1-k-1} \xRightarrow{\substack{\boxed{\text{Lem E.4}}\\ \boxed{\text{CHM}}}} \boxed{1-k-2} \xRightarrow{\substack{\boxed{\text{Lem E.5}}\\ \boxed{\text{DSG2}}}} \boxed{1-k-3}\ldots\boxed{1-q_1-3} \xRightarrow{\substack{\boxed{\text{Lem E.6}}\\ \boxed{\text{DSG2}}}} \boxed{1-(q_1+1)-1}$$

$$\boxed{1-(q_1+1)-1} \xRightarrow{\substack{\boxed{\text{Lem E.7}}\\ \boxed{\text{SHM}}}} \boxed{1-(q_1+1)-2} \xRightarrow{\substack{\boxed{\text{Lem E.8}}\\ \boxed{\text{DSG2}}}} \boxed{1-(q_1+1)-3} \xRightarrow{\substack{\boxed{\text{Lem E.9}}\\ \boxed{\text{DSG2, CRH}}}} \boxed{2-1-1}$$

$$\boxed{2-1-1}\ldots\boxed{2-(k-1)-2} \xRightarrow{\substack{\boxed{\text{Lem E.9}}\\ \boxed{\text{DSG2, CRH}}}} \boxed{2-k-1} \xRightarrow{\substack{\boxed{\text{Lem E.10}}\\ \boxed{\text{DSG2}}}} \boxed{2-k-2}\ldots\boxed{2-\nu-2} \xRightarrow{\substack{\boxed{\text{Lem E.11}}\\ \boxed{\text{DSG3}}}} \boxed{Final}$$

Using the above structure, we have the following reduction:

$$\begin{aligned}\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PE-CCA}}(\kappa) \leq\ & \mathsf{Adv}_{\mathscr{B}_1}^{\mathrm{DSG1}}(\kappa) + (2q_1+2\nu+3)\mathsf{Adv}_{\mathscr{B}_2}^{\mathrm{DSG2}}(\kappa) + q_1\mathsf{Adv}_{\mathscr{B}_3}^{\mathrm{P-CMH}}(\kappa) \\ & + \mathsf{Adv}_{\mathscr{B}_4}^{\mathrm{P-SMH}}(\kappa) + \nu\mathsf{Adv}_{\mathscr{B}_5}^{\mathrm{CRH}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_6}^{\mathrm{DSG3}}(\kappa)\end{aligned}$$

where $\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4, \mathscr{B}_5$ and $\mathscr{B}_6$ are PPT algorithms whose running times are same as that of $\mathscr{A}$. This completes the theorem. □

Theorem 5.3. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies the conditions (1) and (3) of* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has* PMH *security, the assumptions, DSG1, DSG2 and DSG3 hold in* $\mathscr{J}$ *and* H *is a collision resistant hash function, then the proposed predicate encryption scheme,* PE *in sec.5.1 for the predicate* $\sim$ *is adaptive-predicate IND-CCA secure.*

Proof. Similar to the proof of the Theorem 5.2. The reduction of the proof is given by

$$\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PE-CCA}}(\kappa) \leq \mathsf{Adv}_{\mathscr{B}_1}^{\mathrm{DSG1}}(\kappa) + (2q+2\nu+1)\mathsf{Adv}_{\mathscr{B}_2}^{\mathrm{DSG2}}(\kappa) + \nu\mathsf{Adv}_{\mathscr{B}_3}^{\mathrm{CRH}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_4}^{\mathrm{DSG3}}(\kappa)$$

where q and ν respectively be the number of key and decryption queries made $\mathscr{A}$ and $\mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4$ are PPT algorithms whose running times are same as that of $\mathscr{A}$. This completes the theorem. □

6 Framework for Predicate Signcryption

In this section, we present the predicate signcryption schemes (for syntax and security definition of PSC, we refer to Appendix C) from the pair encoding schemes. The proposed signcryption satisfies the "combined-setup" [31], i.e., the distributions of the public parameters and keys of the implicit primitive schemes, PS and PE are identical. Here we consider the signcryptions in $\mathcal{C}t\mathcal{E}\&\mathcal{S}t\mathcal{S}$-paradigm of [31] to guarantee the faster execution of the subroutines in $\mathsf{Signcrypt}$ and $\mathsf{Unsigncrypt}$, stronger security and the publicly verifiability (non-repudiation).

Let $\mathsf{PS} := (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{PS.Sign}, \mathsf{PS.Ver})$ and $\mathsf{PE} := (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{PE.Encrypt}, \mathsf{PE.Decrypt})$ respectively be the predicate signature scheme in sec.3.6 and predicate encryption scheme in sec.5.1 constructed form a pair encoding scheme $\mathsf{P} := (\mathsf{Param}, \mathsf{Enc1}, \mathsf{Enc2}, \mathsf{Pair})$ for predicate $\sim$. Let $\mathsf{OTS} := (\mathsf{OTS.Gen}, \mathsf{OTS.Sign}, \mathsf{OTS.Ver})$ and $\mathcal{C} := (\mathsf{C.Setup}, \mathsf{Commit}, \mathsf{Open})$ respectively be the one-time signature scheme and commitment scheme. To distinguish the hash values, $\hbar_s$ and $\hbar_e$ involved in PS and PE, we keep the first argument of the hash function, H to be 1 and 0 respectively, viz., $\hbar_s := H(1, \mathsf{vk}, y_s)$ and $\hbar_e := H(0, \mathsf{com}, \delta_{y_s}, \mathsf{vk}, C_{\mathsf{cpa}})$

– $\mathsf{Setup}(1^\kappa, \boldsymbol{j})$: Same as the Setup in sec.3.6 except the $\mathcal{PP}$ additionally contains the public commitment key, $\mathcal{CK}$.

– $\mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$: Same as the KeyGen in sec.3.6.

– $$\mathsf{Signcrypt}(m, \mathcal{SK}_x, y_s, y_e) := \left(\begin{array}{c} \boxed{(\mathsf{com}, \mathsf{decom}) := \mathsf{Commit}(m) \parallel (\mathsf{vk}, \mathsf{signk}) := \mathsf{Gen}(1^\kappa)} \\ \boxed{\delta_{y_s} := \mathsf{PS.Sign}(\mathsf{vk}, \mathcal{SK}_x, y_s) \parallel C_{\mathsf{cpa}} := \mathsf{PE.Encrypt}^*(\mathsf{decom}, y_e)} \\ \boxed{\hbar_e := H(0, \mathsf{com}, \delta_{y_s}, \mathsf{vk}, C_{\mathsf{cpa}})} \\ \boxed{C_0 := g^{s(\theta_1 \hbar_e + \theta_2)}, \text{where } s \text{ is a randomness in } \mathsf{PE.Encrypt}^*} \\ \boxed{\delta_o := \mathsf{OTS.Sign}(C_0 || y_s, \mathsf{signk})} \\ \boxed{\text{returns } \mathsf{U} := (\mathsf{com}, \delta := (\delta_{y_s}, \delta_o, \mathsf{vk}), \mathsf{CT} := (C_{\mathsf{cpa}}, C_0))} \end{array}\right)$$

– $$\mathsf{Unsigncrypt}(\mathsf{U}, \mathcal{SK}_x, y_s) := \begin{cases} m & \text{if } \left(\begin{array}{c} \boxed{\mathsf{OTS.Ver}(C_0 || y_s, \delta_o, \mathsf{vk}) = 1} \\ \boxed{\mathsf{PS.Ver}(\mathsf{vk}, \delta_{y_s}, y_s) = 1 \parallel \text{ let } d := \mathsf{PE.Decrypt}(\mathsf{CT}, \mathcal{SK}_x)} \\ \boxed{\text{let } m := \mathsf{Open}(\mathsf{com}, d)} \end{array}\right) \\ \perp & \text{otherwise.} \end{cases}$$

Correctness. Follows from the correctness of primitive predicate signature scheme, PS, predicate encryption scheme, PE, commitment scheme, $\mathcal{C}$ and one-time signature scheme, OTS.

Remark 6.1. Note that the CCA secure ciphertext is also represented as $\mathsf{CT} := (C_{\mathsf{cpa}}, C_0)$, where the routine $\mathsf{Encrypt}$ can be thought as subroutine $\mathsf{Encrypt}^*$ (the $\mathsf{Encrypt}$ of CPA construction in [2]), then followed by the computation of C_0.

Fact 6.2. Following the Facts 3.6 and 5.4, we have if $\boldsymbol{c}_y$ of the underlying pair encoding scheme has constant-size, then the corresponding signcryption will be constant-size and the number of pairings in $\mathsf{Unsigncrypt}$ is constant-size. In other word, if $|\boldsymbol{c}_y| = \omega_1$, then size of the signcryption (mainly the #group elements) is $2\omega_1 + 3$ and the number of pairings in $\mathsf{Unsigncrypt}$ is $2(\omega_1 + 1)$. Since Ver and $\mathsf{Decrypt}$ run in parallel in $\mathsf{Unsigncrypt}$, the number of pairings are counted to be $\omega_1 + 1$.

Remark 6.3. The $\mathsf{Signcrypt}$ and $\mathsf{Unsigncrypt}$ work almost in black-box manner using the black-boxes, OTS scheme, commitment scheme, predicate signature scheme in sec.3.6 and predicate encryption scheme sec.5.1 except the $\hbar_e$ is computed as $H(0, \mathsf{com}, \delta_{y_s}, \mathsf{vk}, C_{\mathsf{cpa}})$ in stead of $H(C_{\mathsf{cpa}})$ in $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$.

Discussion 6.4. We see later that for confidentiality (resp. unforgeability) of the proposed signcryption, we require hiding (resp. no) property of the primitive commitment scheme, $\mathcal{C}$. However, to assure the non-repudiation, we have to rely on the relax-binding property of $\mathcal{C}$ as discussed in [31]. Before moving to the security section, we pay our attention on the variant of signcryptions:

– If we ignore the commitment stuffs from the construction, in that case $C_{\mathsf{cpa}} \leftarrow \mathsf{PE.Encrypt}^*(m, y_e)$, then this variant of signcryption has the same performance as the proposed signcryption except the non-repudiation. Since, the non-repudiation is not attained, so the only security of the primitive commitment scheme required to go through the proof is the hiding property.

– If we ignore the OTS scheme and apply the modification, $\delta_{y_s} \leftarrow \mathsf{PS.Sign}(\mathsf{com}||y_e, \mathcal{SK}_x, y_e)$, we have the same results as the proposed signcryption except we have the weak unforgeability. For proving the weak unforgeability, we require the relax-binding property of $\mathcal{C}$.

6.1 Security of Signcryption

Theorem 6.1. *Our proposed predicate signcryption scheme in sec.6 is perfectly private.*

Proof. Since, the $\mathcal{SK}_x$ is only used to generate δ_{y_s} and the PS scheme in sec.3.6 is perfectly private, so we are done. □

Theorem 6.2. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has both the security,* SMH *and* CMH*, the assumptions, DSG1, DSG2 and DSG3 hold in* $\mathcal{J}$*, the one-time signature scheme,* OTS *has strong unforgeability, the commitment scheme,* $\mathcal{C}$ *has the hiding property and* H *is a collision resistant hash function, then the proposed predicate signcryption scheme,* PSC *in sec.6 for the predicate* $\sim$ *is adaptive-predicates IND-CCA secure.*

Proof. We refer to Appendix F. □

Theorem 6.3. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has the* CMH *security, the assumptions, DSG1, DSG2 and DSG3 hold in* $\mathcal{J}$*, the one-time signature scheme,* OTS *has strong unforgeability and* H *is a collision resistant hash function, then the proposed predicate signcryption scheme,* PSC *in sec.6 for the predicate* $\sim$ *is adaptive-predicates strong unforgeable.*

Proof. We refer to Appendix F. □

6.2 Instantiations of Predicate Signcryption

We provide the first black-box construction of predicate signcryption schemes from the pair encoding schemes. All the signcryption schemes are shown to be strong unforgeable and IND-CCA secure in the adaptive-predicates model and achieve the signer privacy. Since, the Sign (resp. Ver) and $\mathsf{Encrypt}$ (resp. $\mathsf{Decrypt}$) run in parallel in $\mathsf{Signcrypt}$ (resp. $\mathsf{Unsigncrypt}$), the execution is faster as compared to other approach. Using the different pair encoding schemes [2, 5], we instantiate various signcryptions which are listed below:

– Obtained here are the predicate signcryptions for regular languages in both policies, key-policy (KP) and signcryption-policy (SCP) which support the large universe alphabet set. The schemes are constructed respectively from the pair encoding schemes, Scheme 3 and Scheme 7 of [2].

– The unbounded ABSC schemes with large universes in KP and SCP flavors are instantiated using the pair encoding Scheme 4 of [2]) and its dual [5] respectively.

– We provide a KP-ABSC scheme with constant-size signcryptions and the number of pairings required to unsigncrypt is also constant. The signcryption is constructed from the pair encoding scheme, Scheme 5 of [2]. Since, $|\boldsymbol{c}_y| = 6$, following the **Fact 6.2**, we have $|\mathsf{U}| = 15$ and number of pairings in $\mathsf{Unsigncrypt}$ is 14. Similarly, by applying dual [5] on the Scheme 5 of [2], we obtain SCP-ABSC constant-size short keys.

– A new class predicate signcryption, key-policy over doubly-spatial signcryption (KP-DSSC) is constructed from the pair encoding scheme, Scheme 6 of [2]. Again, by applying dual [5] on the Scheme 6 of [2], we obtain another class predicate signcryption, signcryption-policy over doubly-spatial signcryption (SCP-DSSC). Similar to KP-DSS (resp. SP-DSS), the new class, KP-DSSC (resp. SCP-DSSC) generalizes the existing class, KP-ABSC (resp. SCP-ABSC).

– The cost-free ABSC schemes in both KP and SCP flavors with small universes are constructed from the pair encoding schemes, Scheme 8 and Scheme 10 of [2] respectively.

– The cost-free KP-ABSC and SCP-ABSC schemes with large universes are constructed respectively from the pair encoding schemes, Scheme 12 and Scheme 13 of [2].

7 Conclusion and Future Work

In this paper, we first time showed that the pair encodings provide the adaptively unforgeable predicate signatures with prefect privacy. Then, we have shown that the pair encodings can also be applied to construct fully (CCA) secure predicate encryption with almost the same cost as the CPA-secure PE of [2]. Finally, we explored a generic framework for predicate signcryptions using the pair encodings. We instantiated many practical schemes for all constructions. In future, similar to the prime order variants [3, 13] of [2, 34], we will be focusing for the prime order variant of all the constructions in this paper.

References

[1] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In *EUROCRYPT*, volume 2332 of *LNCS*, pages 83–107. Springer, 2002.

[2] Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *EUROCRYPT*, volume 8441 of *LNCS*, pages 557–577. Springer, 2014.

[3] Nuttapong Attrapadung. Dual system encryption framework in prime-order groups. Cryptology ePrint Archive, Report 2015/390, 2015. `http://eprint.iacr.org/`.

[4] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. Conversions among several classes of predicate encryption and applications to abe with various compactness tradeoffs. Cryptology ePrint Archive, Report 2015/431, 2015. `http://eprint.iacr.org/`.

[5] Nuttapong Attrapadung and Shota Yamada. Duality in abe: Converting attribute based encryption for dual predicate and dual policy via computational encodings. In *CT-RSA*, volume 9048 of *LNCS*, pages 616–637. Springer, 2015.

[6] Mihir Bellare and Georg Fuchsbauer. Policy-based signatures. In *PKC*, volume 8383 of *LNCS*, pages 520–537. Springer, 2014.

[7] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: a new vision for public-key cryptography. *Commun. ACM*, 55(11):56–64, 2012.

[8] Xavier Boyen. Mesh signatures. In *EUROCRYPT*, volume 4515 of *LNCS*, pages 210–227. Springer, 2007.

[9] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *PKC*, volume 8383 of *LNCS*, pages 501–519. Springer, 2014.

[10] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT*, volume 3027 of *LNCS*. Springer, 2004.

[11] David Chaum and Eugéne van Heyst. Group signatures. In *EUROCRYPT*, volume 547 of *LNCS*, pages 257–265. Springer, 1991.

[12] Cheng Chen, Jie Chen, Hoon Wei Lim, Zhenfeng Zhang, and Dengguo Feng. Combined public-key schemes: The case of ABE and ABS. In *PROVSEC*, volume 7496 of *LNCS*, pages 53–69. Springer, 2012.

[13] Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system abe in prime-order groups via predicate encodings. In *EUROCRYPT*, volume 9057 of *LNCS*, pages 595–624. Springer, 2015.

[14] Pratish Datta, Ratna Dutta, and Sourav Mukhopadhyay. Functional signcryption: Notion, construction, and applications. Cryptology ePrint Archive, Report 2015/913, 2015. `http://eprint.iacr.org/`.

[15] Keita Emura, Atsuko Miyaji, and Mohammad Shahriar Rahman. Dynamic attributebased signcryption without random oracles. *International Journal of Applied Cryptography*, 2(11):199–211, 2012.

[16] Paul Erdös, Peter Frankl, and Zoltán Füredi. Families of finite sets in which no set is covered by the union of r others. *Israel Journal of Mathematics*, 51(1-2):79–89, 1985.

[17] Martin Gagné, Shivaramakrishnan Narayan, and Reihaneh Safavi-Naini. Threshold attribute-based signcryption. In *SCN*, volume 6280 of *LNCS*, pages 154–171. Springer, 2010.

[18] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98. ACM, 2006.

[19] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, volume 4965 of *LNCS*, pages 415–432. Springer, 2008.

[20] Stuart Haber and Benny Pinkas. Securely combining public-key cryptosystems. In *ACM Conference on Computer and Communications Security*, pages 215–224. ACM, 2001.

[21] Mike Hamburg. Spatial encryption. Cryptology ePrint Archive, Report 2011/389, 2011. `http://eprint.iacr.org/`.

[22] Ravi Kumar, Sridhar Rajagopalan, and Amit Sahai. Coding constructions for blacklisting problems without computational assumptions. In *CRYPTO*, volume 1666 of *LNCS*, pages 609–623. Springer, 1999.

[23] Allison Lewko and Brent Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In *TCC*, volume 5978 of *LNCS*, pages 455–479. Springer, 2010.

[24] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, volume 6110 of *LNCS*, pages 62–91. Springer, 2010.

[25] Benoît Libert and Jean-Jacques Quisquater. Efficient signcryption with key privacy from gap diffie-hellman groups. In *PKC*, volume 2947 of *LNCS*, pages 187–200. Springer, 2004.

