J Cryptol (2016) 29:833–878
DOI: 10.1007/s00145-015-9211-7

Journal of CRYPTOLOGY

Constant-Size Structure-Preserving Signatures: Generic Constructions and Simple Assumptions[1]

Masayuki Abe · Ryo Nishimaki
NTT Secure Platform Laboratories, NTT Corporation, Tokyo, Japan
abe.masayuki@lab.ntt.co.jp; nishimaki.ryo@lab.ntt.co.jp

Melissa Chase
Microsoft Research, Redmond, WA, USA
melissac@microsoft.com

Bernardo David
Aarhus University, Aarhus, Denmark
bernardo@cs.au.dk

Markulf Kohlweiss
Microsoft Research, Cambridge, UK
markulf@microsoft.com

Miyako Ohkubo
Security Fundamentals Laboratory, NSRI, NICT, Tokyo, Japan
m.ohkubo@nict.go.jp

Communicated by Hugo Krawczyk.

Received 26 January 2014
Online publication 8 July 2015

Abstract. This paper presents efficient structure-preserving signature schemes based on simple assumptions such as decisional linear. We first give two general frameworks for constructing fully secure signature schemes from weaker building blocks such as variations of one-time signatures and random message secure signatures. They can be seen as refinements of the Even–Goldreich–Micali framework, and preserve many desirable properties of the underlying schemes such as constant signature size *and structure preservation*. We then instantiate them based on simple (i.e., not q-type) assumptions over symmetric and asymmetric bilinear groups. The resulting schemes are structure-preserving and yield constant-size signatures consisting of 11–14 group elements, which compares favorably to existing schemes whose security relies on q-type assumptions.

Keywords. Structure-preserving signatures, Tagged one-time signatures, Partially one-time signatures, Extended random message attacks.

[1] Full version of [2] incorporating a part of results in [3].

1. Introduction

A structure-preserving signature (SPS) scheme [4] is a digital signature scheme with two structural properties: (1) the verification keys, messages, and signatures are all elements of a bilinear group; and (2) the verification algorithm checks a conjunction of pairing product equations over the key, the message, and the signature. This makes them compatible with the efficient non-interactive proof system for pairing product equations by Groth and Sahai (GS) [33]. Structure-preserving cryptographic primitives promise to combine the advantages of optimized number theoretic non-black-box constructions with the modularity and insight into protocols that use only generic cryptographic building blocks.

Indeed the instantiation of known generic constructions with an SPS scheme and the GS proof system has led to many new and more efficient schemes: Groth [32] showed how to construct an efficient simulation-sound zero-knowledge proof system (ss-NIZK) building on generic constructions of [20,37,41]. Abe et al. [4,7] show how to obtain efficient round-optimal blind signatures by instantiating a framework by Fischlin [23]. SPS are also important building blocks for a wide range of cryptographic functionalities such as anonymous proxy signatures [25], delegatable anonymous credentials [9], transferable e-cash [26] and compact verifiable shuffles [18]. Most recently, Hofheinz and Jager [34] show how to construct a structure- preserving tree-based signature scheme with a tight security reduction following the approach of [21,29]. This signature scheme is then used to build a ss-NIZK which in turn is used with the Naor and Yung [38] and Sahai [40] paradigm to build the first CCA-secure public-key encryption scheme with a tight security reduction. Examples for other schemes that benefit from efficient SPS are [8,10,11,14,24,27,30,31,35,39].

Because properties (1) and (2) are the only dependencies on the SPS scheme made by these constructions, any structure-preserving signature scheme can be used as a drop-in replacement. Unfortunately, all known efficient instantiations of SPS [4,5,7] are based on so-called q-type or interactive assumptions. An open question since Groth's seminal work [32] (only partially answered by Chase and Kohlweiss [17]) is to construct a SPS scheme that is both efficient—in particular *constant-size* in the number of signed group elements—and that is based on assumptions that are as weak as those required by the GS proof system itself.