[26] Hemanta Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. Cryptology ePrint Archive, Report 2008/328, 2008. `http://eprint.iacr.org/`.

[27] Hemanta Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In *CT-RSA*, volume 6558 of *LNCS*, pages 376–392. Springer, 2011.

[28] Takahiro Matsuda, Kanta Matsuura, and Jacob C. N. Schuldt. Efficient constructions of signcryption schemes and signcryption composability. In *INDOCRYPT*, volume 5922 of *LNCS*, pages 321–342. Springer, 2009.

[29] Mridul Nandi and Tapas Pandit. Generic conversions from cpa to cca secure functional encryption. Cryptology ePrint Archive, Report 2015/457, 2015. `http://eprint.iacr.org/`.

[30] Tatsuaki Okamoto and Katsuyuki Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In *Public Key Cryptography*, volume 6571 of *LNCS*, pages 35–52. Springer, 2011.

[31] Tapas Pandit, Sumit Kumar Pandey, and Rana Barua. Attribute-based signcryption : Signer privacy, strong unforgeability and ind-cca2 security in adaptive-predicates attack. In *PROVSEC*, volume 8782 of *LNCS*, page 274290. Springer, 2014.

[32] Y. Sreenivasa Rao and Ratna Dutta. Expressive bandwidth-efficient attribute based signature and signcryption in standard model. In *ACISP*, volume 8544 of *LNCS*, pages 209–225. Springer, 2014.

[33] Brent Waters. Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO*, LNCS, pages 619–636. Springer, 2009.

[34] Hoeteck Wee. Dual system encryption via predicate encodings. In *TCC*, volume 8349 of *LNCS*, pages 455–479. Springer, 2014.

[35] Shota Yamada, Nuttapong Attrapadung, Goichiro Hanaoka, and Noboru Kunihiro. Generic constructions for chosen-ciphertext secure attribute based encryption. In *Public Key Cryptography*, volume 6571 of *LNCS*, pages 71–89. Springer, 2011.

[36] Shota Yamada, Nuttapong Attrapadung, Bagus Santoso, Jacob C. N. Schuldt, Goichiro Hanaoka, and Noboru Kunihiro. Verifiable predicate encryption and applications to cca security and anonymous predicate authentication. In *Public Key Cryptography*, volume 7293 of *LNCS*, pages 243–261. Springer, 2012.

[37] Yuliang Zheng. Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In *CRYPTO*, volume 1294 of *LNCS*, pages 165–179. Springer, 1997.

A Some Results of Linear Algebra

Let's recall the three types of elementary row operations on a matrix.

- **type**-1: Interchange rows i and j (in sort, we write $R_i \leftrightarrow R_j$).
- **type**-2: Multiply row i by k, with $k \neq 0$ (in sort, $kR_i \to R_i$).
- **type**-3: Add k times row i to row j (in sort, $R_i + kR_j \to R_i$).

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

Definition A.1. 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 A.2. A 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 and (2) each column of R which contains the leading non-zero entry of some row has all its other entries 0.

Definition A.3. 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 rigorously is given below.

Theorem A.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 equivalent.

Theorem A.2. *Suppose the matrix $\boldsymbol{M}$ is obtained from $\boldsymbol{B}$ by applying n elementary column operations, i.e., $\boldsymbol{B}\boldsymbol{\mathcal{E}}_1^\top \boldsymbol{\mathcal{E}}_2^\top \cdots \boldsymbol{\mathcal{E}}_n^\top = \boldsymbol{M}$, where $\boldsymbol{\mathcal{E}}_i$'s are elementary matrices. Then, $\boldsymbol{v}$ is a solution of the system $\boldsymbol{MX} = \boldsymbol{0}$ if and only if $\boldsymbol{\mathcal{E}}_1^\top \boldsymbol{\mathcal{E}}_2^\top \cdots \boldsymbol{\mathcal{E}}_n^\top \boldsymbol{v}$ is a solution of $\boldsymbol{BX} = \boldsymbol{0}$.*

Theorem A.3. *Let R be a ring with 1. Let $\boldsymbol{B} \in R^{m\times n}$ be a matrix such that $B_{i1} = \delta_{i,1}$, where $\delta_{i,1}$ is a Kronecker delta function. Let $\boldsymbol{B}_{\mathsf{M}} \in R^{m\times(n+1)}$ be a matrix defined by*

$$\boldsymbol{B}_{\mathsf{M}} := \begin{bmatrix} t & \\ 0 & \\ \vdots & \boldsymbol{B} \\ 0 & \end{bmatrix}, \text{ for } t \in R.$$

Then, $(v_1 \ldots, v_n)^\top$ is a solution of $\boldsymbol{BX} = \boldsymbol{0}$ if and only if for each $v_0 \in R$, $(v_0, -tv_0 + v_1, v_2, \ldots, v_n)$ is a solution of the system $\boldsymbol{B}_{\mathsf{M}}\boldsymbol{X} = \boldsymbol{0}$.

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

Assumption: The factorization problem is intractable. For our purpose, we mainly apply the elementary row operations of type-2 and type-3, but for simple representation of the solutions, one may use elementary row and column operations of type-1. The Theorem A.1 and A.2 assume the fact that $k \neq 0$ (involved in type-2 operation) implies that k in invertible. When the matrices are considered over a field, we do not have any problem. But if we are not in 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 could help out from the said trouble. Of course it does: let $0 \neq k \in \mathbb{Z}_N$, then we show that k is co-prime to N which in turn implies that k is invertible in $\mathbb{Z}_N$. If k is not co-prime to N, then one can break the factorization problem through the finding $gcd(k, N) > 1$ in polynomial time of the security parameter, κ.

A.1 Scheme 4 : Unbounded KP-ABE with Large Universes of [2]

Here we consider the pair encoding, Scheme 4 of [2] which realized the unbounded KP-ABE with large universe. We show that this pair encoding satisfies the **Conditions 3.2**. The condition (1) is so obvious. To verify the condition (2), we see that for each the random coin s_i, there is a component c_ι such that $c_\iota(\boldsymbol{s}, \boldsymbol{h}) = s_i$. Therefore, it is an example, where all the coins are h-free. For verifying the condition (3), we first notice that only $b_1 \neq 0$. Hence, we have to show that $E_{1j} = 0$ for $j \in [2, \omega_1]$. We find from the correctness of the scheme that the monomials containing k_1 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.

Scheme 4 : Unbounded KP-ABE with Large Universes of [2]
$\mathsf{Param} \to 6.\ \boldsymbol{h} := (h_0, h_1, \phi_1, \phi_2, \phi_3, \eta)$
$\mathsf{Enc1}(\Gamma := (\boldsymbol{M}, \rho)) \to \boldsymbol{k}(\alpha, \boldsymbol{r}, \boldsymbol{h}) := (k_1, k_2, k_3, \{k_{4,i}, k_{5,i}, k_{6,i}\}_{i\in[\ell]})$, 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 $c_1 := s$, $c_2 := s\eta$, $c_3 := s\phi_1 + w\phi_2$, $c_4 := w$, $c_{5,y} := w\phi_3 + s_y(h_0 + h_1y)$, $c_{6,y} := s_y$ and $\boldsymbol{s} := (s, w, \{s_y\}_{y\in S})$
$\mathsf{Correctness}$: If $x \sim y$, i.e., $\Gamma(S) = \mathsf{True}$, there exists reconstruction coefficients $\{\mu_i\}_{i\in\mathcal{I}}$, with let $\mathcal{I} := \{i \in [\ell] \mid \rho(i) \in S\}$ s.t $\sum_{i\in\mathcal{I}} \mu_i\boldsymbol{M}_i\boldsymbol{v}^\top = v_1 = r\phi_2$. So the following linear combination reveals αs as : $k_1c_1 - k_2c_2 - k_3c_3 + \sum_{i\in\mathcal{I}} \mu_i(k_{4,i}c_4 - k_{5,i}c_{5,\rho(i)} + k_{6,i}c_{6,\rho(i)}) = \alpha s - rw\phi_2 + \sum_{i\in\mathcal{I}} \mu_i(\boldsymbol{M}_i\boldsymbol{v}^\top w) = \alpha s$

Example A.2. To understand that the process of sampling for the Scheme 4, we customize the set of attributes S. Let $S := \{y_2, y_3, y_4\} \subset \mathbb{Z}_N$. $\mathsf{Enc2}(S) \to \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) := (c_1 := s, c_2 := s\eta, c_3 := s\phi_1 + w\phi_2, c_4 := w, \{c_{5,y}, c_{6,y}\}_{y\in S})$, where $c_{5,y} := w\phi_3 + s_y(h_0 + h_1y)$, $c_{6,y} := s_y$ and $\boldsymbol{s} := (s_0 := s, s_1 := w, s_2, s_3, s_4)$ with $s_i := s_{y_i}$. This is a case, where **all the coins are h-free**. The matrix[7] of the system is given by

$$\boldsymbol{A}^\top := \begin{pmatrix} \boxed{1} & \eta & \phi_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \phi_2 & \boxed{1} & \phi_3 & 0 & \phi_3 & 0 & \phi_3 & 0 \\ 0 & 0 & 0 & 0 & h_0 + h_1y_2 & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 + h_1y_3 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_0 + h_1y_4 & \boxed{1} \end{pmatrix}$$

[7]The box in the j^{th} row indicates that the coin s_j is h-free and the column containing the box is the leading h-free column for the j^{th} row.

Here $S_{\mathsf{hf}} := \{1,4,6,8,10\}$, $T_{\mathsf{hf}} := \{0,1,2,3,4\}$, $S_{\mathsf{phf}} := \{(1,0),(4,1),(6,2),(8,3),(10,4)\}$, $S_{\mathsf{non\text{-}fv},0} := \{1\}$, $S_{\mathsf{non\text{-}fv},0} := \{1\}$, $S_{\mathsf{non\text{-}fv},1} := \{4\}$, $S_{\mathsf{non\text{-}fv},2} := \{6\}$, $S_{\mathsf{non\text{-}fv},3} := \{8\}$, $S_{\mathsf{non\text{-}fv},4} := \{10\}$, $S_{\mathsf{non\text{-}fv}} := \{1,4,6,8,10\}$ and $S_{\mathsf{fv}} := \{2,3,5,7,9\}$. Therefore, x_i are chosen randomly from $\mathbb{Z}_N$ for $i \in S_{\mathsf{fv}}$. The non-free variables are computed as: $x_1 := -\eta x_2 - \phi_2 x_3$, $x_4 := -\phi_2 x_3 - \phi_3(x_5 + x_7 + x_9)$, $x_6 := -(h_0 + h_1 y_2)x_5$, $x_8 := -(h_0 + h_1 y_3)x_7$, $x_{10} := -(h_0 + h_1 y_4)x_7$

A.2 Scheme 10 : CP-ABE with Small Universes of [2]

This pair encoding scheme was extracted from the fully secure CP-ABE [24]. Again the condition (1) is obvious. For the random coins $s, s_1, \ldots, s_\ell$, the condition 2(a) holds. But, for $v_2, \ldots, v_k$, the condition 2(b) holds. For all the v_j, the unique $h_{i'}$ is ϕ. So, we require that during setup ϕ is chosen to be co-prime to N. The condition (3) works out through the similar argument as in sec.A.1.

Scheme 10 : CP-ABE with Small Universes of [2]
$\mathsf{Param}(\lvert\mathcal{U}\rvert) \to \lvert\mathcal{U}\rvert + 1$. $\boldsymbol{h} := (\phi, \{h_i\}_{i\in\mathcal{U}})$
$\mathsf{Enc1}(S \subseteq \mathcal{U}) \to \boldsymbol{k}(\alpha, \boldsymbol{r}, \boldsymbol{h}) := (k_1 := \alpha + \phi r, \{k_{2,x} := r h_x\}_{x\in S}, k_3 := r)$, where $\boldsymbol{r} := r$.
For $\Gamma := (\boldsymbol{M}, \rho)$, where $\boldsymbol{M} \in \mathbb{Z}_N^{\ell\times k}$ and $\rho : [\ell] \to \mathcal{U}$, an injective row labeling $\mathsf{Enc2}(\Gamma := (\boldsymbol{M}, \rho)) \to \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) := (c_1, \{c_{2,i}, c_{3,i}\}_{i\in[\ell]})$, where $c_1 := s$, $c_{2,i} := \phi \boldsymbol{M}_i \boldsymbol{v}^\top + s_i h_{\rho(i)}$, $c_{3,i} := s_i$ and $\boldsymbol{s} := (s, v_2, \ldots, v_k, s_1, \ldots, s_\ell)$, $\boldsymbol{v} := (s, v_2, \ldots, v_k)$
$\mathsf{Correctness}$: If $\Gamma(S) = \mathsf{True}$, we have $\sum_{i\in\mathcal{I}} \mu_i \boldsymbol{M}_i \boldsymbol{v}^\top = \alpha$. So the following linear combination reveals αs as : $k_1 c_1 + \sum_{i\in\mathcal{I}} \mu_i (k_3 c_{2,i} - k_{2,\rho(i)} c_{3,i}) = \alpha s$

Example A.3. To understand that the process of sampling for the Scheme 10 of [2], we customize the monotone access structure. Let $\Gamma := (\boldsymbol{M}, \rho)$, where

$$\boldsymbol{M} := \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 2 & 1 \\ 3 & 1 & 3 \end{pmatrix} \text{ and } \rho : [4] \to \mathcal{U} \text{ is some row labeling function.}$$

$\mathsf{Enc2}(\Gamma) \to \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{h}) := (c_1, \{c_{2,i}, c_{3,i}\}_{i\in[4]})$, 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 $\boldsymbol{s} := (s_0 := s', s_1 := v_2, s_2 := v_3, s_3 := s'_1, s_4 := s'_2, s_5 := s'_3, s_6 := s'_4)$, $\boldsymbol{v} := (s', v_2, v_3)$. This is a case, where **all the coins are not h-free**. For all non h-free coins there is unique h-term which is ϕ. The matrix of the system is given by

$$\boldsymbol{A}^\top := \begin{pmatrix} \boxed{1} & \phi & 0 & 2\phi & 0 & 3\phi & 0 & 3\phi & 0 \\ 0 & 2\phi & 0 & 3\phi & 0 & 2\phi & 0 & \phi & 0 \\ 0 & 3\phi & 0 & 4\phi & 0 & \phi & 0 & 3\phi & 0 \\ 0 & h_{\rho(1)} & \boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_{\rho(2)} & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & h_{\rho(3)} & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_{\rho(4)} & \boxed{1} \end{pmatrix}$$

Here $S_{\mathsf{hf}} := \{1,3,5,7,9\}$, $S_{\mathsf{non\text{-}hf}} := \{2,4,6,8\}$, $T_{\mathsf{hf}} := \{0,3,4,5,6\}$, $T_{\mathsf{non\text{-}hf}} := \{1,2\}$, $S_{\mathsf{phf}} := \{(1,0),(3,3),(5,4),(7,5),(9,6)\}$, $S_{\mathsf{non\text{-}fv},0} := \{1\}$, $S_{\mathsf{non\text{-}fv},3} := \{3\}$, $S_{\mathsf{non\text{-}fv},4} := \{5\}$, $S_{\mathsf{non\text{-}fv},5} := \{7\}$, $S_{\mathsf{non\text{-}fv},6} := \{9\}$ and $S_{\mathsf{non\text{-}fv}} := \{1,4,6,8,10\}$. We now apply the sequence of elementary row operations of type-2 and type-3 to make each non h-free row reduced: $\phi^{-1}R_2 \to R_2$, $\phi^{-1}R_3 \to R_3$, $2^{-1}R_2 \to R_2$,

$R_1 + (-\phi)R_2 \to R_1$, $R_3 + (-3)R_2 \to R_3$, $R_4 + (-h_{\rho(1)})R_2 \to R_4$, $(-2)R_3 \to R_3$, $R_1 + (-\phi/2)R_3 \to R_1$, $R_2 + (-3/2)R_3 \to R_2$, $R_4 + 3h_{\rho(1)}/2R_3 \to R_4$ and $R_5 + (-h_{\rho(2)})R_3 \to R_5$. The red color boxes of the row reduced matrix indicate the new leading element of the corresponding rows.

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

$S_{\mathsf{new}} := \{1, 3\}$. So $S_{\mathsf{non\text{-}fv}} := \{1, 2, 3, 4, 5, 7, 9\}$ and $S_{\mathsf{fv}} := \{6, 8\}$. The rest are same as Example A.2.

B Predicate Encryption

A predicate encryption (PE) scheme for a predicate tuple family, $\sim$ consists of four PPT algorithms - Setup, KeyGen, Encrypt and Decrypt.

- Setup: It takes a security parameter κ and a system parameter index $\boldsymbol{j}$ as input, outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$.
- KeyGen: It takes as input public parameters $\mathcal{PP}$, master secret $\mathcal{MSK}$ and a key index $x \in \mathcal{X}$ and outputs a secret key $\mathcal{SK}_x$ corresponding to x.
- Encrypt: It takes public parameters $\mathcal{PP}$, a message $m \in \mathcal{M}$ and an associated data index $y \in \mathcal{Y}$ and returns a ciphertext C, which implicitly contains y.
- Decrypt: It takes as inputs the public parameters $\mathcal{PP}$, a ciphertext C and a key $\mathcal{SK}_x$. It returns a value form $\mathcal{M} \cup \{\perp\}$.

Correctness. For all $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$, all $x \in \mathcal{X}$, $\mathcal{SK}_x \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$, all $y \in \mathcal{Y}$ and for all messages $m \in \mathcal{M}$, it is required that if $x \sim y$ then $\mathsf{Decrypt}(\mathcal{PP}, \mathsf{Encrypt}(\mathcal{PP}, m, y), \mathcal{SK}_x) = m$.

Definition B.1 (Verifiability[29]**).** A predicate encryption scheme PE with public index is said to have the verifiability if there is a PPT algorithm, Verify such that for all ciphertext C (possibly ill-format) with the public associated index y, and all $x, \tilde{x}$ with $x \sim y, \tilde{x} \sim y$ we have

$$\mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x, \tilde{x}) = 1 \Rightarrow \mathsf{Decrypt}(\mathcal{PP}, \mathsf{C}, \mathcal{SK}_x) = \mathsf{Decrypt}(\mathcal{PP}, \mathsf{C}, \mathcal{SK}_{\tilde{x}}) \tag{5}$$

and it is a weak format-verifier, i.e., it returns 1 for all correctly-format ciphertext.[8]

The property in equation (5) is called soundness of the verifiability and the weak format-verifier is called the completeness of the verifiability.

[8]So if $\mathsf{Verify}(\mathcal{PP}, \mathsf{C}, x, \tilde{x}) = 0$ for $x \sim y, \tilde{x} \sim y$ then C must be ill-format.

B.1 Security of Predicate Encryption

Definition B.2 (Adaptive-Predicate IND-CCA Security)**.** A PE scheme is said to be *adaptively CCA-secure* (AP-IND-CCA) if for all PPT adversary $\mathscr{A} := (\mathscr{A}_1, \mathscr{A}_2)$, the advantage $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PE-CCA}}(\kappa)$ is at most a negligible function in security parameter κ, where $\mathscr{A}$ is provided the access to keyGen oracle, $\mathcal{O}_K$ and decryption oracle, $\mathcal{O}_D$ and NRn is the natural restriction that (C^*, x) with $x \sim y^*$ was never queried to $\mathcal{O}_D$ and for each key index x queried to $\mathcal{O}_K$, $x \not\sim y^*$.

$$\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PE-CCA}}(\kappa) := \left| \Pr \left[\begin{array}{c} (\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j}), \\ (m_0, m_1, y^*, st) \longleftarrow \mathscr{A}_1^{\{\mathcal{O}_K,\, \mathcal{O}_D\}}(\mathcal{PP}),\ b \xleftarrow{\mathrm{U}} \{0,1\}, \\ \mathsf{C}^* \longleftarrow \mathsf{Encrypt}(\mathcal{PP}, m_b, y^*),\ b' \longleftarrow \mathscr{A}_2^{\{\mathcal{O}_K,\, \mathcal{O}_D\}}(\mathcal{PP}, \mathsf{C}^*, st) \end{array} : b = b' \wedge \mathsf{NRn} \right] - \frac{1}{2} \right|.$$

Likewise in Selective-Predicate IND-CCA (SP-IND-CCA) security, the adversary $\mathscr{A}$ submits the challenge index y^* before receiving $\mathcal{PP}$ of PE.

A weaker notion of security can be defined similarly as above except, $\mathscr{A}$ is not allowed to access to $\mathcal{O}_D$ oracle. It is called IND-CPA security in both adaptive-predicate (AP-IND-CPA) and selective predicate (SP-IND-CPA) models.

C Predicate Signcryption

A predicate signcryption (PSC) scheme for a predicate tuple family, $\sim$ consists of four PPT algorithms - Setup, KeyGen, $\mathsf{Signcrypt}$ and $\mathsf{Unsigncrypt}$.

- Setup: It takes a security parameter κ and a system parameter $\boldsymbol{j}$ as input, outputs the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$.
- KeyGen: It takes public parameters $\mathcal{PP}$, master secret $\mathcal{MSK}$ and a key index $x \in \mathcal{X}$ as input and outputs a secret key $\mathcal{SK}_x$ corresponding to x.
- $\mathsf{Signcrypt}$: It takes public parameters $\mathcal{PP}$, a message $m \in \mathcal{M}$, a signing key $\mathcal{SK}_x$, an associated index $y_s \in \mathcal{Y}$ for signer with $x \sim y_s$ and an associated index $y_e \in \mathcal{Y}$ for receiver as input and returns a signcryption U for (y_s, y_e) (we assume that U implicitly contains y_e).
- $\mathsf{Unsigncrypt}$: It takes as input public parameters $\mathcal{PP}$, a signcryption U, a secret key $\mathcal{SK}_x$ and an associated index $y_s \in \mathcal{Y}$ for signer. It returns a value from $\mathcal{M} \cup \{\perp\}$.

Correctness. For all $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$, all $m \in \mathcal{M}$, all key indices $x \in \mathcal{X}$, $\mathcal{SK}_x \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x)$, all signer associated indices $y_s \in \mathcal{Y}$ with $x \sim y_s$, all receiver associated indices $y_e \in \mathcal{Y}$, all signcryptions $\mathsf{U} \longleftarrow \mathsf{Signcrypt}(\mathcal{PP}, m, \mathcal{SK}_x, y_s, y_e)$ and all key indices $\tilde{x} \in \mathcal{X}$ with $\tilde{x} \sim y_e$, $\mathcal{SK}_{\tilde{x}} \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, \tilde{x})$, it is required that $\mathsf{Unsigncrypt}(\mathcal{PP}, \mathsf{U}, \mathcal{SK}_{\tilde{x}}, y_s) = m$.

C.1 Security of Predicate Signcryption

Definition C.1 (Signer Privacy)**.** A PSC scheme is said to be perfectly private if for all $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}$, all key indices $x_1, x_2 \in \mathcal{X}$, all keys $\mathcal{SK}_{x_1} \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x_1)$, $\mathcal{SK}_{x_2} \longleftarrow \mathsf{KeyGen}(\mathcal{PP}, \mathcal{MSK}, x_2)$, all messages $m \in \mathcal{M}$, all signer associated indices $y_s \in \mathcal{Y}$ such that $x_1 \sim y_s$ and $x_2 \sim y_s$, and all receiver associated indices $y_e \in \mathcal{Y}$, the distributions of $\mathsf{Signcrypt}(\mathcal{PP}, m, \mathcal{SK}_{x_1}, y_s, y_e)$ and $\mathsf{Signcrypt}(\mathcal{PP}, m, \mathcal{SK}_{x_2}, y_s, y_e)$ are identical.

Definition C.2 (Adaptive-Predicates IND-CCA Security)**.** An PSC scheme is said to be *adaptively CCA-secure* (APs-IND-CCA) if for all PPT adversary $\mathscr{A} := (\mathscr{A}_1, \mathscr{A}_2)$, the advantage $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PSC-CCA}}(\kappa)$ is at most a negligible function in security parameter κ, where $\mathscr{A}$ is provided the access to keyGen oracle, $\mathcal{O}_K$, signcrypt oracle, $\mathcal{O}_S$ and unsigncrypt oracle, $\mathcal{O}_U$, and NRn is the natural restriction that (U^*, x, y_s^*) with $x \sim y_e^*$ was never queried to $\mathcal{O}_U$ and for each key index x queried to $\mathcal{O}_K$, $x \not\sim y_e^*$.

$$\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PSC-CCA}}(\kappa) := \left| \Pr\left[\begin{array}{c} (\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j}), \\ (m_0, m_1, x, y_s^*, y_e^*, st) \longleftarrow \mathscr{A}_1^{\{\mathcal{O}_K,\, \mathcal{O}_S,\, \mathcal{O}_U\}}(\mathcal{PP}), \\ b \xleftarrow{\mathrm{U}} \{0,1\},\ \mathsf{U}^* \longleftarrow \mathsf{Signcrypt}(\mathcal{PP}, m_b, \mathcal{SK}_x, y_s^*, y_e^*), \\ b' \longleftarrow \mathscr{A}_2^{\{\mathcal{O}_K,\, \mathcal{O}_S,\, \mathcal{O}_U\}}(\mathcal{PP}, \mathsf{U}^*, st) \end{array} : b = b' \wedge \mathsf{NRn} \right] - \frac{1}{2} \right|.$$

Remark C.1. Likewise in *selective-predicate(s)* IND-CCA security, $\mathscr{A}$ submits either receiver associated index y_e^* or sender associated y_s^* or both before receiving $\mathcal{PP}$ of PSC.

Definition C.3 (Adaptive-Predicates Unforgeability)**.** A PSC scheme is said to be *adaptive-predicates existential unforgeable* (APs-UF-CMA) if for all PPT $\mathscr{A}$, the advantage $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PSC-UF}}(\kappa)$ is at most negligible function in κ, where $\mathscr{A}$ is provided the access to keyGen oracle, $\mathcal{O}_K$, signcrypt oracle, $\mathcal{O}_S$ and unsigncrypt oracle, $\mathcal{O}_U$, and NRn is the natural restriction that for each tuple, (m, x, y_s, y_e) queried to $\mathcal{O}_S$ oracle, $(m, y_s, y_e) \neq (m^*, y_s^*, y_e^*)$ and for each key index $x \in \mathcal{X}$ queried to $\mathcal{O}_K$ oracle, $x \not\sim y_s^*$.

$$\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PSC-UF}}(\kappa) := \Pr\left[\begin{array}{c} (\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j}), \\ (\mathsf{U}^*, y_s^*, y_e^*) \longleftarrow \mathscr{A}^{\{\mathcal{O}_K,\, \mathcal{O}_S,\, \mathcal{O}_U\}}(\mathcal{PP}), \\ m^* \longleftarrow \mathsf{Unsigncrypt}(\mathcal{PP}, \mathsf{U}^*, \mathcal{SK}_x, y_s^*, y_e^*), \\ \text{where } x \sim y_e^* \end{array} : m^* \neq \perp \wedge \mathsf{NRn} \right].$$

Remark C.2. The above unforgeability is also called *weak unforgeability* in the sense that in forgery $\mathscr{A}$ is not allowed to forge for the queried messages. In *strong unforgeability* (we use notation, APs-sUF-CMA), the restriction $(m, y_s, y_e) \neq (m^*, y_s^*, y_e^*)$ is replaced by $(\mathsf{U}, m, y_s, y_e) \neq (\mathsf{U}^*, m^*, y_s^*, y_e^*)$, where U is the reply for the query (m, x, y_s, y_e) to $\mathcal{O}_S$ oracle.

Remark C.3. Similar to the above, there is an another variant of unforgeability, called *selective-predicate(s) unforgeability* in both weak and strong sense, where $\mathscr{A}$ submits either signer index y_s^* or receiver index y_e^* or both before receiving $\mathcal{PP}$ of PSC.

D Lemmas used in Theorem 3.5 for Predicate Signature

Lemma D.1. Game$_{Real}$ *and* Game$_{Res}$ *are indistinguishable under the DSG2 assumption*[9]*. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}_{\mathscr{A},\mathrm{PS}}^{\mathrm{Real}}(\kappa) - \mathsf{Adv}_{\mathscr{A},\mathrm{PS}}^{\mathrm{Res}}(\kappa)| \leq \mathsf{Adv}_{\mathscr{B}}^{\mathrm{DSG2}}(\kappa)$.

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, $(\mathcal{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0,1\}$ is given to $\mathscr{B}$. The only difference between the games, Game$_{Real}$ and Game$_{Res}$ is that for all query key indices x and challenge associated data index y^* : $x \sim_{p_2} y^*$, 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 $p_2|F|N$. Then, there are three possibilities of F: (1) $F = p_2$, (2) $F = p_1p_2$ and (3) $F = p_2p_3$. We remark the aforesaid cases are recognized using the

[9]In Lemma 27 of [2], DSG1 and DSG2 assumptions were considered. In contrast, we show that one assumption, DSG2 can capture the intractability of factorization problem.

parameters of the given instance of DSG2. Suppose $F = p_2$. Let $B := N/F = p_1p_3$ and then by checking $T_\beta^B \overset{?}{=} \Theta$, $\mathscr{B}$ 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 $e(T_\beta, Y_2) \overset{?}{=} 1$, $\mathscr{B}$ can break the DSG2 assumption. □

Lemma D.2. Game_{Res} *and* Game_0 *are indistinguishable under the DSG1 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{\text{Res}}_{\mathscr{A},\text{PS}}(\kappa) - \mathsf{Adv}^{0}_{\mathscr{A},\text{PS}}(\kappa)| \leq \mathsf{Adv}^{\text{DSG1}}_{\mathscr{B}}(\kappa)$.

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

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

Query Phase: It consists of the following queries in adaptive manner:

– $\mathsf{KeyGen}(x)$: It is normal key. $\mathscr{B}$ can handle the key queries of $\mathscr{A}$, since the $\mathcal{MSK}$ is known to him.

– $\mathsf{Sign}(m, x, y)$: If $x \not\sim y$, it returns $\perp$. It is normal signature. $\mathscr{B}$ can answer the queries of $\mathscr{A}$, since he can construct $\mathcal{SK}_x$ using the $\mathcal{MSK}$ known to him.

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 $\hbar^* := H(m^*, y^*)$. It runs $(\boldsymbol{c}_{y^*}, \omega_2) \longleftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}^{\mathsf{M}}_{y^*} := (c_0^*, \boldsymbol{c}_{y^*})$. It picks $\boldsymbol{s}' := (s', s'_1, \ldots, s'_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. Finally, it computes a vText as $\mathcal{V} := (\mathcal{V}_{\mathsf{INT}} := e(g^\alpha, T_\beta)^{s'}, \boldsymbol{\mathcal{V}}_{y^*} := T_\beta^{\boldsymbol{c}^{\mathsf{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})})$. $\mathscr{B}$ returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathsf{INT}}$ else 0.

Analysis: We will show that all the stuffs are perfectly distributed as required. $\mathscr{B}$ 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 $g^{t_1 \boldsymbol{c}^{\mathsf{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})} = g^{\boldsymbol{c}^{\mathsf{M}}_{y^*}(t_1\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})}$ and $g_2^{t_2 \boldsymbol{c}^{\mathsf{M}}_{y^*}(\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})} = g_2^{\boldsymbol{c}^{\mathsf{M}}_{y^*}(t_2\boldsymbol{s}', \hat{\boldsymbol{h}}_{\mathsf{M}})}$. $\mathscr{B}$ implicitly sets $\boldsymbol{s} :\equiv t_1\boldsymbol{s}' \mod p_1$ and for $\beta = 1$, $\hat{\boldsymbol{s}} :\equiv t_2\boldsymbol{s}' \mod p_2$. By CRT, $\boldsymbol{s}' \mod p_1$ is independent from $\boldsymbol{s}' \mod p_2$ and therefore $\boldsymbol{s}$ and $\hat{\boldsymbol{s}}$ are perfectly distributed as required. All together, we have the stuffs simulated by $\mathscr{B}$ are identical to that of Game_{Res} if $\beta = 0$ else Game_0. □

Lemma D.3. $\text{Game}_{1-(k-1)-3}$ *and* Game_{1-k-1} *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-(k-1)-3}_{\mathscr{A},\text{PS}}(\kappa) - \mathsf{Adv}^{1-k-1}_{\mathscr{A},\text{PS}}(\kappa)| \leq \mathsf{Adv}^{\text{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq \nu_1$.

Proof. We establish a PPT simulator $\mathscr{B}$ who receives an instance of DSG2, $(\mathcal{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\text{U}} \{0,1\}$ and depending on the distribution of β, it either simulates $\text{Game}_{1-(k-1)-3}$ or Game_{1-k-1}

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

Query Phase: It consists of the following queries in adaptive manner:

– $\mathsf{KeyGen}(x)$: Let x_j be the j^{th} query key index. $\mathscr{B}$ 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 of sf-type 3 key. It runs $(\boldsymbol{k}_{x_j}, m_2) \longleftarrow \mathsf{Enc1}(x_j, N)$ with $|\boldsymbol{k}_{x_j}| = m_1$. It picks $\alpha'_j \xleftarrow{\text{U}} \mathbb{Z}_N$, $\boldsymbol{r}_j \xleftarrow{\text{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{m_1}$. It computes the sf-type 3 key as defined below.

$$\mathcal{SK}_{x_j} := g^{\boldsymbol{k}_{x_j}(\alpha, \boldsymbol{r}_j, \boldsymbol{h})}.(W_2W_3)^{\boldsymbol{k}_{x_j}(\alpha'_j, \boldsymbol{0}, \boldsymbol{0})}.\boldsymbol{R}_3$$

It implicitly sets $\hat{\alpha}_j := w_2\alpha'_j$, where $W_2W_3 = g_2^{w_2}g_3^{w_3}$. So, $\mathcal{SK}_{x_j}$ is properly distributed sf-type 3 key.
- If $j = k$ then it is either normal or sf-type 1 key. It runs $(\boldsymbol{k}_{x_k}, m_2) \longleftarrow \mathsf{Enc1}(x_k, N)$ with $|\boldsymbol{k}_{x_k}| = m_1$. It picks $\boldsymbol{r}'_k, \hat{\boldsymbol{r}}'_k \xleftarrow{\text{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{m_1}$. $\mathscr{B}$ generates $\mathcal{SK}_{x_j}$ using T_β of the instance of DSG2.

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

$\mathscr{B}$ 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 $g^{\boldsymbol{k}_{x_k}(\alpha,\ \boldsymbol{r}'_k,\ \boldsymbol{h})}.g^{t_1\boldsymbol{k}_{x_k}(0,\ \hat{\boldsymbol{r}}'_k,\ \boldsymbol{h})} = g^{\boldsymbol{k}_{x_k}(\alpha,\ \boldsymbol{r}'_k + t_1\hat{\boldsymbol{r}}'_k,\ \boldsymbol{h})}$ and $g_2^{t_2\boldsymbol{k}_{x_k}(0,\ \hat{\boldsymbol{r}}'_k,\ \boldsymbol{h})} = g_2^{\boldsymbol{k}_{x_k}(0,\ t_2\hat{\boldsymbol{r}}'_k,\ \hat{\boldsymbol{h}})}$. $\mathscr{B}$ 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 perfectly distributed normal (resp. sf-type 1) key if $\beta = 0$ (resp. $\beta = 1$).

– $\mathsf{Sign}(m, x, y)$: If $x \not\sim y$, it returns $\perp$. It is normal signature. $\mathscr{B}$ can answer the queries of $\mathscr{A}$ as the $\mathcal{MSK}$ is known to him.

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 $\hbar^* := H(m^*, y^*)$. It runs $(\boldsymbol{c}_{y^*}, \omega_2) \longleftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}_{y^*}^{\mathsf{M}} := (c_0^*, \boldsymbol{c}_{y^*})$. It picks $\boldsymbol{s}' := (s', s'_1, \ldots, s'_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. Finally, it computes a vText as $\mathcal{V} := (\mathcal{V}_{\mathsf{INT}} := e(g^\alpha, Z_1Z_2)^{s'}, \boldsymbol{\mathcal{V}}_{y^*} := (Z_1Z_2)^{\boldsymbol{c}_{y^*}^{\mathsf{M}}(\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})})$. $\mathscr{B}$ returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathsf{INT}}$ else 0.