1.1. *Our Contribution*

We begin by presenting two new generic constructions of signature schemes that are secure against chosen message attacks (CMA) from variations of one-time signatures and signatures secure against random message attacks (RMA). Both constructions inherit the structure-preserving and constant-size properties from the underlying components. We then instantiate the building blocks with the desired properties over bilinear groups. They yield constant-size structure-preserving signature schemes whose signatures consist of only 11–14 group elements and whose security can be proven based on simple assumptions such as decisional linear (DLIN) for symmetric bilinear groups and analogues of DDH and DLIN for asymmetric bilinear groups. These are the first constant-size structure-preserving signature schemes that eliminate the use of interactive or q-type assumptions while achieving reasonable efficiency. We give more details on our generic constructions and their instantiations:

- The first generic construction (SIG1, Sect. 4.1) combines a new variation of one-time signatures which we call *tagged one-time signatures* (TOS) and signatures secure against *random message attacks* (RMA). A TOS is a signature scheme that attaches a fresh tag to each signature. It is unforgeable with respect to tags used only once. In our construction, a message is signed with our TOS using a fresh random tag, and then the tag is signed with the second signature scheme, denoted by rSIG. Since rSIG only signs random tags, RMA security is sufficient.
  In Sect. 5, we construct structure-preserving TOS and rSIG based on DLIN over symmetric (Type-I) bilinear groups. Our TOS yields constant-size signatures and optimally small tags that consist of only one group element. The resulting structure-preserving signature scheme produces signatures consisting of 14 group elements, and relies solely on the DLIN assumption.[1]
- The second generic construction (SIG2, Sect. 4.2) combines *partial one-time signatures* and signatures secure against *extended random message attacks* (XRMA). The latter is a new notion that we explain below. A partial one-time signature scheme, denoted by POS, is a one-time signature scheme in which only a part of the key is renewed for every signing operation. The notion was first introduced by Bellare and Shoup [12] under the name of two-tier signatures. In our construction, a message is signed with POS and then the one-time portion of the public-key is certified by the second signature scheme, denoted by xSIG. The difference between a TOS and POS is that a one-time public-key is associated with a one-time secret-key. Since the one-time secret-key is needed for signing, it must be known to the reduction in the security proof. XRMA security guarantees that xSIG is unforgeable even if the adversary is given auxiliary information associated with the randomly chosen messages (e.g. the random coins used for selecting the message). The auxiliary information allows the reduction algorithm of the security proof of the second scheme to use the one-time secret-key to generate the POS component correctly.
  In Sect. 6, we construct structure-preserving POS and xSIG signature schemes based on assumptions that are analogues of DDH and DLIN in Type-III bilinear groups. The resulting SIG2 is structure-preserving and produces signatures consisting of 11 or 14 group elements depending on whether messages belong to either or both source groups.

The role of TOS and POS is to compress a message into a constant number of random group elements. This observation is interesting in light of [6] that implies the impossibility of constructing collision resistant and shrinking structure-preserving hash functions, which could immediately yield constant-size signatures. Our (extended) RMA-secure signature schemes are structure-preserving variants of Waters' dual-signature scheme [44]. In general, the difficulty of constructing CMA-secure SPS arises from the fact that the exponents of the group elements chosen by the adversary as a message are

[1] The optimal TOS proposed in this paper was first presented in [3]. We included it here as it saves one group element in a tag compared to the original construction in [2], and reduces the resulting signature size from 17 in [2] to 14.

not known to the reduction in the security proof. On the other hand, for RMA security, it is the challenger that chooses the message and therefore the exponents can be known in reductions. This is the crucial advantage for constructing (extended) RMA-secure structure-preserving signature schemes based on Waters' dual-signature scheme.

As our SPSs can be drop-in replacements for existing SPS, we only briefly introduce recent applications in Sect. 7. They include group signatures, tightly-secure structure-preserving signatures and public-key encryption, and efficient adaptive oblivious transfer.

1.2. *Related Works*

1.2.1. *On Generic Constructions*

Even et al. [22] proposed a generic framework (the EGM framework) that combines a one-time signature scheme and a signature scheme that is secure against non-adaptive chosen message attacks (NACMA) to construct a signature scheme that is secure against adaptive chosen message attacks (CMA).