Analysis: We will show that all the stuffs are perfectly distributed as required. Let $Z_1Z_2 = g^{z_1}g_2^{z_2}$. Then by linearity of P, we have $g^{z_1\boldsymbol{c}_{y^*}^{\mathsf{M}}(\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})} = g^{\boldsymbol{c}_{y^*}^{\mathsf{M}}(z_1\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})}$ and $g_2^{z_2\boldsymbol{c}_{y^*}^{\mathsf{M}}(\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})} = g_2^{\boldsymbol{c}_{y^*}^{\mathsf{M}}(z_2\boldsymbol{s}', \hat{\boldsymbol{h}}_{\mathsf{M}})}$. $\mathscr{B}$ implicitly sets $\boldsymbol{s} :\equiv z_1\boldsymbol{s}' \mod p_1$ and $\hat{\boldsymbol{s}} :\equiv z_2\boldsymbol{s}' \mod p_2$. By CRT, $\boldsymbol{s}' \mod p_1$ is independent from $\boldsymbol{s}' \mod p_2$ and therefore $\boldsymbol{s}$ and $\hat{\boldsymbol{s}}$ are perfectly distributed as required. All together, we have the stuffs simulated by $\mathscr{B}$ are identical to that of $\text{Game}_{1-(k-1)-3}$ if $\beta = 0$ else Game_{1-k-1}. □

Lemma D.4. Game_{1-k-1} *and* Game_{1-k-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* $|\mathsf{Adv}_{\mathscr{A},\text{PS}}^{1-k-1}(\kappa) - \mathsf{Adv}_{\mathscr{A},\text{PS}}^{1-k-2}(\kappa)| \leq \mathsf{Adv}_{\mathscr{B}}^{\text{P}-\text{CMH}}(\kappa)$ *for* $1 \leq k \leq \nu_1$.

Proof. Suppose $\mathscr{A}$ can distinguish Game_{1-k-1} and Game_{1-k-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 $\mathscr{B}$. $\mathscr{B}$ chooses $\alpha, \theta_1, \theta_2 \xleftarrow{\text{U}} \mathbb{Z}_N$, $\boldsymbol{h} \xleftarrow{\text{U}} \mathbb{Z}_N^n$ and sets $\boldsymbol{h}_{\mathsf{M}} := (\theta_1, \theta_2, \boldsymbol{h})$. Let $H : \{0,1\}^* \longrightarrow \mathbb{Z}_N$ be a hash function. Then, it provides $\mathcal{PP} := [\mathcal{J}, g, g^{\boldsymbol{h}_{\mathsf{M}}}, g_T^\alpha := e(g,g)^\alpha, Z_3 := g_3, H]$ to $\mathscr{A}$ and keeps $\mathcal{MSK} := (\alpha)$ and g_2 to itself.

Query Phase: It consists of the following queries in adaptive manner:

– $\mathsf{KeyGen}(x)$: Let x_j be the j^{th} query key index. $\mathscr{B}$ 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 of 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 of sf-type 1 or sf-type 2 key. It runs $(\boldsymbol{k}_{x_k}, m_2) \longleftarrow \mathsf{Enc1}(x_k, N)$ with $|\boldsymbol{k}_{x_k}| = m_1$. It picks $\boldsymbol{r}_k \xleftarrow{\text{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{m_1}$. $\mathscr{B}$ makes a query with x_k to $\mathcal{CH}$ and let $T := g_2^{\boldsymbol{k}_{x_k}(\beta, \hat{\boldsymbol{r}}_k, \hat{\boldsymbol{h}})}$ be the reply, where $\beta = 0$ or random element from $\mathbb{Z}_N$. Then $\mathscr{B}$ returns the following key to $\mathscr{A}$

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

  Therefore, $\mathcal{SK}_{x_j}$ is perfectly distributed sf-type 1 key if $\beta = 0$ else sf-type 2.

– $\mathsf{Sign}(m, x, y)$: If $x \not\sim y$, it returns $\perp$. It is normal signature. $\mathscr{B}$ can answer the queries of $\mathscr{A}$ as the $\mathcal{MSK}$ is known to him.

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 $\hbar^* := H(m^*, y^*)$. It runs $(\boldsymbol{c}_{y^*}, \omega_2) \longleftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}^{\mathsf{M}}_{y^*} := (c_0^*, \boldsymbol{c}_{y^*})$. It picks $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. Then, $\mathscr{B}$ makes a query with y^* to $\mathcal{CH}$ and let $D := g_2^{\boldsymbol{c}_{y^*}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}})}$ be the reply. Finally, it computes a vText as $\mathcal{V} := \big(\mathcal{V}_{\mathsf{INT}} := e(g,g)^{\alpha s}, \boldsymbol{\mathcal{V}}_{y^*} := g^{\boldsymbol{c}^{\mathsf{M}}_{y^*}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}})}.g_2^{\boldsymbol{c}^{\mathsf{M}}_{y^*}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}})}\big)$, where $g_2^{\boldsymbol{c}^{\mathsf{M}}_{y^*}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}})} := (g_2^{\hat{s}(\theta_1 \hbar^* + \theta_2)}, D)$. $\mathscr{B}$ returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathsf{INT}}$ else 0.

Analysis:

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

– $\mathsf{Perfectness}$: By the assumption: $c_{y^*,1}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}) = \hat{s}$, the first component of D is $g_2^{\hat{s}}$. So, the first component of $g_2^{\boldsymbol{c}^{\mathsf{M}}_{y^*}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}})}$ can be computed as $g_2^{\hat{s}(\theta_1 \hbar^* + \theta_2)} := (g_2^{\hat{s}})^{\theta_1 \hbar^* + \theta_2}$. $\mathscr{B}$ implicitly sets $(\hat{\theta}_1, \hat{\theta}_2) :\equiv (\theta_1, \theta_2) \mod p_2$. By CRT, $(\hat{\theta}_1, \hat{\theta}_2)$ is independent from $(\theta_1, \theta_2) \mod p_1$ and therefore $\mathcal{V}$ is perfectly distributed sf-type 1 vText. All together, we have the stuffs simulated by $\mathscr{B}$ are identical to that of Game_{1-k-1} if $\beta = 0$ else Game_{1-k-2}.

□

Lemma D.5. Game_{1-k-2} *and* Game_{1-k-3} *are indistinguishable under the DSG2 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{1-k-2}_{\mathscr{A},\text{PS}}(\kappa) - \mathsf{Adv}^{1-k-3}_{\mathscr{A},\text{PS}}(\kappa)| \leq \mathsf{Adv}^{\text{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq \nu_1$.

Proof. We establish a PPT simulator $\mathscr{B}$ who receives an instance of DSG2, $(\mathcal{J}, g, Z_1 Z_2, W_2 W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\text{U}} \{0,1\}$ and depending on the distribution of β, it either simulates Game_{1-k-2} or Game_{1-k-3}. Description of the simulation is same as that of the Lemma D.3 except the answering kth key query. Below we only describes the simulation of kth query:

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

$$\mathcal{SK}_{x_k} := g^{\boldsymbol{k}_{x_k}(\alpha, \boldsymbol{r}'_k, \boldsymbol{h})}.(W_2W_3)^{\boldsymbol{k}_{x_k}(\alpha'_k, \boldsymbol{0}, \boldsymbol{0})}.T_\beta^{\boldsymbol{k}_{x_j}(0, \hat{\boldsymbol{r}}'_k, \boldsymbol{h})}.\boldsymbol{R}_3$$

If $W_2W_3 = g_2^{w_2}g_3^{w_3}$ 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 $\hat{\boldsymbol{r}}_k := t_2\hat{\boldsymbol{r}}'_k$. 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 perfectly distributed sf-type 2 (resp. sf-type 3) key if $\beta = 1$ (resp. $\beta = 0$).

□

Lemma D.6. Game$_{2-(k-1)-2}$ *and* Game$_{2-k-1}$ *are indistinguishable under the DSG2 assumption and collision resistant property of H. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}_{\mathscr{A},\mathrm{PS}}^{2-(k-1)-2}(\kappa) - \mathsf{Adv}_{\mathscr{A},\mathrm{PS}}^{2-k-1}(\kappa)| \leq \mathsf{Adv}_{\mathscr{B}}^{\mathrm{DSG2}}(\kappa) + \mathsf{Adv}_{\mathscr{B}}^{\mathrm{CRH}}(\kappa)$ *for* $1 \leq k \leq \nu_2$.

Proof. We establish a PPT simulator $\mathscr{B}$ who receives an instance of DSG2, $(\mathcal{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0,1\}$ and depending on the distribution of β, it either simulates Game$_{2-(k-1)-2}$ or Game$_{2-k-1}$.

Setup: Same as Lemma D.3.

Query Phase: It consists of the following queries in adaptive manner:

– $\mathsf{KeyGen}(x)$: Here all the keys are of sf-type 3 and simulation of the keys are same as the sf-type 3 keys of Lemma D.3.

– $\mathsf{Sign}(m, x, y)$: If $x \not\sim y$, it returns $\perp$. Let (m_j, x_j, y_j) be the j^{th} signature query made $\mathscr{A}$. $\mathscr{B}$ answers the signature $\boldsymbol{\delta}_{y_j}$ as follows:

- If $j > k$, it is normal signature. $\mathscr{B}$ can answer the queries of $\mathscr{A}$ as the $\mathcal{MSK}$ is known to him.
- If $j < k$, it is sf-type 2 signature. It first computes the normal signature $\boldsymbol{\delta}_{y_j}$, picks $\iota'_j \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and then returns
$$\tilde{\boldsymbol{\delta}}_{y_j} := \boldsymbol{\delta}_{y_j}.(W_2W_3)^{(0,\ \iota'_j,\ 0,\ldots,0)}.$$
If $W_2W_3 = g_2^{w_2}g_3^{w_3}$, then $\mathscr{B}$ implicitly sets $\iota_j := w_2\iota'_j$. So, $\tilde{\boldsymbol{\delta}}_{y_j}$ is properly distributed sf-type 2 signature.
- If $j = k$, it is either normal signature or sf-type 1 signature. It runs $(\boldsymbol{k}_{x_k}, m_2) \longleftarrow \mathsf{Enc1}(x_k, N)$ and $\mathsf{Pair}(x_k, y_k) \longrightarrow \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1}$. It picks $\boldsymbol{v}_{\mathsf{sp}} \xleftarrow{\mathrm{U}} \mathbf{V}^\perp$ 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 given below
$$\boldsymbol{\delta}_{y_k} := g^{(0, \boldsymbol{k}_{x_k}\boldsymbol{E})}.g^{\boldsymbol{v}_{\mathsf{sp}}}.T_\beta^{(-1,\ 0,\ \ldots,0)}.T_\beta^{(0,\ \theta_1\hbar_k+\theta_2,\ \ldots,0)}.\boldsymbol{R}_3$$
Let $g^\tau := T_\beta\big|_{\mathbb{G}_{p_1}}$ and for $\beta = 1$, $g_2^{t_2} := T_\beta\big|_{\mathbb{G}_{p_2}}$. Then, the $\mathbb{G}_{p_1}$ component of $\boldsymbol{\delta}_{y_k}$ can be written as $g^{\boldsymbol{u}+\boldsymbol{v}_{\mathsf{sp}}}$, where $\boldsymbol{u} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_{x_k}\boldsymbol{E})$ 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 $g_2^{\hat{\boldsymbol{u}}}$ where $\mathscr{B}$ implicitly sets $b :\equiv -t_2 \mod p_2$ and $\iota :\equiv t_2(\theta_1\hbar_k + \theta_2) \mod p_2$. Since $\theta_1\hbar_k + \theta_2 \mod p_1$ are independent from $\theta_1\hbar_k + \theta_2 \mod p_2$ by CRT, therefore $\boldsymbol{\delta}_{y_k}$ is 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 $\hbar^* := H(m^*, y^*)$. It runs $(\boldsymbol{c}_{y^*}, \omega_2) \longleftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$ and sets $\boldsymbol{c}_{y^*}^{\mathsf{M}} := (c_0^*, \boldsymbol{c}_{y^*})$. It picks $\boldsymbol{s}' := (s', s'_1, \ldots, s'_{\omega_2}) \xleftarrow{\mathrm{U}} \mathbb{Z}_N^{\omega_2+1}$. Finally, it computes a vText as $\mathcal{V} := (\mathcal{V}_{\mathsf{INT}} := e(g^\alpha, Z_1Z_2)^{s'}, \boldsymbol{\mathcal{V}}_{y^*} := (Z_1Z_2)^{\boldsymbol{c}_{y^*}^{\mathsf{M}}(\boldsymbol{s}', \boldsymbol{h}_{\mathsf{M}})})$. $\mathscr{B}$ returns 1 if $e(\boldsymbol{\delta}_{y^*}, \boldsymbol{\mathcal{V}}_{y^*}) = \mathcal{V}_{\mathsf{INT}}$ else 0.

Analysis: Now, we mainly concentrate on the joint distribution of k^{th} signature and vText as there may be a correlation between them. More precisely, we observe the distributional relation between $c_0^*(\hat{\boldsymbol{s}}, \hat{\boldsymbol{\theta}}) := \hat{s}(\hat{\theta}_1\hbar^* + \hat{\theta}_2) :\equiv \tilde{s}(\theta_1\hbar^* + \theta_2) \mod p_2$ and $c_{y^*,1}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}) := \hat{s} :\equiv \tilde{s} \mod p_2$ with $\tilde{s} :\equiv z_1 s'$ involved in $\boldsymbol{c}_{y^*}^{\mathsf{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}})$ of vText. Unfortunately, a similar kind of relation is found in $\hat{\boldsymbol{u}}$, viz., between $b :\equiv -t_2 \mod p_2$ and $\iota :\equiv t_2(\theta_1\hbar_j + \theta_2) \mod p_2$. But that correlation does not hamper our life: since H has collision resistant property and $(m_j, y_j) \neq (m^*, y^*)$, we have $\hbar_j \neq \hbar^*$ and hence, $\theta_1\hbar_j + \theta_2 \mod p_2$ and $\theta_1\hbar^* + \theta_2 \mod p_2$ are independently[10] and uniformly distributed over $\mathbb{Z}_{p_2}$. Therefore, $(\tilde{s}, \tilde{s}(\theta_1\hbar^* + \theta_2)) \mod p_2$ is uncorrelated from (b, ι). All together, we have the stuffs simulated by $\mathscr{B}$ are identical to that of $\text{Game}_{2-(k-1)-2}$ if $\beta = 0$ else Game_{2-k-1}. □

Lemma D.7. Game_{2-k-1} *and* Game_{2-k-2} *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}_{\mathscr{A},\text{PS}}^{2-k-1}(\kappa) - \mathsf{Adv}_{\mathscr{A},\text{PS}}^{2-k-2}(\kappa)| \leq \mathsf{Adv}_{\mathscr{B}}^{\text{DSG2}}(\kappa)$ *for* $1 \leq k \leq \nu_2$.

Proof. We establish a PPT simulator $\mathscr{B}$ who receives an instance of DSG2, $(\mathscr{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\text{U}} \{0,1\}$ and depending on the distribution of β, it either simulates Game_{2-k-1} or Game_{2-k-2}. The simulation is almost similar to Lemma D.6 except the 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 of either sf-type 1 or sf-type 2. It runs $(\boldsymbol{k}_{x_k}, m_2) \longleftarrow \mathsf{Enc1}(x_k, N)$ and $\mathsf{Pair}(x_k, y_k) \longrightarrow \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1}$. It picks $\iota_k' \xleftarrow{\text{U}} \mathbb{Z}_N$, $\boldsymbol{v}_{\mathsf{sp}} \xleftarrow{\text{U}} \mathbf{V}^\perp$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{\omega_1+1}$. It computes $\hbar_k := H(m_k, y_k)$ and then returns the signature as given below

$$\boldsymbol{\delta}_{y_k} := g^{(0, \boldsymbol{k}_{x_k}\boldsymbol{E})}.g^{\boldsymbol{v}_{\mathsf{sp}}}.T_\beta^{(-1,\ 0,\ \ldots,0)}.T_\beta^{(0,\ \theta_1\hbar_k+\theta_2,\ \ldots,0)}.(W_2W_3)^{(0,\ \iota_k',\ 0,\ldots,0)}.\boldsymbol{R}_3$$

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 $g^{\boldsymbol{u}+\boldsymbol{v}_{\mathsf{sp}}}$, where $\boldsymbol{u} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_{x_k}\boldsymbol{E})$ 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 $g_2^{\hat{\boldsymbol{u}}}$, with $\hat{\boldsymbol{u}} := (b, \iota, 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1+1}$ where $\mathscr{B}$ implicitly sets $b :\equiv -t_2 \mod p_2$ (resp. $b :\equiv 0 \mod p_2$) and $\iota :\equiv t_2(\theta_1\hbar_k + \theta_2) + w_2\iota_k' \mod p_2$ (resp. $\iota :\equiv w_2\iota_k' \mod p_2$). Therefore, $\boldsymbol{\delta}_{y_k}$ is perfectly distributed sf-type 1 (resp. sf-type 2) signature if $\beta = 1$ (resp. $\beta = 0$). □

Lemma D.8. $\text{Game}_{2-\nu_2-2}$ *and* Game_{Final} *are indistinguishable under the DSG3 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}_{\mathscr{A},\text{PS}}^{2-\nu_2-2}(\kappa) - \mathsf{Adv}_{\mathscr{A},\text{PS}}^{Final}(\kappa)| \leq \mathsf{Adv}_{\mathscr{B}}^{\text{DSG3}}(\kappa)$.

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

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

Query Phase: It consists of the following queries in adaptive manner:

[10]To show independent, we require that $\hbar_j - \hbar^* \not\equiv 0 \mod p_2$. From $\hbar_j - \hbar^* \not\equiv 0 \mod N$, we have $\hbar_j - \hbar^* \not\equiv 0 \mod p$ for at least one p such that $p \in \{p_1, p_2, p_3\}$. One can show that $\hbar_j - \hbar^* \not\equiv 0 \mod p$ for all p with $p \in \{p_1, p_2, p_3\}$ assuming factorization problem is hard. However, if $\hbar_j - \hbar^* \equiv 0 \mod p_2$ we can find a factor F of N with $p_2|F$ and which leads to break the DSG2 assumption, a contradiction.

– $\mathsf{KeyGen}(x)$: It is sf-type 3 key. It runs $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x)$. Then it picks $\boldsymbol{r} \xleftarrow{\text{U}} \mathbb{Z}_N^{m_2}$, $\hat{\alpha}' \xleftarrow{\text{U}} \mathbb{Z}_N$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{m_1}$. Finally it returns

$$\mathcal{SK}_x := (g^\alpha Y_2)^{\boldsymbol{k}_x(1,\boldsymbol{0},\boldsymbol{0})}.g^{\boldsymbol{k}_x(0,\boldsymbol{r},\boldsymbol{h})}.g_2^{\boldsymbol{k}_x(\hat{\alpha}',\boldsymbol{0},\boldsymbol{0})}.\boldsymbol{R}_3$$

If $Y_2 = g_2^{y_2}$, $\mathscr{B}$ implicitly sets $\hat{\alpha} := y_2 + \hat{\alpha}' \mod p_2$ and so, $\mathcal{SK}_x$ is a perfectly distributed sf-type 3 key.

– $\mathsf{Sign}(m, x, y)$: If $x \not\sim y$, it returns $\perp$. It is sf-type 2 signature. $\mathscr{B}$ first creates sf-type 3 key $\mathcal{SK}_x$ and then using $\mathcal{SK}_x$, it can compute the sf-type 2 signature as described in **Remark 3.11**.

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

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

Analysis: $\mathsf{perfectness}$: All the components simulated above are perfectly distributed as required. Therefore, all the stuffs simulated by $\mathscr{B}$ are identical to that of Game$_{2-\nu_2-2}$ if $\beta = 0$ else Game$_{Final}$. ☐

E Lemmas used in Theorem 5.2 for Predicate Encryption

Lemma E.1. Game$_{Real}$ *and* Game$_{Res}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}_{\mathscr{A},\text{PE}}^{\text{Real}}(\kappa) - \mathsf{Adv}_{\mathscr{A},\text{PE}}^{\text{Res}}(\kappa)| \le \mathsf{Adv}_{\mathscr{B}}^{\text{DSG2}}(\kappa)$.

Proof. It can be followed from the Lemma 27 of [2] or Lemma D.1 in this paper. ☐

Lemma E.2. Game$_{Res}$ *and* Game$_0$ *are indistinguishable under the DSG1 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}_{\mathscr{A},\text{PE}}^{\text{Res}}(\kappa) - \mathsf{Adv}_{\mathscr{A},\text{PE}}^{0}(\kappa)| \le \mathsf{Adv}_{\mathscr{B}}^{\text{DSG1}}(\kappa)$.

Proof. The proof is similar to the Lemma 28 of [2] and Lemma D.2 in this paper. ☐

Lemma E.3. Game$_{1-(k-1)-3}$ *and* Game$_{1-k-1}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}_{\mathscr{A},\text{PE}}^{1-(k-1)-3}(\kappa) - \mathsf{Adv}_{\mathscr{A},\text{PE}}^{1-k-1}(\kappa)| \le \mathsf{Adv}_{\mathscr{B}}^{\text{DSG2}}(\kappa)$ *for* $1 \le k \le q_1$.

Proof. For proof, refer to the proof of the Lemma 29 of [2] and Lemma D.3 in this paper. ☐

Lemma E.4. Game$_{1-k-1}$ *and* Game$_{1-k-2}$ *are indistinguishable under* CMH *security of the primitive pair encoding scheme,* P*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}_{\mathscr{A},\text{PE}}^{1-k-1}(\kappa) - \mathsf{Adv}_{\mathscr{A},\text{PE}}^{1-k-2}(\kappa)| \le \mathsf{Adv}_{\mathscr{B}}^{\text{P}-\text{CMH}}(\kappa)$ *for* $1 \le k \le q_1$.

Proof. Following the proof of Lemma 30 of [2] and Lemma D.4 in this paper, it can be proven. Note that condition (1) in **Conditions 3.2** will be used here. ☐

Lemma E.5. Game$_{1-k-2}$ *and* Game$_{1-k-3}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-k-2}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{1-k-3}_{\mathscr{A},\mathrm{PE}}(\kappa)| \le \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \le k \le q_1$*.*

Proof. The proof is similar to that of Lemma 31 of [2] and Lemma D.5 in this paper. □

Lemma E.6. Game$_{1-q_1-3}$ *and* Game$_{1-(q_1+1)-1}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-q_1-3}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{1-(q_1+1)-1}_{\mathscr{A},\mathrm{PE}}(\kappa)| \le \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$*.*

Proof. For proof, we refer to Lemma 32 of [2]. □

Lemma E.7. Game$_{1-(q_1+1)-1}$ *and* Game$_{1-(q_1+1)-2}$ *are indistinguishable under* SMH *security of of the primitive pair encoding scheme,* P*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-(q_1+1)-1}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{1-(q_1+1)-2}_{\mathscr{A},\mathrm{PE}}(\kappa)| \le \mathsf{Adv}^{\mathrm{P-SMH}}_{\mathscr{B}}(\kappa)$*.*

Proof. The proof is similar to Lemma 33 of [2]. Note that condition (1) in **Conditions 3.2** is applied here. □

Lemma E.8. Game$_{1-(q_1+1)-2}$ *and* Game$_{1-(q_1+1)-3}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-(q_1+1)-2}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{1-(q_1+1)-3}_{\mathscr{A},\mathrm{PE}}(\kappa)| \le \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$*.*

Proof. The proof can be done in similar manner as in Lemma 34 of [2]. □

Lemma E.9. Game$_{2-(k-1)-2}$ *and* Game$_{2-k-1}$ *are indistinguishable under the DSG2 assumption and collision resistant property of* H*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{2-(k-1)-2}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{2-k-1}_{\mathscr{A},\mathrm{PE}}(\kappa)| \le \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa)$ *for* $1 \le k \le \nu$*.*

Proof. We establish a PPT simulator $\mathscr{B}$ who receives an instance of DSG2, $(\mathcal{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\mathrm{U}} \{0,1\}$ and depending on the distribution of β, it either simulates Game$_{2-(k-1)-2}$ or Game$_{2-k-1}$.

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

Query Phase-1: It consists of the following queries in adaptive manner:

– $\mathsf{KeyGen}(x)$: Here all the keys are of sf-type 3 and simulation of the keys are same as the sf-type 3 keys of Lemma E.3.

– $\mathsf{Decrypt}(\mathsf{CT}, x)$: Let (CT_j, x_j) be the j^{th} decryption query made $\mathscr{A}$. $\mathscr{B}$ first constructs the alt-key $\mathcal{SK}^{\mathsf{M}}_{x_j}$ as shown below and then answers to $\mathscr{A}$ by running $\mathsf{AltDecrypt}$ algorithm :

- If $j > k$, it is normal alt-key. $\mathscr{B}$ can compute the key as the $\mathcal{MSK}$ is known to him.

- If $j < k$, it is sf-type 2 alt-key. It first computes the normal alt-key $\mathcal{SK}_{x_j}^{\mathsf{M}}$, picks $\iota'_j \xleftarrow{\text{U}} \mathbb{Z}_N$ and then creates the sf-type 2 alt-key as follows:
$$\tilde{\mathcal{SK}}_{x_j}^{\mathsf{M}} := \mathcal{SK}_{x_j}^{\mathsf{M}}.(W_2W_3)^{(0,\ \iota'_j,\ 0,\ldots,0)}.$$
If $W_2W_3 = g_2^{w_2}g_3^{w_3}$, then $\mathscr{B}$ implicitly sets $\iota_j := w_2\iota'_j$. So, $\tilde{\mathcal{SK}}_{x_j}^{\mathsf{M}}$ is properly distributed sf-type 2 alt-key.
- If $j = k$, it is either normal or sf-type 1 alt-key. It runs $(\boldsymbol{k}_{x_k}, m_2) \longleftarrow \mathsf{Enc1}(x_k, N)$ and $\mathsf{Pair}(x_k, y_k) \longrightarrow \boldsymbol{E} \in \mathbb{Z}_N^{m_1\times\omega_1}$. It picks $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{\omega_1+1}$. It computes the alt-key as given below :
$$\mathcal{SK}_{x_k}^{\mathsf{M}} := g^{(0,\boldsymbol{k}_{x_k}\boldsymbol{E})}.T_\beta^{(-1,\ 0,\ \ldots,0)}.T_\beta^{(0,\ \theta_1\hbar_k+\theta_2,\ \ldots,0)}.\boldsymbol{R}_3; \quad \text{where } \hbar_k := H(C_{\mathsf{cpa}}^{(k)})$$
Let $g^\tau := T_\beta|_{\mathbb{G}_{p_1}}$ and for $\beta = 1$, $g_2^{t_2} := T_\beta|_{\mathbb{G}_{p_1}}$. Then, it sets $g^{\boldsymbol{u}} := \mathcal{SK}_{x_k}^{\mathsf{M}}|_{\mathbb{G}_{p_1}}$, where $\boldsymbol{u} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_{x_k}\boldsymbol{E})$ and $\boldsymbol{\psi} := (\tau(\theta_1\hbar_k + \theta_2), 0, \ldots, 0)$. If $\beta = 1$, it sets $g_2^{\hat{\boldsymbol{u}}} := \mathcal{SK}_{x_k}^{\mathsf{M}}|_{\mathbb{G}_{p_2}}$ with $\hat{\boldsymbol{u}} := (b, \iota, 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1+1}$, where $\mathscr{B}$ implicitly sets $b :\equiv -t_2 \mod p_2$ and $\iota :\equiv t_2(\theta_1\hbar_k + \theta_2) \mod p_2$.

Challenge Phase: $\mathscr{A}$ provides two equal length messages m_0, m_1 and the challenge index y^* to $\mathscr{B}$. Then, $\mathscr{B}$ picks $b \xleftarrow{\text{U}} \{0,1\}$. It runs $(\boldsymbol{c}_{y^*}, \omega_2) \longleftarrow \mathsf{Enc2}(y^*, N)$ with $|\boldsymbol{c}_{y^*}| = \omega_1$. It picks $\boldsymbol{s'} := (s', s'_1, \ldots, s'_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. It first computes $C_{\mathsf{cpa}}^* := (y^*, \boldsymbol{C}_{y^*} := (Z_1Z_2)^{\boldsymbol{c}_{y^*}(\boldsymbol{s'},\boldsymbol{h})}, C_{\mathsf{INT}} := m^*.e(g^\alpha, Z_1Z_2)^{s'})$ and then computes $\hbar^* := H(C_{\mathsf{cpa}}^*)$. Finally, it returns the challenge ciphertext $\mathsf{CT}^* := (y^*, \boldsymbol{C}_{y^*}^{\mathsf{M}} := (Z_1Z_2)^{\boldsymbol{c}_{y^*}^{\mathsf{M}}(\boldsymbol{s'},\boldsymbol{h}_{\mathsf{M}})}, C_{\mathsf{INT}} := m_b.e(g^\alpha, Z_1Z_2)^{s'})$. Recall that $\mathsf{CT}^* = (C_{\mathsf{cpa}}^*, C_0^*)$ with $C_0^* := (Z_1Z_2)^{s'(\theta_1\hbar^*+\theta_2)}$. If $Z_1Z_2 = g^{z_1}g_2^{z_2}$, it implicitly sets $\boldsymbol{s} := z_1\boldsymbol{s'} \mod p_1$ and $\hat{\boldsymbol{s}} := z_2\boldsymbol{s'} \mod p_2$. Since, $\boldsymbol{s'} \mod p_1$ is independent from $\boldsymbol{s'} \mod p_2$, therefore CT^* is perfectly distributed sf-type 1 challenge ciphertext

Query Phase-2: Similar to phase-1 except, suppose the k^{th} decryption query is made in Phase-2, then $\mathscr{B}$ solves the given instance of DSG2 assumption (described later) and aborts if $\mathsf{CT}^* \neq \mathsf{CT}_k$ and $\hbar^* = \hbar^{(k)}$.

Guess: $\mathscr{A}$ sends a guess b' to $\mathscr{B}$. If $b' = b$ then $\mathscr{B}$ returns 1 else 0.

Analysis: By the natural restriction of the security game, $\mathscr{A}$ is allowed to decryption query CT_k if $\mathsf{CT}^* \neq \mathsf{CT}_k$. On the based of the analysis part in the proof of the Theorem D.6, $\hbar^* = \hbar^{(k)}$ could hamper the joint distribution, but we show that if this happens, then $\mathscr{B}$ can solve the DSG2 assumption: We start with
$$\boxed{\mathsf{CT}^* \neq \mathsf{CT}_k \ \text{ and } \ \hbar^* = \hbar^{(k)}} \tag{6}$$
Since, H is a collision resistant hash function, from the equation (6), we have
$$\boxed{C_0^* \neq C_0^{(k)} \ \text{ and } \ C_{\mathsf{cpa}}^* = C_{\mathsf{cpa}}^{(k)}} \tag{7}$$
From the definition of $\mathsf{AltDecrypt}$ and $C_1^* = C_1^{(k)}$, we have the following equations:
$$\boxed{C_0^{(k)}|_{\mathbb{G}_{p_3}} = \Theta \ \text{ and } e(g, C_0^{(k)}) = e(g^{\theta_1\hbar^{(k)}+\theta_2}, C_1^{(k)})} \tag{8}$$
From the challenge ciphertext, we have
$$\boxed{C_0^*|_{\mathbb{G}_{p_3}} = \Theta \ \text{ and } e(g, C_0^*) = e(g^{\theta_1\hbar^*+\theta_2}, C_1^*)} \tag{9}$$

Using the 2nd part of the equations (6), (7), (8) and (9), we have $e(g, C_0^*) = e(g, C_0^{(k)})$ which in turn implies that

$$\boxed{C_0^*\big|_{\mathbb{G}_{p_1}} = C_0^{(k)}\big|_{\mathbb{G}_{p_1}}} \tag{10}$$

Since $C_0^{(k)}\big|_{\mathbb{G}_{p_3}} = \Theta$, $C_0^*\big|_{\mathbb{G}_{p_3}} = \Theta$, using equation (10), we must have $Y_2 := (C_0^*)^{-1}.C_0^{(k)} \in \mathbb{G}_{p_2}$. Since $C_0^* \neq C_0^{(k)}$, we have $Y_2 \neq \Theta$. Therefore, $\mathscr{B}$ can break the given instance of DSG2 assumption using Y_2. $\square$

Lemma E.10. Game$_{2-k-1}$ *and* Game$_{2-k-2}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{2-k-1}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{2-k-2}_{\mathscr{A},\mathrm{PE}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq \nu$.

Proof. We establish a PPT simulator $\mathscr{B}$ who receives an instance of DSG2, $(\mathcal{J}, g, Z_1Z_2, W_2W_3, Z_3, T_\beta)$ with $\beta \xleftarrow{\text{U}} \{0,1\}$ and depending on the distribution of β, it either simulates Game$_{2-k-1}$ or Game$_{2-k-2}$. The simulation is almost similar to Lemma E.9 except the answering k^{th} decryption query. Note that in this case, we do not require the collision resistant property of H. We illustrate here only the k^{th} alt-key : The k^{th} alt-key is of either sf-type 1 or sf-type 2. It runs $(\boldsymbol{k}_{x_k}, m_2) \longleftarrow \mathsf{Encl}(x_k, N)$ and $\mathsf{Pair}(x_k, y_k) \longrightarrow \boldsymbol{E} \in \mathbb{Z}_N^{m_1 \times \omega_1}$. It picks $\iota'_k \xleftarrow{\text{U}} \mathbb{Z}_N$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{\omega_1+1}$. It computes the alt-key as given below

$$\mathcal{SK}^{\mathsf{M}}_{x_k} := g^{(0, \boldsymbol{k}_{x_k}\boldsymbol{E})}.T_\beta^{(-1,\ 0,\ \ldots, 0)}.T_\beta^{(0,\ \theta_1\hbar_k+\theta_2,\ \ldots, 0)}.(W_2W_3)^{(0,\ \iota'_k,\ 0,\ldots,0)}.\boldsymbol{R}_3; \quad \text{where } \hbar_k := H(C^{(k)}_{\mathsf{cpa}})$$

Let $W_2W_3 = g_2^{w_2}g_3^{w_3}$. Let $g^\tau := T_\beta\big|_{\mathbb{G}_{p_1}}$ and for $\beta = 1$, $g_2^{t_2} := T_\beta\big|_{\mathbb{G}_{p_1}}$. Then, it sets $g^{\boldsymbol{u}} = \mathcal{SK}^{\mathsf{M}}_{x_k}\big|_{\mathbb{G}_{p_1}}$, where $\boldsymbol{u} := (-\tau, \boldsymbol{\psi} + \boldsymbol{k}_{x_k}\boldsymbol{E})$ and $\boldsymbol{\psi} := (\tau(\theta_1\hbar_k + \theta_2), 0, \ldots, 0)$. If $\beta = 1$ (resp. $\beta = 0$), it sets $g_2^{\hat{\boldsymbol{u}}} := \mathcal{SK}^{\mathsf{M}}_{x_k}\big|_{\mathbb{G}_{p_2}}$ with $\hat{\boldsymbol{u}} := (b, \iota, 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1+1}$, where $\mathscr{B}$ implicitly sets $b :\equiv -t_2 \mod p_2$ (resp. $b :\equiv 0 \mod p_2$) and $\iota :\equiv t_2(\theta_1\hbar_k + \theta_2) + w_2\iota'_k \mod p_2$ (resp. $\iota :\equiv w_2\iota'_k \mod p_2$). Therefore, $\mathcal{SK}^{\mathsf{M}}_{x_k}$ is perfectly distributed sf-type 1 (resp. sf-type 2) alt-key if $\beta = 1$ (resp. $\beta = 0$). $\square$

Lemma E.11. Game$_{2-\nu}$ *and* Game$_{Final}$ *are indistinguishable under the DSG3 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{2-\nu}_{\mathscr{A},\mathrm{PE}}(\kappa) - \mathsf{Adv}^{Final}_{\mathscr{A},\mathrm{PE}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG3}}_{\mathscr{B}}(\kappa)$.

Proof. For proof, we refer to Lemma 35 of [2] and Lemma D.8 in this paper. Note that condition (3) in **Conditions 3.2** is applied here. $\square$

F Security of the Proposed Predicate Signcryption 6

Although $\mathsf{Signcrypt}$ and $\mathsf{Unsigncrypt}$ run almost in black-box way, neither the confidentiality nor unforgeability of the proposed predicate signcryption are proven as black-box proof of PE in sec.5.1 and PS in sec. 3.6 respectively. The reason behind is that the adversary is given access to signcryption oracle in the adaptive-predicates IND-CCA (resp. UF-CMA) model of predicate signcryption scheme which could not be answered using black-box proof of PE in sec.5.1 (resp. PS in sec. 3.6).