In fact, our generic constructions can be seen as refinements of the EGM framework. There are two reasons why the original framework falls short for our purpose. *The first* is that relaxing to NACMA does not seem to help much in constructing efficient structure-preserving signatures since the messages are still under the control of the adversary, and the exponents of the messages are not known to the reduction algorithm in the security proof. As mentioned above, resorting to (extended) RMA is a great help in this regard. In [22], they also showed that CMA-secure signatures exist *iff* RMA-secure signatures exist. The proof, however, does not follow their framework and their impractical construction is mainly a feasibility result. In fact, we argue that RMA-security alone is not sufficient for the original EGM framework. As mentioned above, the necessity of XRMA security arises in the reduction that uses RMA-security to argue security of the ordinary signature scheme, as the reduction not only needs to know the random one-time public-keys, but also their corresponding one-time secret-keys in order to generate the one-time signature components of the signatures. The auxiliary information in the XRMA definition facilitates access to these secret-keys. Similarly, tagged one-time signatures avoid this problem as tags do not have associated secret values. This observation applies also to a variation of the EGM framework in [42] that combines a trapdoor hash function and a NACMA-secure signature scheme. *The second reason* that the EGM approach is not quite suited to our task is that the EGM framework produces signatures that are linear in the size of one-time public-keys of the one-time signature scheme, and known structure-preserving one-time signature schemes have one-time public-keys that scale linearly with the number of group elements to be signed. Here, tagged or partial one-time signature schemes come in handy as they have one-time public-keys separated from long-term public-keys. Thus, to obtain constant-size signatures, we only require the one-time keys to be constant-size while allowing the long-term part to scale in the size of the message.

1.2.2. *On Efficient Instantiations*

All previous constructions of structure-preserving signature schemes either are inefficient, or use strong assumptions, or do not yield constant-size signatures. In particular, there are few schemes that are based on simple assumptions. Hofheinz and Jager [34] constructed an SPS scheme by following the EGM framework. The resulting scheme allows a tight security reduction to DLIN, but the size of signatures depends logarithmically on the number of signing operations as their NACMA-secure scheme is tree-based (like the Goldwasser–Micali–Rivest signature scheme [29]). Chase and Kohlweiss [17] and Camenisch et al. [15] constructed SPS schemes with security based on DLIN that improve the performance of Groth's scheme [32] by several orders of magnitude. The size of the resulting signatures, however, is still linear in the number of signed group elements.

2. Preliminaries

2.1. *Notation*

By $X := Y$, we denote that object Y is referred to as X. For set X, notation $a \leftarrow X$ denotes a uniform sampling from X. Multiple independent samples from the same set X are denoted by $a_1, a_2, a_3, \ldots \leftarrow X$. By $Y \leftarrow A(X)$, we denote the process where algorithm A is executed with X as input and its output is labeled as Y. When A is an oracle algorithm that interacts with oracle $\mathcal{O}$, it is denoted as $Y \leftarrow A^{\mathcal{O}}(X)$. By $\Pr[X \mid A_1, A_2, \ldots, A_k]$ we denote the probability that event X happens after executing the sequence of algorithms $A_1, \ldots, A_k$. The probability is taken over all coin flips observed in $A_1, \ldots, A_k$ unless otherwise noted. We say that a function ϵ is negligible in security parameter λ if $\epsilon < \lambda^{-c}$ holds for all constant $c > 0$ and all sufficiently large λ. We refer to probabilistic polynomial- time algorithms as p.p.t. algorithms. Unless stated otherwise, we assume that all algorithms are potentially probabilistic.

2.2. *Bilinear Groups*

Let $\mathcal{G}$ be a bilinear group generator that takes security parameter 1^λ and outputs a description of bilinear groups $\Lambda := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are groups of prime order p, and e is an efficient and non-degenerate bilinear map $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. In this paper, generators for $\mathbb{G}_1$ and $\mathbb{G}_2$ are implicit in Λ, and default random generators G and $\hat{G}$ are chosen explicitly and independently. Groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are called the source groups and $\mathbb{G}_T$ is called the target group. We use multiplicative notation for $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$. By $\mathbb{G}_1^*$, we denote $\mathbb{G}_1 \backslash \{1\}$, which is the set of all elements in $\mathbb{G}_1$ except the identity. The same applies to $\mathbb{G}_2$ and $\mathbb{G}_T$ as well. Following the terminology in [28], we say that Λ is Type-III when there is no efficient mapping between $\mathbb{G}_1$ and $\mathbb{G}_2$ in either direction.

In the Type-III setting, we denote elements in $\mathbb{G}_2$ by putting a tilde on a variable like $\tilde{X}$ for visual aid. By using the same letter for elements in $\mathbb{G}_2$ and $\mathbb{G}_1$ with a hat on the $\mathbb{G}_2$ element, e.g., X and $\hat{X}$, we denote a pair of elements in relation $\log_G X = \log_{\hat{G}} \hat{X}$. Should their relation be explicitly stated, we write $X \sim \hat{X}$. Note that default random generators G and $\hat{G}$ are independent of each other but notational consistency retains.

We count the number of group elements to measure the size of cryptographic objects such as keys, messages, and signatures. For Type-III groups, we denote the size by (x, y) when it consists of x and y elements from $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. We refer to the setting as Type-I when $\mathbb{G}_1 = \mathbb{G}_2$ (i.e., there are efficient mappings in both directions). This is also called the symmetric setting. In this case, we define $\Lambda := (p, \mathbb{G}, \mathbb{G}_T, e)$. When we need to be specific, the group description yielded by $\mathcal{G}$ will be written as Λ_{asym} or Λ_{sym}.

2.3. *Assumptions*

Let $\mathcal{G}$ be a generator of bilinear groups. All hardness assumptions we deal with are defined relative to $\mathcal{G}$. We first define the computational and decisional Diffie–Hellman assumptions (CDH_1, DDH_1) and decisional linear assumption (DLIN_1) for Type-III bilinear groups. The corresponding more standard assumptions, CDH, DDH, and DLIN, in Type-I groups are obtained by setting $\mathbb{G}_1 = \mathbb{G}_2$ and $G = \hat{G}$ in the respective definitions.

Definition 1. (*Computation co-Diffie–Hellman assumption*: CDH_1) Given $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, $G \leftarrow \mathbb{G}_1^*$, $\hat{G} \leftarrow \mathbb{G}_2^*$, G^x, G^y, $\hat{G}^x$, and $\hat{G}^y$ for $x, y \leftarrow \mathbb{Z}_p$, any p.p.t. algorithm $\mathcal{A}$ outputs G^{xy} with negligible probability $\text{Adv}^{\mathsf{co\text{-}cdh}}_{\mathcal{G},\mathcal{A}}(\lambda)$ in λ.

Definition 2. (*Decisional Diffie—Hellman assumption in* $\mathbb{G}_1$: DDH_1) Given $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, $G \leftarrow \mathbb{G}_1^*$, and (G^x, G^y, Z_b) where $Z_1 = G^{xy}$ and $Z_0 = G^z$ for random $x, y, z \leftarrow \mathbb{Z}_p$ and random bit b, any p.p.t. algorithm $\mathcal{A}$ decides whether $b = 1$ or 0 with negligible advantage $\text{Adv}^{\mathsf{ddh1}}_{\mathcal{G},\mathcal{A}}(\lambda)$ in λ.

Definition 3. (*Decisional linear assumption in* $\mathbb{G}_1$: DLIN_1) Given $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, $(G_1, G_2, G_3) \leftarrow (\mathbb{G}_1^*)^3$ and (G_1^x, G_2^y, Z_b) where $Z_1 = G_3^{x+y}$ and $Z_0 = G_3^z$ for random $x, y, z \leftarrow \mathbb{Z}_p$ and random bit b, any p.p.t. algorithm $\mathcal{A}$ decides whether $b = 1$ or 0 with negligible advantage $\text{Adv}^{\mathsf{dlin1}}_{\mathcal{G},\mathcal{A}}(\lambda)$ in λ.