F.1 Semi-Functional Stuffs for Confidentiality and Unforgeability

To obtain the adaptive-predicates confidentiality and unforgeability, we use the dual system proof technique [33], but its signcryption version [31]. We consider two kinds of signcryptions, the replied signcryptions (queried through the signcryption oracle) and challenge signcryption. The former has three forms, N (normal), sf-type I and sf-type II, whereas the later is of five forms, N, sf-type 1, sf-type 2, sf-type 3 and sf-type 4. For simplicity, we ignore the one-time signature and write signcryption := (signature (δ_{y_s}), ciphertext (CT)). We also consider a new stuff, called verification text key (in short $\mathsf{vTextKey}$) which is composed of alt-key and vText, i.e., better to write vTextKey := (alt-key, vText). This vTextKey will be used to unsigncrypt the signcryption using a new algorithm, $\mathsf{AltUnsigncrypt}$. Similar to the forms of signcryption, we consider two kinds of vTextKeys, one is used to answer the unsigncryption queries and other to unsigncrypt the forgery signcryption. The former has three forms, N, sf-type I and sf-type II, whereas the later is of five forms, N, sf-type 1, sf-type 2, sf-type 3 and sf-type 4. Since, the signcryption (resp. unsigncryption) is obtained by running the two routines, Sign (resp. Ver)and $\mathsf{Encrypt}$ (resp. $\mathsf{Decrypt}$) almost in black-box manner, we described the different forms of signcryptions (resp. vTextKeys) through already defined different forms of signatures (resp. alt-keys) and ciphertexts (resp. vTexts). For this purpose, we define a (type converter) function $f_{\mathsf{convrt}} : \{\mathsf{N}, \mathsf{I}, \mathsf{II}, 1, 2, 3, 4\} \rightarrow \{(i, j) \mid i, j \in \{\mathsf{N}, 1, 2\}\}$, which takes the type of a signcryption (resp. vTextKey) as an input and outputs a pair (i, j) of form of signature (resp. alt-key) and form of ciphertext (resp. vText). The function f_{convrt} is completely defined by the image as $f_{\mathsf{convrt}}(\mathsf{N}) := (\mathsf{N}, \mathsf{N})$, $f_{\mathsf{convrt}}(\mathsf{I}) := (1, \mathsf{N})$, $f_{\mathsf{convrt}}(\mathsf{II}) := (2, \mathsf{N})$, $f_{\mathsf{convrt}}(1) := (\mathsf{N}, 1)$, $f_{\mathsf{convrt}}(2) := (1, 1)$, $f_{\mathsf{convrt}}(3) := (2, 1)$ and $f_{\mathsf{convrt}}(4) := (2, 2)$. From the description of f_{convrt}, we have the form of ciphertexts (resp. vTexts) in the signcryptions (resp. vTextKeys) of sf-type I and sf-type II are always normal.

– $\mathsf{SFSetup}(1^\kappa, \boldsymbol{j})$: It runs $(\mathcal{PP}, \mathcal{MSK}) \longleftarrow \mathsf{Setup}(1^\kappa, \boldsymbol{j})$ and in addition it returns semi-functional parameters, $g_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2}$, $\hat{\theta}_1, \hat{\theta}_2 \xleftarrow{\text{U}} \mathbb{Z}_N$ and $\hat{\boldsymbol{h}} \xleftarrow{\text{U}} \mathbb{Z}_N^n$. We set $\hat{\boldsymbol{h}}_{\mathsf{M}} := (\hat{\theta}_1, \hat{\theta}_2, \hat{\boldsymbol{h}})$.

– $\mathsf{SFKeyGen}(\mathcal{PP}, \mathcal{MSK}, x, g_2, \mathsf{type}, \hat{\boldsymbol{h}})$: It runs $(\boldsymbol{k}_x, m_2) \longleftarrow \mathsf{Enc1}(x, N)$ with $|\boldsymbol{k}_x| = m_1$. It chooses $\hat{\alpha} \xleftarrow{\text{U}} \mathbb{Z}_N$, $\boldsymbol{r}, \hat{\boldsymbol{r}} \xleftarrow{\text{U}} \mathbb{Z}_N^{m_2}$ and $\boldsymbol{R}_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}^{m_1}$. It outputs the semi-functional key $\mathcal{SK}_x := (x, \boldsymbol{K}_x)$, where $\boldsymbol{K}_x$ is given by

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

– $\mathsf{SFEncrypt}(\mathcal{PP}, m, y, g_2, \mathsf{type}, \hat{\boldsymbol{h}}_{\mathsf{M}})$: It runs $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$ and picks $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}), \hat{\boldsymbol{s}} := (\hat{s}, \hat{s}_1, \ldots, \hat{s}_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. It computes $\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) := (c_0(s, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{G}^{\omega_1+1}$ and $\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}}) := (c_0(\hat{s}, \hat{\boldsymbol{\theta}}), \boldsymbol{c}_y(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}})) \in \mathbb{G}^{\omega_1+1}$, where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, $\hat{\boldsymbol{\theta}} := (\hat{\theta}_1, \hat{\theta}_2, \hbar)$, $\hbar := H(C_{\mathsf{cpa}})$, $C_{\mathsf{cpa}} := (y, \boldsymbol{C}_y := g^{\boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})}, C_{\mathsf{INT}} := m.g_T^{\alpha s})$, $c_0(s, \boldsymbol{\theta}) := s(\theta_1 \hbar + \theta_2)$ and $c_0(\hat{s}, \hat{\boldsymbol{\theta}}) := \hat{s}(\hat{\theta}_1 \hbar + \hat{\theta}_2)$. It returns the semi-function ciphertext as $\mathsf{CT} := (y, \boldsymbol{C}_y^{\mathsf{M}} := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}})}, C_{\mathsf{INT}})$.

$$\mathsf{CT} := \begin{cases} (y, \boldsymbol{C}_y^{\mathsf{M}} := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}})}.g_2^{\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}})}, C_{\mathsf{INT}} := m.g_T^{\alpha s} & \text{if } \mathsf{type}= 1 \\ (y, \boldsymbol{C}_y^{\mathsf{M}} := g^{\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}})}.g_2^{\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}})}, C_{\mathsf{INT}} := m.g_t;\ g_t \xleftarrow{\text{U}} \mathbb{G}_T & \text{if } \mathsf{type}= 2 \end{cases}$$

– $\mathsf{SFSign}(\mathcal{PP}, m, \mathcal{SK}_x, y, g_2, \mathsf{type})$: If $x \not\sim y$, returns $\perp$. It runs $\boldsymbol{\delta}_y \longleftarrow \mathsf{Sign}(\mathcal{PP}, m, \mathcal{SK}_x, y)$. Note that $\boldsymbol{\delta}_y = g^{\boldsymbol{u}+\boldsymbol{v}_{\mathsf{sp}}}.\boldsymbol{R}_3$ with $\boldsymbol{R}_3 \in \mathbb{G}_{p_3}^{\omega_1+1}$. It picks $b, \iota \xleftarrow{\text{U}} \mathbb{Z}_N$ and returns the semi-functional signature

$\boldsymbol{\delta}_y.g_2^{\hat{\boldsymbol{u}}}$, where $\hat{\boldsymbol{u}} \in \mathbb{Z}_N^{\omega_1+1}$ is given by

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

– $\mathsf{SFSigncrypt}(\mathcal{PP}, m, \mathcal{SK}_x, y_s, y_e, g_2, \mathsf{type}, \hat{\boldsymbol{h}}_{\mathsf{M}})$: If $x \not\sim y$, returns $\perp$. It first runs $(\mathsf{com}, \mathsf{decom}) \longleftarrow \mathsf{Commit}(m)$ and $(\mathsf{vk}, \mathsf{signk}) \longleftarrow \mathsf{Gen}(1^\kappa)$. Let $(i, j) := f_{\mathsf{convrt}}(\mathsf{type})$. It runs $\boldsymbol{\delta}_{y_s} \longleftarrow \mathsf{SFSign}(\mathcal{PP}, \mathsf{vk}, \mathcal{SK}_x, y_s, g_2, i)$ and $\mathsf{CT} \longleftarrow \mathsf{SFEncrypt}(\mathcal{PP}, \mathsf{decom}, y, g_2, j, \hat{\boldsymbol{h}}_{\mathsf{M}})$, where $\hbar_e := H(0, \mathsf{com}, \delta_{y_s}, \mathsf{vk}, C_{\mathsf{cpa}})$ and $C_0 := g^{s(\theta_1 \hbar_e + \theta_2)}$. It computes the one-time signature $\delta_o := \mathsf{OTS.Sign}(C_0||y_s, \mathsf{signk})$. It returns the semi-functional signcryption $\mathsf{U} := (\mathsf{com}, \delta := (\delta_{y_s}, \delta_o, \mathsf{vk}), \mathsf{CT} := (C_{\mathsf{cpa}}, C_0))$

– $\mathsf{SFAltKeyGen}(\mathcal{PP}, \mathcal{MSK}, \mathsf{CT}, x, g_2, \mathsf{type})$: It phrases CT as (C_{cpa}, C_0), computes $\hbar := H(C_{\mathsf{cpa}})$ and picks $\tau \xleftarrow{\text{U}} \mathbb{Z}_N$, $R_0 \xleftarrow{\text{U}} \mathbb{G}_{p_3}$. It first generates the normal key, $\mathcal{SK}_x := [x,\ \boldsymbol{K}_x := g^{\boldsymbol{k}_x(\alpha, \boldsymbol{r}, \boldsymbol{h})}.\boldsymbol{R}_3]$. Then, it creates the alt-key $\mathcal{SK}_x^{\mathsf{M}} := (K_0, \boldsymbol{\Psi}.\boldsymbol{K}_x^{\boldsymbol{E}}) \in \mathbb{G}^{\omega_1+1}$, where $K_0 := g^{-\tau} R_0$, $\boldsymbol{\Psi} := g^{\boldsymbol{\psi}}$ with $\boldsymbol{\psi} := (\tau(\theta_1 \hbar + \theta_2), 0, \ldots, 0) \in \mathbb{Z}_N^{\omega_1}$ and $\boldsymbol{E} \leftarrow \mathsf{Pair}(x, y)$. It picks $b, \iota \xleftarrow{\text{U}} \mathbb{Z}_N$ and returns the semi-functional alt-key $\mathcal{SK}_x^{\mathsf{M}}.g_2^{\hat{\boldsymbol{u}}}$, where $\hat{\boldsymbol{u}} \in \mathbb{Z}_N^{\omega_1+1}$ is given by

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

– $\mathsf{SFVText}(\mathcal{PP}, m, y, g_2, \mathsf{type}, \hat{\boldsymbol{h}}_{\mathsf{M}})$: It runs $(\boldsymbol{c}_y, \omega_2) \longleftarrow \mathsf{Enc2}(y, N)$ and picks $\boldsymbol{s} := (s, s_1, \ldots, s_{\omega_2}), \hat{\boldsymbol{s}} := (\hat{s}, \hat{s}_1, \ldots, \hat{s}_{\omega_2}) \xleftarrow{\text{U}} \mathbb{Z}_N^{\omega_2+1}$. It computes $\boldsymbol{c}_y^{\mathsf{M}}(\boldsymbol{s}, \boldsymbol{h}_{\mathsf{M}}) := (c_0(s, \boldsymbol{\theta}), \boldsymbol{c}_y(\boldsymbol{s}, \boldsymbol{h})) \in \mathbb{G}^{\omega_1+1}$ and $\boldsymbol{c}_y^{\mathsf{M}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}}_{\mathsf{M}}) := (c_0(\hat{s}, \hat{\boldsymbol{\theta}}), \boldsymbol{c}_y(\hat{\boldsymbol{s}}, \hat{\boldsymbol{h}})) \in \mathbb{G}^{\omega_1+1}$, where $|\boldsymbol{c}_y| = \omega_1$, $\boldsymbol{\theta} := (\theta_1, \theta_2, \hbar)$, $\hat{\boldsymbol{\theta}} := (\hat{\theta}_1, \hat{\theta}_2, \hbar)$, $\hbar := H(m, y)$, $c_0(s, \boldsymbol{\theta}) := s(\theta_1 \hbar + \theta_2)$ and $c_0(\hat{s}, \hat{\boldsymbol{\theta}}) := \hat{s}(\hat{\theta}_1 \hbar + \hat{\theta}_2)$. It returns the semi-function verification text as

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

– $\mathsf{SFVTextKey}(\mathcal{PP}, \mathcal{MSK}, \mathsf{U}, x, y_s, g_2, \mathsf{type}, \hat{\boldsymbol{h}}_{\mathsf{M}})$: It runs $\mathcal{SK}_x^{\mathsf{M}} \longleftarrow \mathsf{SFAltKeyGen}(\mathcal{PP}, \mathcal{MSK}, \mathsf{CT}, x, g_2, i)$ and $\mathcal{V} \longleftarrow \mathsf{SFVText}(\mathcal{PP}, \mathsf{vk}, y_s, g_2, j, \hat{\boldsymbol{h}}_{\mathsf{M}})$, where $(i, j) = f_{\mathsf{convrt}}(\mathsf{type})$. It returns the semi-functional vTextKey, $\mathcal{VK} := (\mathcal{SK}_x^{\mathsf{M}}, \mathcal{V})$.

– $\mathsf{AltDecrypt}(\mathcal{PP}, \mathsf{CT}, \mathcal{SK}_x^{\mathsf{M}})$: This is same as $\mathsf{Decrypt}$ algorithm, but here we do not need to compute the alt-key as it is supplied. For sake of completeness: It picks $R \xleftarrow{\text{U}} \mathbb{G}_{p_3}$. If $x \not\sim y$ or $e(gR, C_0) \neq e(g^{\theta_1 \hbar + \theta_2}, C_1)$, it returns $\perp$ else $C_{\mathsf{INT}}/e(\mathcal{SK}_x^{\mathsf{M}}, \boldsymbol{C}_y^{\mathsf{M}})$.

– $\mathsf{AltVer}(\boldsymbol{\delta}_{y_s}, \boldsymbol{\mathcal{V}})$: This is same as Ver algorithm, but we do not require to compute the vText as it is supplied. Let $\boldsymbol{\mathcal{V}} = (\mathcal{V}_{\mathsf{INT}}, \boldsymbol{\mathcal{V}}_{y_s})$. It $e(\boldsymbol{\delta}_{y_s}, \boldsymbol{\mathcal{V}}_{y_s}) = \mathcal{V}_{\mathsf{INT}}$ returns 1, else 0.

– $\mathsf{AltUnsigncrypt}(\mathcal{PP}, \mathsf{U}, \mathcal{VK}, y_s)$: Let $\mathcal{VK} = (\mathcal{SK}_x^{\mathsf{M}}, \boldsymbol{\mathcal{V}})$. This is same as $\mathsf{Unsigncrypt}$ algorithm, but here we do not need to compute the alt-key $\mathcal{SK}_x^{\mathsf{M}}$ and vText $\boldsymbol{\mathcal{V}}$ respectively involved in the routines, $\mathsf{Decrypt}$ and Ver as they are supplied. In other word, it is same as $\mathsf{Unsigncrypt}$ algorithm, except the $\mathsf{Decrypt}$ and Ver are replaced by $\mathsf{AltDecrypt}$ and AltVer respectively.

We note that the stuffs of a particular form defined above may not be used in both, the proof confidentiality and unforgeability. For example, the signcryptions (resp. vTextKeys) of the forms, sf-type 1, sf-type 2, sf-type 3 and sf-type 4 are not used in the proof unforgeability (resp. confidentiality).

F.2 The Proof of Confidentiality

Theorem F.1. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has both the security,* SMH *and* CMH*, the assumptions, DSG1, DSG2 and DSG3 hold in* $\mathcal{J}$*, the one-time signature scheme,* OTS *has strong unforgeability, the commitment scheme,* $\mathcal{C}$ *has the hiding property and* H *is a collision resistant hash function, then the proposed predicate signcryption scheme,* PSC *in sec.6 for the predicate* $\sim$ *is adaptive-predicates IND-CCA secure.*

Proof. Suppose there are at most q, ν_1 and ν_2 number of key, unsigncryption and signcryption queries respectively made by an adversary $\mathscr{A}$, then the security proof consists of hybrid argument over a sequence of $3q_1 + 2(\nu_1 + \nu_2) + 10$ games, where among the q key queries, q_1 and q_2 respectively be the number of phase 1 and phase 2 key queries.

Let $\mathsf{U}^* := (\mathsf{com}^*, \delta^* := (\delta_{y_s^*}, \delta_o^*, \mathsf{vk}^*), \mathsf{CT}^* := (C^*_{\mathsf{cpa}}, C^*_0))$ denote the challenge signcryption for the data indices (y_s^*, y_e^*). Let (U, x, y_s) with $\mathsf{U} := (\mathsf{com}, \delta := (\delta_{y_s}, \delta_o, \mathsf{vk}), \mathsf{CT} := (C_{\mathsf{cpa}}, C_0))$ be any unsigncryption query. We define an event E as

$$\mathsf{E} := [(\mathsf{vk}^* = \mathsf{vk}) \wedge (\delta_o^* || C_0^* || y_s^* \neq \delta_o || C_0 || y_s)]$$

We will apply the hybrid arguments over the following games, where all the unsigncryptions queries are answered by the suitable forms of vTextKeys using the algorithm $\mathsf{AltUnsigncrypt}$:

- Game_{Real} := The original APs-IND-CCA security game.
- $\text{Game}_{\widehat{Real}}$:= Same as Game_{Real} except the challenger always returns $\perp$ on unsigncryption query if E occurs.
- Game_{Res} := This is same as $\text{Game}_{\widehat{Real}}$ except $x \not\sim_N y^*$ is replaced by $x \not\sim_{p_2} y^*$ for each key query x made by $\mathscr{A}$.
- Game_0 ($= \text{Game}_{1-0-3}$) is just like Game_{Res} except that the challenge signcryption is of sf-type 1.
- In Game_{1-k-1} (for $1 \le k \le q_1$) is same as $\text{Game}_{1-(k-1)-3}$ except the k^{th} queried key is sf-type 1.
- Game_{1-k-2} (for $1 \le k \le q_1$) is same as Game_{1-k-1} except the k^{th} queried key is sf-type 2.
- Game_{1-k-3} (for $1 \le k \le q_1$) is same as Game_{1-k-2} except the k^{th} queried key is sf-type 3.
- $\text{Game}_{1-(q_1+1)-i}$ (for $1 \le i \le 3$) is same as Game_{1-q_1-3} except the last q_2 queried keys are of sf-type i.
- In Game_{2-k-1} (for $1 \le k \le \nu_1$) is same as $\text{Game}_{2-(k-1)-2}$ except the k^{th} unsigncryption query is answered by vTextKey of the form, sf-type I. (In this sequel, we define $\text{Game}_{2-0-2} = \text{Game}_{1-(q_1+1)-3}$)
- Game_{2-k-2} (for $1 \le k \le \nu_1$) is same as Game_{2-k-1} except the k^{th} unsigncryption query is answered by vTextKey of the form, sf-type II.
- In Game_{3-k-1} (for $1 \le k \le \nu_2$) is same as $\text{Game}_{3-(k-1)-2}$ except the k^{th} replied signcryption is of sf-type I. (In this sequel, we define $\text{Game}_{3-0-2} = \text{Game}_{2-\nu_1-2}$)
- Game_{3-k-2} (for $1 \le k \le \nu_2$) is same as Game_{3-k-1} except the k^{th} replied signcryption is of II.
- Game_4 is similar to $\text{Game}_{3-\nu_2-2}$ except that the challenge signcryption is of sf-type 2.
- Game_5 is similar to Game_4 except that the challenge signcryption is of sf-type 3.