For DDH_1 and DLIN_1, we define an analogous assumption in $\mathbb{G}_2$ (DDH_2) by swapping $\mathbb{G}_1$ and $\mathbb{G}_2$ in the respective definitions. In Type-III bilinear groups, it is assumed that both DDH_1 and DDH_2 hold simultaneously. The assumption is called the symmetric external Diffie–Hellman assumption (SXDH), and we define advantage $\text{Adv}^{\mathsf{sxdh}}_{\mathcal{G},\mathcal{C}}$ by $\text{Adv}^{\mathsf{sxdh}}_{\mathcal{G},\mathcal{C}}(\lambda) := \text{Adv}^{\mathsf{ddh1}}_{\mathcal{G},\mathcal{A}}(\lambda) + \text{Adv}^{\mathsf{ddh2}}_{\mathcal{G},\mathcal{B}}(\lambda)$. We extend DLIN in a similar manner:

Definition 4. (*External decision linear assumption in* $\mathbb{G}_1$: XDLIN_1) Given $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, $(G_1, G_2, G_3) \leftarrow (\mathbb{G}_1^*)^3$ and $(G_1^x, G_2^y, \hat{G}_1, \hat{G}_2, \hat{G}_3, \hat{G}_1^x, \hat{G}_2^y, Z_b)$ where $(G_1, G_2, G_3) \sim (\hat{G}_1, \hat{G}_2, \hat{G}_3)$, $Z_1 = G_3^{x+y}$, and $Z_0 = G_3^z$ for random $x, y, z \leftarrow \mathbb{Z}_p$ and random bit b, any p.p.t. algorithm $\mathcal{A}$ decides whether $b = 1$ or 0 with negligible advantage $\text{Adv}^{\mathsf{xdlin1}}_{\mathcal{G},\mathcal{A}}(\lambda)$ in λ.

The XDLIN_1 assumption is equivalent to the DLIN_1 assumption in the generic bilinear group model [13,43] where one can simulate the extra elements, $\hat{G}_1, \hat{G}_2, \hat{G}_3, \hat{G}_1^x, \hat{G}_2^y$, in XDLIN$_1$ from $G_1, G_2, G_3, G_1^x, G_2^y$ in DLIN$_1$. We define the XDLIN$_2$ assumption analogously by giving $\hat{G}_3^{x+y}$ or $\hat{G}_3^z$ as Z_b, to $\mathcal{A}$ instead. Then we define the simultaneous external DLIN assumption, SXDLIN, that assumes that both XDLIN$_1$ and XDLIN$_2$ hold at the same time. By $\mathsf{Adv}^{\mathsf{xdlin2}}_{\mathcal{G},\mathcal{A}}$ ($\mathsf{Adv}^{\mathsf{sxdlin}}_{\mathcal{G},\mathcal{A}}$, resp.), we denote the advantage function for XDLIN$_2$ (and SXDLIN, resp.).

Definition 5. (*Double pairing assumption in* $\mathbb{G}_1$ [4]: DBP$_1$) Given $\Lambda \leftarrow \mathcal{G}(1^\lambda)$ and $(G_z, G_r) \leftarrow (\mathbb{G}_1^*)^2$, any p.p.t. algorithm $\mathcal{A}$ outputs $(Z, R) \in (\mathbb{G}_2^*)^2$ that satisfies $1 = e(G_z, Z)\, e(G_r, R)$ with negligible probability $\mathsf{Adv}^{\mathsf{dbp1}}_{\mathcal{G},\mathcal{A}}(\lambda)$ in λ.

The double pairing assumption in $\mathbb{G}_2$ (DBP$_2$) is defined in the same manner by swapping $\mathbb{G}_1$ and $\mathbb{G}_2$. It is known that DBP$_1$ (DBP$_2$, resp.) is implied by DDH$_1$ (DDH$_2$, resp.) and the reduction is tight [7]. Note that the double pairing assumption does not hold in Type-I groups since $Z = G_r$, $R = G_z^{-1}$ is a trivial solution. Thus in Type-I groups we will instead use the following extension:

Definition 6. (*Simultaneous double pairing assumption* [16]: SDP) Given $\Lambda \leftarrow \mathcal{G}(1^\lambda)$ and $(G_z, G_r, H_z, H_s) \leftarrow (\mathbb{G}^*)^4$, any p.p.t. algorithm $\mathcal{A}$ outputs $(Z, R, S) \in (\mathbb{G}^*)^3$ that satisfies $1 = e(G_z, Z)\, e(G_r, R) \wedge 1 = e(H_z, Z)\, e(H_s, S)$ with negligible probability $\mathsf{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{A}}(\lambda)$ in λ.

As shown in [16], for the Type-I setting the simultaneous double pairing assumption holds relative to $\mathcal{G}$ if the decisional linear assumption holds for $\mathcal{G}$.

3. Definitions

3.1. *Common Setup*

All building blocks make use of a common setup algorithm Setup that takes the security parameter 1^λ and outputs a global parameter gk that is given to all other algorithms. Usually gk consists of a description Λ of a bilinear group setup and a default generator for each group. In this paper, we include several additional generators in gk for technical reasons. Note that when the resulting signature scheme is used in multi-user applications different additional generators need to be assigned to individual users or one needs to fall back on the common reference string model, whereas Λ and the default generators can be shared. Thus we count the size of gk when we assess the efficiency of concrete instantiations. For ease of notation, we make gk implicit except w.r.t. key generation algorithms.

3.2. *Signature Schemes*

We use the following syntax for signature schemes suitable for the multi-user and multi-algorithm setting. We follow standard syntax with the following modifications: the key generation function takes as input global parameter gk generated by Setup (instead of security parameter 1^λ), and the message space $\mathcal{M}$ is determined solely by gk (instead of being determined by the public-key).

Definition 7. (*Signature scheme*) A signature scheme SIG is a triple of polynomial-time algorithms $(\mathsf{Key}, \mathsf{Sign}, \mathsf{Vrf})$:

- $\mathsf{SIG.Key}(gk)$ generates a public-key vk and a secret-key sk.
- $\mathsf{SIG.Sign}(sk, msg)$ takes sk and message msg and outputs a signature σ.
- $\mathsf{SIG.Vrf}(vk, msg, \sigma)$ outputs 1 for acceptance or 0 for rejection.

Correctness requires that $1 = \mathsf{SIG.Vrf}(vk, msg, \sigma)$ holds for any gk generated by Setup, any keys generated as $(vk, sk) \leftarrow \mathsf{SIG.Key}(gk)$, any message $msg \in \mathcal{M}$, and any signature $\sigma \leftarrow \mathsf{SIG.Sign}(sk, msg)$.

Definition 8. (*Unforgeability against adaptive chosen message attacks*) A signature scheme is unforgeable against adaptive chosen message attacks (UF-CMA) if for any probabilistic polynomial-time oracle algorithms $\mathcal{A}$ the following advantage function $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG},\mathcal{A}}$ is bounded by a negligible function in λ.

$$\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{l|l} msg^{\dagger} \notin Q_m \ \wedge & gk \leftarrow \mathsf{Setup}(1^{\lambda}), \\ 1 = \mathsf{SIG.Vrf}(vk, \sigma^{\dagger}, msg^{\dagger}) & (vk, sk) \leftarrow \mathsf{SIG.Key}(gk), \\ & (\sigma^{\dagger}, msg^{\dagger}) \leftarrow \mathcal{A}^{\mathcal{O}_s}(vk) \end{array}\right]$$

$\mathcal{O}_s$ is a signing oracle that, on receiving message msg_j, performs $\sigma_j \leftarrow \mathsf{SIG.Sign}(sk, msg_j)$, returns σ_j to $\mathcal{A}$, and records msg_j to Q_m, which is an initially empty list.