– Game$_{Final}$ is similar to Game$_5$ except that the challenge signcryption is of sf-type 4.

In Game$_{Final}$, the decommitment decom_b of the challenge message m_b is masked with an independently and uniformly chosen element from $\mathbb{G}_T$ implying the component C_{INT} does not leak any information about decom_b. Since, the primitive commitment schemes, $\mathcal{C}$ has hiding property, so com_b does not reveal any information about m_b from adversary point of view. Therefore, the adversary $\mathscr{A}$ has no advantage in Game$_{Final}$. The outline of the hybrid arguments over the games are structured in the box:

$$\begin{tikzcd}
\boxed{Real} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.2}}}{\boxed{\text{CMA}}}"] & \boxed{\widehat{Real}} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.3}}}{\boxed{\text{DSG2}}}"] & \boxed{Res} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.4}}}{\boxed{\text{DSG1}}}"] & \boxed{0} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.5}}}{\boxed{\text{DSG2}}}"] & \boxed{1-1-1}\ldots\boxed{1-(k-1)-3} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.5}}}{\boxed{\text{DSG2}}}"] & \boxed{1-k-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{1-k-1} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.6}}}{\boxed{\text{CHM}}}"] & \boxed{1-k-2} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.7}}}{\boxed{\text{DSG2}}}"] & \boxed{1-k-3}\ldots\boxed{1-q_1-3} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.8}}}{\boxed{\text{DSG2}}}"] & \boxed{1-(q_1+1)-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{1-(q_1+1)-1} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.9}}}{\boxed{\text{SHM}}}"] & \boxed{1-(q_1+1)-2} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.10}}}{\boxed{\text{DSG2}}}"] & \boxed{1-(q_1+1)-3} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.11}}}{\boxed{\text{DSG2, H}}}"] & \boxed{2-1-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{2-1-1}\ldots\boxed{2-(k-1)-2} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.11}}}{\boxed{\text{DSG2, CRH}}}"] & \boxed{2-k-1} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.12}}}{\boxed{\text{DSG2}}}"] & \boxed{2-k-2}\ldots\boxed{2-\nu_1-2} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.13}}}{\boxed{\text{DSG2, CRH}}}"] & \boxed{3-1-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{3-1-1}\ldots\boxed{3-(k-1)-2} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.13}}}{\boxed{\text{DSG2, CRH}}}"] & \boxed{3-k-1} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.14}}}{\boxed{\text{DSG2}}}"] & \boxed{3-k-2}\ldots\boxed{3-\nu_2-2} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.15}}}{\boxed{\text{DSG2, CRH}}}"] & \boxed{4}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{4} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.16}}}{\boxed{\text{DSG2}}}"] & \boxed{5} \arrow[r, Rightarrow, "\overset{\boxed{\text{Lem F.17}}}{\boxed{\text{DSG3}}}"] & \boxed{Final}
\end{tikzcd}$$

Using the above structure and Lemma F.18, we have the following reduction:

$$\begin{aligned}\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PSC-CCA}}(\kappa) \leq \mathsf{Adv}_{\mathscr{B}_0}^{\mathrm{OTS-sUF}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_1}^{\mathrm{DSG1}}(\kappa) + (2q_1 + 2\nu_1 + 2\nu_2 + 5)\mathsf{Adv}_{\mathscr{B}_2}^{\mathrm{DSG2}}(\kappa) + q_1\mathsf{Adv}_{\mathscr{B}_3}^{\mathrm{P-CMH}}(\kappa) \\ + \mathsf{Adv}_{\mathscr{B}_4}^{\mathrm{P-SMH}}(\kappa) + (\nu_1 + \nu_2 + 1)\mathsf{Adv}_{\mathscr{B}_5}^{\mathrm{CRH}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_6}^{\mathrm{DSG3}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_7}^{\mathrm{Hiding}}(\kappa)\end{aligned}$$

where $\mathscr{B}_0, \mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4, \mathscr{B}_5, \mathscr{B}_6$ and $\mathscr{B}_7$ are PPT algorithms whose running times are same as that of $\mathscr{A}$. This completes the theorem. □

Discussion F.1. By the definition of Game$_{\widehat{Real}}$, $\mathscr{B}$ returns $\perp$ to $\mathscr{A}$ if E occurs. When E does not occur, there are three possibilities, (a) $[(\mathsf{vk}^* = \mathsf{vk}) \wedge (\delta_o^*||C_0^*||y_s^* = \delta_o||C_0||y_s)]$, (b) $[(\mathsf{vk}^* \neq \mathsf{vk}) \wedge (\delta_o^*||C_0^*||y_s^* = \delta_o||C_0||y_s)]$ and (c) $[(\mathsf{vk}^* \neq \mathsf{vk}) \wedge (\delta_o^*||C_0^*||y_s^* \neq \delta_o||C_0||y_s)]$. Since, for a valid unsigncryption query $\mathsf{U} = (\mathsf{com}, \delta, \mathsf{CT})$, the case (a) implies that $(\mathsf{U}^*, y_s^*) = (\mathsf{U}, y_s)$ which is forbidden by the natural restriction of the APs-IND-CCA game. The case (b) is impossible as $C_0^* = C_0$ implies $\mathsf{vk}^* = \mathsf{vk}$ which is absurd. Therefore, from the game, Game$_{\widehat{Real}}$ onwards $\mathscr{B}$ answers the unsigncryption queries of $\mathscr{A}$ by running $\mathsf{AltUnsigncrypt}$ algorithm if the case (c) only occurs else returns $\perp$.

Remark F.2. By construction of signcryption, we have $\hbar_s \neq \hbar_e$ and since, the function $f(X) := \theta_1 X + \theta_2$ is pairwise independent function, we do not need to pay attention on distributional relation between the stuffs involved in signature and alt-key (resp. ciphertext and vText) while simulating these stuffs in sf-type 1 form.

Lemma F.2. Game$_{Real}$ *and* Game$_{\widehat{Real}}$ *are indistinguishable under the strong unforgeability of the OTS scheme,* Π_{OTS}*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{\mathrm{Real}}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{\widehat{Real}}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{OTS-sUF}}_{\mathscr{B}}(\kappa)$.

Proof. Suppose $\mathscr{A}$ can distinguish the games with a non-negligible probability, then we will program a PPT algorithm $\mathscr{B}$ for breaking the strong unforgeability of the one-time signature, OTS with the same probability. Here $\mathscr{B}$ plays the role of an adversary in sUF-CMA game and the role of a challenger in APs-IND-CCA game. Let $\mathcal{CH}$ be the challenger for OTS. $\mathcal{CH}$ runs $(\mathsf{vk}^*, \mathsf{signk}^*) \longleftarrow \mathsf{OTS.Gen}$ and gives vk^* to $\mathscr{B}$. Then, $\mathscr{B}$ runs the Setup algorithm, keeps $\mathcal{MSK}$ to itself and gives the public parameters $\mathcal{PP}$ to $\mathscr{A}$.

Query Phase-1: It consists of the following queries in adaptive manner:

– KeyGen: Let x be any key query made by $\mathscr{A}$. Since, $\mathscr{B}$ knows $\mathcal{MSK}$, it replies $\mathcal{SK}_x$ to $\mathscr{A}$.

– $\mathsf{Signcrypt}$: Let (m, x, y_s, y_e) be any signcryption query made by $\mathscr{A}$. Then, $\mathscr{B}$ constructs a key $\mathcal{SK}_x$ using $\mathcal{MSK}$. Then, using this key it runs $\mathsf{Signcrypt}$ algorithm (in sec.6) and answers the signcryption U to $\mathscr{A}$.

– $\mathsf{Unsigncrypt}$: Let (U, x, y_s), where $\mathsf{U} := (\mathsf{com}, \delta, \mathsf{CT})$ be any unsigncryption query made by $\mathscr{A}$. If this query satisfies the event E, $\mathscr{B}$ returns δ_o and aborts. $\mathscr{B}$ first constructs the normal vTextKey $\mathcal{VK} := (\mathcal{SK}^{\mathsf{M}}_x, \mathcal{V})$, then using $\mathcal{VK}$ it runs $\mathsf{AltUnsigncrypt}$ and returns the output to $\mathscr{A}$.

Challenge Phase: $\mathscr{A}$ submits two equal length message m_0, m_1, a key index x, a challenge sender associated data index y^*_s and a challenge receiver associated data index y^*_e to $\mathscr{B}$. Then, $\mathscr{B}$ computes the key $\mathcal{SK}_x$ as it knows $\mathcal{MSK}$. It picks $b \xleftarrow{\mathrm{U}} \{0, 1\}$ and runs $\mathsf{Signcrypt}(\mathcal{PP}, m_b, \mathcal{SK}_x, y^*_s, y^*_e)$, where it queries for one-time signature to $\mathcal{CH}$ for the message $C^*_0||y^*_s$ and gets the replied signature δ^*_o. It returns $\mathsf{U}^* := (\mathsf{com}^*, \delta^*, \mathsf{CT}^*)$, where $\delta^* := (\delta_{y^*_s}, \delta^*_o, \mathsf{vk}^*)$ to $\mathscr{A}$.

Query Phase-2: Similar to phase-1.

Guess: $\mathscr{A}$ sends a guess b' to $\mathscr{B}$. ($\mathscr{B}$ is nothing to do with this b')

Analysis: Since both the games are identical except the event E with probability ξ. By the event E, we have $\delta^*_o||C^*_0||y^*_s \neq \delta_o||C_0||y_s$. Therefore, δ_o is a valid forge for the message $C_0||y_s$.

□

Lemma F.3. Game$_{\widehat{Real}}$ *and* Game$_{Res}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{\widehat{Real}}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{\mathrm{Res}}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$.

Proof. Similar to Lemma E.1. □

Lemma F.4. Game$_{Res}$ *and* Game$_0$ *are indistinguishable under the DSG1 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{\mathrm{Res}}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{0}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG1}}_{\mathscr{B}}(\kappa)$.

Proof. The only difference between the games is the form of the challenge signcryption, normal or sf-type 1. In both forms of signcryptions, the signature is appeared to be normal, but the ciphertexts are normal and sf-type 1 accordingly the challenge signcryptions are normal and sf-type 1. Therefore, the proof could be done in similar way as in Lemma E.2. □

Lemma F.5. Game$_{1-(k-1)-3}$ *and* Game$_{1-k-1}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-(k-1)-3}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq q_1$.

Proof. For proof, refer to Lemma E.3. □

Lemma F.6. Game$_{1-k-1}$ *and* Game$_{1-k-2}$ *are indistinguishable under* CMH *security of the primitive pair encoding scheme,* P*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{P-CMH}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq q_1$.

Proof. Following the proof of Lemma E.4, it can be proven. □

Lemma F.7. Game$_{1-k-2}$ *and* Game$_{1-k-3}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-k-3}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq q_1$.

Proof. The proof is similar to that of Lemma E.5. □

Lemma F.8. Game$_{1-q_1-3}$ *and* Game$_{1-(q_1+1)-1}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-q_1-3}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-(q_1+1)-1}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$.

Proof. For proof, we refer to Lemma E.6 □

Lemma F.9. Game$_{1-(q_1+1)-1}$ *and* Game$_{1-(q_1+1)-2}$ *are indistinguishable under* SMH *security of the primitive pair encoding scheme,* P*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-(q_1+1)-1}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-(q_1+1)-2}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{P-SMH}}_{\mathscr{B}}(\kappa)$.

Proof. The proof is similar to Lemma E.7. □

Lemma F.10. Game$_{1-(q_1+1)-2}$ *and* Game$_{1-(q_1+1)-3}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-(q_1+1)-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-(q_1+1)-3}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$.

Proof. The proof can be done in similar manner as in Lemma E.8. □

Lemma F.11. Game$_{2-(k-1)-2}$ *and* Game$_{2-k-1}$ *are indistinguishable under the DSG2 assumption and collision resistant property of* H*. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{2-(k-1)-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{2-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq \nu_1$.

Proof. The proof is similar to that of Lemma E.9. Following the Discussion F.1 for a valid unsigncryption query, we must have $\mathsf{vk}^* \neq \mathsf{vk}$, which in turn implies that $\hbar^*_e \neq \hbar_e$. Therefore, the proof will be simpler than than the proof of Lemma E.9. □

Lemma F.12. Game$_{2-k-1}$ *and* Game$_{2-k-2}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{2-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{2-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq \nu_1$.

Proof. The proof is similar to that of Lemma E.10 □

Lemma F.13. Game$_{3-(k-1)-2}$ *and* Game$_{3-k-1}$ *are indistinguishable under the DSG2 assumption and collision resistant property of H. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{3-(k-1)-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{3-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq \nu_2$.

Proof. In both the games, the queried key is sf-type 3, the unsigncryption queries are answered by vTextKeys of the form, sf-type II, the challenge signcryption is of sf-type 1 (that means signature part is normal whereas ciphertext is sf-type 1) and the ciphertexts are normal in all queried signcryptions. The only difference between the games is the form of k^{th} queried signcryption, viz., the form of signature in k^{th} queried signcryption, i.e., it is either normal or sf-type 1. Therefore, following the proof of the Lemma D.6, it can be done. Since, the signature part in the challenge signcryption is normal form, so the collision resistant property of H will be used only to guarantee $\hbar^*_e \neq \hbar_s$ in the proof. □

Lemma F.14. Game$_{3-k-1}$ *and* Game$_{3-k-2}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{3-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{3-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq \nu_2$.

Proof. The only difference between the games is the forms of k^{th} queried signcryption, viz., the form of signature, i.e., it is either sf-type 1 or sf-type 2. We refer to proof of Lemma D.7. □

Lemma F.15. Game$_{3-\nu_2-2}$ *and* Game$_4$ *are indistinguishable under the DSG2 assumption and collision resistant property of H. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{3-\nu_2-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{4}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa)$.

Proof. In both the games, the queried key is sf-type 3, the unsigncryption queries are answered by vTextKeys of the form, sf-type II, the queried signcryptions are of sf-type II (signature part is sf-type 2 whereas ciphertext normal) and the ciphertext in the challenge signcryption is sf-type 1. The only difference between the games is the form of signature in the challenge signcryption, i.e., it is either normal or sf-type 1. Therefore, the proof can be done following the proof of the Lemma D.6. □

Lemma F.16. Game$_4$ *and* Game$_5$ *are indistinguishable under the DSG3 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{4}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{5}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$.

Proof. The only difference between the games is the form of the signature in the challenge signcryption, i.e., it is either sf-type 1 or sf-type 2. We refer to proof of the Lemma D.7. □

Lemma F.17. Game$_5$ *and* Game$_{Final}$ *are indistinguishable under the DSG3 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^{5}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{Final}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG3}}_{\mathscr{B}}(\kappa)$.

Proof. The only difference between the games is the form of the ciphertext in the challenge signcryption, i.e., it is either sf-type 1 or sf-type 2. For proof, we refer to the Lemma E.11 □

Lemma F.18. *For every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $\mathsf{Adv}^{Final}_{\mathscr{A},\mathrm{PSC}}(\kappa) \leq \mathsf{Adv}^{\mathrm{Hiding}}_{\mathscr{B}}(\kappa)$.

Proof. In the final game, Game_{Final} the decommitment part, decom_b of the challenge message m_b is information theoretically hidden in the challenge signcryption, viz., ciphertext. The only counter part of m_b remains in challenge signcryption is the commitment part com_b. Since, the commitment scheme Π_{Commit} has hiding property, com_b does not leak any information about m_b from adversary point of view. We skip the details of the simulation. □

Theorem F.19. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has the* PMH *security, the assumptions, DSG1, DSG2 and DSG3 hold in* $\mathcal{J}$*, the one-time signature scheme,* OTS *has strong unforgeability, the commitment scheme,* $\mathcal{C}$ *has the hiding property and* H *is a collision resistant hash function, then the proposed predicate signcryption scheme,* PSC *in sec.6 for the predicate* $\sim$ *is adaptive-predicates IND-CCA secure.*

Proof. Similar to the proof of Theorem F.1. The reduction of the proof is given by

$$\begin{aligned}\mathsf{Adv}_{\mathscr{A}}^{\mathrm{PSC-CCA}}(\kappa) \leq \mathsf{Adv}_{\mathscr{B}_0}^{\mathrm{OTS-sUF}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_1}^{\mathrm{DSG1}}(\kappa) + (2q + 2\nu_1 + 2\nu_2 + 3)\mathsf{Adv}_{\mathscr{B}_2}^{\mathrm{DSG2}}(\kappa) \\ + (\nu_1 + \nu_2 + 1)\mathsf{Adv}_{\mathscr{B}_3}^{\mathrm{CRH}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_4}^{\mathrm{DSG3}}(\kappa) + \mathsf{Adv}_{\mathscr{B}_5}^{\mathrm{Hiding}}(\kappa)\end{aligned}$$

where q, ν_1 and ν_2 respectively be the number of key, unsigncryption and signcryption queries and $\mathscr{B}_0, \mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4, \mathscr{B}_5$ are PPT algorithms whose running times are same as that of $\mathscr{A}$. This completes the theorem. □

F.3 The Proof of Unforgeability

Theorem F.20. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has the* CMH *security, the assumptions, DSG1, DSG2 and DSG3 hold in* $\mathcal{J}$*, the one-time signature scheme,* OTS *has strong unforgeability and* H *is a collision resistant hash function, then the proposed predicate signcryption scheme,* PSC *in sec.6 for the predicate* $\sim$ *is adaptive-predicates strong unforgeable.*

Proof. Let $\mathscr{A}$ be an adversary in APs-sUF-CMA model who can break the strong unforgeability of the proposed predicate signcryption scheme with non-negligible advantage ϵ. Let ν_2 be the number of signcryption queries made by $\mathscr{A}$. Let $(m^{(i)}, x^{(i)}, y_s^{(i)}, y_e^{(i)})$ be the i^{th} query and $\mathsf{U}^{(i)} := (\mathsf{com}_i, \delta^{(i)} := (\delta_{y_s^{(i)}}, \delta_o^{(i)}, \mathsf{vk}^{(i)}), \mathsf{CT}^{(i)})$ be the corresponding replied signcryption. Let $\mathsf{U}^* := (\mathsf{com}^*, \delta^*, \mathsf{CT}^*)$ be the forgery made by $\mathscr{A}$ for the message (m^*, y_s^*, y_e^*). We define an event as

$$\mathsf{Forged} := \mathsf{vk}^* \notin \{\mathsf{vk}^{(i)} \big| \ i \in [\nu_2]\}$$

Then, we have

$$\epsilon \leq \Pr[\mathscr{A} \ \text{ Succeeds}] := \Pr[\mathscr{A} \ \text{ Succeeds} \wedge \mathsf{Forged}] + \Pr[\mathscr{A} \ \text{ Succeeds} \wedge \neg\mathsf{Forged}]$$

$$\implies \Pr[\mathscr{A} \ \text{ Succeeds} \wedge \mathsf{Forged}] \geq \epsilon/2 \ \ or \ \ \Pr[\mathscr{A} \ \text{ Succeeds} \wedge \neg\mathsf{Forged}] \geq \epsilon/2$$

Case $\neg(\mathsf{Forged})$: We will develop an algorithm $\mathscr{B}_{\mathsf{OTS}}$ for breaking the string unforgeability of the primitive one-time signature scheme, OTS with advantage at least $\epsilon/2\nu_2$. Let $\mathcal{CH}$ be the challenger for the primitive one-time signature scheme, OTS. The challenger $\mathcal{CH}$ runs $(\mathsf{vk}^*, \mathsf{signk}^*) \longleftarrow \mathsf{OTS.Gen}(1^\kappa)$ and gives vk^* to $\mathscr{B}_{\mathsf{OTS}}$. $\mathscr{B}_{\mathsf{OTS}}$ runs the Setup algorithm (as described in section 6), keeps $\mathcal{MSK}$ to itself and sends $\mathcal{PP}$ to $\mathscr{A}$. Then, it picks $i \xleftarrow{\mathrm{U}} [\nu_2]$ as a guess such that $\mathsf{vk}^* = \mathsf{vk}^{(i)}$. For notational simplicity, we ignore the superscript (i).

- $\mathsf{KeyGen\ Query}$: $\mathscr{B}_{\mathsf{OTS}}$ answers this query using $\mathcal{MSK}$.
- $\mathsf{Signcrypt\ Query}$: Let (m, x, y_s, y_e) be the j^{th} signcryption query to $\mathscr{B}_{\mathsf{OTS}}$ by $\mathscr{A}$.
  - $(j \neq i)$ **:**
    $\mathscr{B}_{\mathsf{OTS}}$ executes $(\mathsf{com}, \mathsf{decom}) := \mathsf{Commit}(m)$, $(\mathsf{vk}, \mathsf{signk}) := \mathsf{OTS.Gen}(1^\kappa)$. It constructs the key $\mathcal{SK}_x$ using $\mathcal{MSK}$. Then, it runs $\delta_{y_s} \leftarrow \mathsf{PS.Sign}(\mathsf{vk}, \mathcal{SK}_x, y_s)$, $C_{\mathsf{cpa}} \leftarrow \mathsf{PE.Encrypt}^*(\mathsf{decom}, y_e)$ and computes $\hbar_e := H(\mathsf{com}, \delta_{y_s}, \mathsf{vk}, C_{\mathsf{cpa}})$. Then, it computes $C_0 := g^{s(\theta_1 \hbar_e + \theta_2)}$ and $\delta_o := \mathsf{OTS.Sign}(C_0||y_s, \mathsf{signk})$. It returns the signcryption $\mathsf{U} := (\mathsf{com}, \delta := (\delta_{y_s}, \delta_o, \mathsf{vk}), \mathsf{CT} := (C_{\mathsf{cpa}}, C_0))$ to $\mathscr{A}$.
  - $(j = i)$ **:**
    Same as above except $\mathscr{B}_{\mathsf{OTS}}$ does not execute $\mathsf{OTS.Gen}(1^\kappa)$ but it sets $\mathsf{vk} := \mathsf{vk}^*$ and it makes an one-time signature query to $\mathcal{CH}$ for the message $C_0||y_s$ and gets the replied signature δ_o.
- $\mathsf{Unsigncrypt\ Query}$: It can answer the query as it knows $\mathcal{MSK}$.
- $\mathsf{Forgery}$: $\mathscr{A}$ outputs a tuple $(\mathsf{U}^*, y_s^*, y_e^*)$, where $\mathsf{U}^* := (\mathsf{com}^*, \delta^*, \mathsf{C}^*)$ and $\delta^* := (\delta_{y_s^*}, \delta_o^*, \mathsf{vk}^*)$. Then, $\mathscr{B}_{\mathsf{OTS}}$ forges the signature δ_o^* for $C_0^*||y_s^*$ to the one-time signature scheme, OTS.

$\mathsf{Analysis}$: With probability $1/\nu_2$, $\mathscr{B}_{\mathsf{OTS}}$ correctly guesses i such that the event Forged is happened. Now, we only have to show that $\delta_o^*||C_0^*||y_s^* \neq \delta_o||C_0||y_s$ (we ignore the superscript, (i)). Indeed, if $\delta_o^*||C_0||y_s^* = \delta_o||C_0^*||y_s$, we have $\delta_o^* = \delta_o$, $y_s^* = y_s$ and $C_0^* = C_0$. Now $C_0^* = C_0$ implies $\hbar_e^* = \hbar_e$, so we have $\mathsf{com}^* = \mathsf{com}$, $\delta_{y_s^*} = \delta_{y_s}$ and $C_{\mathsf{cpa}}^* = C_{\mathsf{cpa}}$. Overall, we have $(\mathsf{U}^*, m^*, y_s^*, y_e^*) = (\mathsf{U}, m, y_s, y_e)$ which leads a contradiction to *APs-sUF-CMA* security model.

Case Forged : Suppose there are at most q key queries and ν_1 unsigncryption queries, then the security proof consists of hybrid argument over a sequence of $3q + 2(\nu_1 + \nu_2) + 7$ games defined below.

- Game$_{Real}$:= The original APs-sUF-CMA game.
- Game$_{\widehat{Real}}$:= Same as Game$_{Real}$ except the event Forged is always happened.
- Game$_{Res}$:= This is same as Game$_{\widehat{Real}}$ except $x \not\sim_N y^*$ is replaced by $x \not\sim_{p_2} y^*$ for each key query x made by $\mathscr{A}$.
- Game$_0$ (= Game$_{1-0-3}$) is just like Game$_{Res}$ except that the vTextKey for verifying the forgery is of sf-type 1.
- In Game$_{1-k-1}$ (for $1 \leq k \leq q$) is same as Game$_{1-(k-1)-3}$ except the k^{th} queried key is sf-type 1.
- Game$_{1-k-2}$ (for $1 \leq k \leq q$) is same as Game$_{1-k-1}$ except the k^{th} queried key is sf-type 2.
- Game$_{1-k-3}$ (for $1 \leq k \leq q$) is same as Game$_{1-k-2}$ except the k^{th} queried key is sf-type 3.
- In Game$_{2-k-1}$ (for $1 \leq k \leq \nu_1$) is same as Game$_{2-(k-1)-2}$ except the k^{th} unsigncryption query is answered by the vTextKey of sf-type I. (In this sequel, we define Game$_{2-0-2}$ = Game$_{1-q-3}$)
- Game$_{2-k-2}$ (for $1 \leq k \leq \nu_1$) is same as Game$_{2-k-1}$ except the k^{th} unsigncryption query is answered by the vTextKey of sf-type II.
- In Game$_{3-k-1}$ (for $1 \leq k \leq \nu_2$) is same as Game$_{3-(k-1)-2}$ except the k^{th} replied signcryption is of sf-type I. (So, in this sequel Game$_{3-0-2}$ = Game$_{2-\nu_1-2}$)
- Game$_{3-k-2}$ (for $1 \leq k \leq \nu_2$) is same as Game$_{3-k-1}$ except the k^{th} replied signcryption is of sf-type II.
- Game$_4$ is similar to Game$_{3-\nu_2-2}$ except that the vTextKey for verifying the forgery is sf-type 2.
- Game$_5$ is similar to Game$_4$ except that the vTextKey for verifying the forgery is sf-type 3.

– Game$_{Final}$ is similar to Game$_5$ except that the vTextKey for verifying the forgery is sf-type 4

The outline of the hybrid arguments over the games are structured in the box:

$$\begin{tikzcd}
\boxed{Real} \arrow[r, Rightarrow, "\boxed{\text{Forged}}"] & \boxed{\widehat{Real}} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.21}}\\ \boxed{\text{DSG2}}}"] & \boxed{Res} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.22}}\\ \boxed{\text{DSG1}}}"] & \boxed{0} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.23}}\\ \boxed{\text{DSG2}}}"] & \boxed{1-1-1}\ldots\boxed{1-(k-1)-3} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.23}}\\ \boxed{\text{DSG2}}}"] & \boxed{1-k-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{1-k-1} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.24}}\\ \boxed{\text{CHM}}}"] & \boxed{1-k-2} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.25}}\\ \boxed{\text{DSG2}}}"] & \boxed{1-k-3}\ldots\boxed{1-q-3} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.26}}\\ \boxed{\text{DSG2}}}"] & \boxed{2-1-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{2-1-1}\ldots\boxed{2-(k-1)-2} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.26}}\\ \boxed{\text{DSG2, CRH}}}"] & \boxed{2-k-1} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.27}}\\ \boxed{\text{DSG2}}}"] & \boxed{2-k-2}\ldots\boxed{2-\nu_1-2} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.28}}\\ \boxed{\text{DSG2, CRH}}}"] & \boxed{3-1-1}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{3-1-1}\ldots\boxed{3-(k-1)-2} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.28}}\\ \boxed{\text{DSG2, CRH}}}"] & \boxed{3-k-1} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.29}}\\ \boxed{\text{DSG2}}}"] & \boxed{3-k-2}\ldots\boxed{3-\nu_2-2} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.30}}\\ \boxed{\text{DSG2, CRH}}}"] & \boxed{4}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{4} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.31}}\\ \boxed{\text{DSG2}}}"] & \boxed{5} \arrow[r, Rightarrow, "\substack{\boxed{\text{Lem F.32}}\\ \boxed{\text{DSG3}}}"] & \boxed{Final}
\end{tikzcd}$$

Using the above structure, we have the following reduction:

$$\begin{aligned}\mathsf{Adv}^{\mathrm{PSC-sUF}}_{\mathscr{A}}(\kappa) \leq{}& \nu_2\mathsf{Adv}^{\mathrm{OTS-sUF}}_{\mathscr{B}_0}(\kappa) + \mathsf{Adv}^{\mathrm{DSG1}}_{\mathscr{B}_1}(\kappa) + (2q + 2\nu_1 + 2\nu_2 + 3)\mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}_2}(\kappa)\\ &+ q\mathsf{Adv}^{\mathrm{P-CMH}}_{\mathscr{B}_3}(\kappa) + (\nu_1 + \nu_2 + 1)\mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}_4}(\kappa) + \mathsf{Adv}^{\mathrm{DSG3}}_{\mathscr{B}_5}(\kappa)\end{aligned}$$

where $\mathscr{B}_0, \mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4$ and $\mathscr{B}_5$ are PPT algorithms whose running times are same as that of $\mathscr{A}$. This completes the theorem. ☐

Lemma F.21. Game$_{\widehat{Real}}$ *and* Game$_{Res}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{\widehat{Real}}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{\mathrm{Res}}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$.

Proof. Similar to Lemma D.1. ☐

Lemma F.22. Game$_{Res}$ *and* Game$_0$ *are indistinguishable under the DSG1 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{\mathrm{Res}}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{0}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG1}}_{\mathscr{B}}(\kappa)$.

Proof. The proof could be done in similar way as in Lemma D.2. ☐

Lemma F.23. Game$_{1-(k-1)-3}$ *and* Game$_{1-k-1}$ *are indistinguishable under the DSG2 assumption. That is, for every adversary* $\mathscr{A}$*, there exists a PPT algorithm* $\mathscr{B}$ *such that* $|\mathsf{Adv}^{1-(k-1)-3}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ *for* $1 \leq k \leq q$.

Proof. For proof, refer to Lemma D.3. ☐

Lemma F.24. *Game_{1-k-1} and Game_{1-k-2} are indistinguishable under* CMH *security of the pair encoding scheme,* P*. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that $|\mathsf{Adv}^{1-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{P-CMH}}_{\mathscr{B}}(\kappa)$ for $1 \leq k \leq q$.*

Proof. Following the proof of Lemma D.4, it can be proven. □

Lemma F.25. *Game_{1-k-2} and Game_{1-k-3} are indistinguishable under the DSG2 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that $|\mathsf{Adv}^{1-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{1-k-3}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ for $1 \leq k \leq q$.*

Proof. The proof is similar to that of Lemma D.5. □

Lemma F.26. *$\text{Game}_{2-(k-1)-2}$ and Game_{2-k-1} are indistinguishable under the DSG2 assumption and collision resistant property of H. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that $|\mathsf{Adv}^{2-(k-1)-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{2-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa)$ for $1 \leq k \leq \nu_1$.*

Proof. The proof is similar to that of Lemma E.9. The collision resistant property of H will be used only to guarantee $\hbar^*_s = \hbar^{(k)}_e$ (following the remark F.2). □

Lemma F.27. *Game_{2-k-1} and Game_{2-k-2} are indistinguishable under the DSG2 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that $|\mathsf{Adv}^{2-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{2-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ for $1 \leq k \leq \nu_1$.*

Proof. The proof is similar to that of Lemma E.10 □

Lemma F.28. *$\text{Game}_{3-(k-1)-2}$ and Game_{3-k-1} are indistinguishable under the DSG2 assumption and collision resistant property of H. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that $|\mathsf{Adv}^{3-(k-1)-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{3-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa)$ for $1 \leq k \leq \nu_2$.*

Proof. In both the games, the vTextKey for verifying the forgery is of sf-type 1, the queried key is sf-type 3, the unsigncryption queries are answered by vTextKeys of sf-type II and the ciphertexts are normal in all the queried signcryptions. The only difference between the games is the form of k^{th} queried signcryption, viz, the form of signature in k^{th} queried signcryption, i.e., it is either normal or sf-type 1. By the event Forged, we have $\mathsf{vk}^* \neq \mathsf{vk}^{(k)}$, so $\hbar^*_s \neq \hbar^{(k)}_s$. Therefore, the proof can be done following the proof of the Lemma D.6. □

Lemma F.29. *Game_{3-k-1} and Game_{3-k-2} are indistinguishable under the DSG2 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that $|\mathsf{Adv}^{3-k-1}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{3-k-2}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$ for $1 \leq k \leq \nu_2$.*

Proof. The only difference between the games is the forms of k^{th} queried signcryption, viz, the form of signature, sf-type 1 or sf-type 2. We refer to proof of Lemma D.7. □

Lemma F.30. *$\text{Game}_{3-\nu_2-2}$ and Game_4 are indistinguishable under the DSG2 assumption and collision resistant property of H. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that $|\mathsf{Adv}^{3-\nu_2-2}_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{4}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa) + \mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}}(\kappa)$.*

Proof. In both the games, the queried key is sf-type 3, the unsigncryption queries are answered by vTextKeys of sf-type II, the queried signcryptions are of sf-type II (signature part is sf-type 2 whereas ciphertext normal) and the vText in the vTextKey for verifying the forgery is of sf-type 1. The only difference between the games is the form of alt-key in the vTextKey for verifying the forgery, i.e., it is either normal or sf-type 1. Therefore, the proof can be done following the proof of the Lemma D.6. The collision resistant property of H will be used only to guarantee $\hbar_s^* = \hbar_e^*$ (following the remark F.2) in the construction of vTextKey for verifying the forgery. $\square$

Lemma F.31. Game$_4$ *and* Game$_5$ *are indistinguishable under the DSG3 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^4_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^5_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}}(\kappa)$.

Proof. The only difference between the games is the form of the alt-key in the vTextKey for verifying the forgery, i.e., it is either normal or sf-type 1. We refer to proof of the Lemma D.7. $\square$

Lemma F.32. Game$_5$ *and* Game$_{Final}$ *are indistinguishable under the DSG3 assumption. That is, for every adversary $\mathscr{A}$, there exists a PPT algorithm $\mathscr{B}$ such that* $|\mathsf{Adv}^5_{\mathscr{A},\mathrm{PSC}}(\kappa) - \mathsf{Adv}^{Final}_{\mathscr{A},\mathrm{PSC}}(\kappa)| \leq \mathsf{Adv}^{\mathrm{DSG3}}_{\mathscr{B}}(\kappa)$.

Proof. The only difference between the games is the form of the vText in the vTextKey for verifying the forgery, i.e., vText is either sf-type 1 or sf-type 2. For proof, we refer to the Lemma E.11 $\square$

Theorem F.33. *Let* P *be a pair encoding scheme for a predicate* $\sim$ *which satisfies* **Conditions 3.2** *and* $\sim$ *is domain-transferable. Suppose* P *has the* PMH *security, the assumptions, DSG1, DSG2 and DSG3 hold in $\mathcal{J}$, the one-time signature scheme,* OTS *has strong unforgeability and H is a collision resistant hash function, then the proposed predicate signcryption scheme,* PSC *in sec.6 for the predicate* $\sim$ *is adaptive-predicates strong unforgeable.*

Proof. Similar to the proof of Theorem F.20. The reduction of the proof is given by

$$\begin{aligned}\mathsf{Adv}^{\mathrm{PSC-sUF}}_{\mathscr{A}}(\kappa) \leq \nu_2 \mathsf{Adv}^{\mathrm{OTS-sUF}}_{\mathscr{B}_0}(\kappa) + \mathsf{Adv}^{\mathrm{DSG1}}_{\mathscr{B}_1}(\kappa) + (2q + 2\nu_1 + 2\nu_2 + 3)\mathsf{Adv}^{\mathrm{DSG2}}_{\mathscr{B}_2}(\kappa) \\ + (\nu_1 + \nu_2)\mathsf{Adv}^{\mathrm{CRH}}_{\mathscr{B}_3}(\kappa) + \mathsf{Adv}^{\mathrm{DSG3}}_{\mathscr{B}_4}(\kappa)\end{aligned}$$

where q, ν_1 and ν_2 respectively be the number of key, unsigncryption and unsigncryption queries made by $\mathscr{A}$ and $\mathscr{B}_0, \mathscr{B}_1, \mathscr{B}_2, \mathscr{B}_3, \mathscr{B}_4$ are PPT algorithms whose running times are same as that of $\mathscr{A}$. This completes the theorem. $\square$