Definition 9. (*Unforgeability against non-adaptive chosen message attacks*) A signature scheme is unforgeable against non-adaptive chosen message attacks (UF-NACMA) if for any probabilistic polynomial-time algorithms $\mathcal{A}$ and any polynomial n in λ, the following advantage function $\mathrm{Adv}^{\mathsf{uf\text{-}nacma}}_{\mathsf{SIG},\mathcal{A}}(\lambda)$ is bounded by a negligible function in λ.

$$\begin{aligned} &\mathrm{Adv}^{\mathsf{uf\text{-}nacma}}_{\mathsf{SIG},\mathcal{A}}(\lambda, n) \\ &:= \Pr\left[\begin{array}{l|l} & gk \leftarrow \mathsf{Setup}(1^{\lambda}), \\ \forall j \in [1, n],\ msg^{\dagger} \neq msg_j \ \wedge & (msg_1, \ldots, msg_n) \leftarrow \mathcal{A}(gk), \\ 1 = \mathsf{SIG.Vrf}(vk, \sigma^{\dagger}, msg^{\dagger}) & (vk, sk) \leftarrow \mathsf{SIG.Key}(gk), \\ & \forall j \in [1, n],\ \sigma_j \leftarrow \mathsf{SIG.Sign}(sk, msg_j), \\ & (\sigma^{\dagger}, msg^{\dagger}) \leftarrow \mathcal{A}(vk, \sigma_1, \ldots, \sigma_n) \end{array}\right] \end{aligned}$$

It is implicit that $\mathcal{A}$ in the first run hands over an internal state to that in the second run.

Definition 10. (*Unforgeability against random message attacks* (UF-RMA) [22]) A signature scheme is unforgeable against random message attacks (UF-RMA) if for any probabilistic polynomial-time algorithms $\mathcal{A}$ and any positive integer n bounded by a polynomial in λ, the following advantage function $\mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{SIG},\mathcal{A}}$ is negligible in λ.

$$\mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l|l} & gk \leftarrow \mathsf{Setup}(1^\lambda), \\ \forall j \in [1,n],\ msg^\dagger \neq msg_j\ \wedge & (vk, sk) \leftarrow \mathsf{SIG.Key}(gk), \\ 1 = \mathsf{SIG.Vrf}(vk, \sigma^\dagger, msg^\dagger) & (msg_1, \ldots, msg_n) \leftarrow \mathcal{M}^n, \\ & \forall j \in [1,n],\ \sigma_j \leftarrow \mathsf{SIG.Sign}(sk, msg_j), \\ & (\sigma^\dagger, msg^\dagger) \leftarrow \mathcal{A}(vk, \sigma_1, msg_1, \ldots, \sigma_n, msg_n) \end{array}\right]$$

We consider a variation of random message attacks where the adversary is given, for example, the random coin used to sample the random message. Our formal definition covers more a general idea of auxiliary information about the message generator as follows. Let MSGGen be a message generation algorithm that takes gk (and random coins as well) as input and outputs $msg \in \mathcal{M}$. Furthermore, MSGGen outputs auxiliary information ω, which may give some hint about the random coins used for selecting msg. The extended random message attack is defined relative to message generator MSGGen as follows.

The above syntax and security notions can be applied to one-time signature schemes by restricting the oracle access only once or parameter n to 1.

Definition 11. [*Unforgeability against extended random message attacks* (UF-XRMA)] A signature scheme is unforgeable against extended random message attacks (UF-XRMA) with respect to message sampler MSGGen if for any probabilistic polynomial-time algorithms $\mathcal{A}$ and any positive integer n bounded by a polynomial in λ, the following advantage function $\mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{SIG},\mathcal{A}}$ is bounded by a negligible function in λ.

$$\mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l|l} & gk \leftarrow \mathsf{Setup}(1^\lambda), \\ & (vk, sk) \leftarrow \mathsf{SIG.Key}(gk), \\ & \forall j \in [1,n], \\ \forall j \in [1,n],\ msg^\dagger \neq msg_j\ \wedge & \quad (msg_j, \omega_j) \leftarrow \mathsf{MSGGen}(gk), \\ 1 = \mathsf{SIG.Vrf}(vk, \sigma^\dagger, msg^\dagger) & \qquad\qquad \sigma_j \leftarrow \mathsf{SIG.Sign}(sk, msg_j), \\ & (\sigma^\dagger, msg^\dagger) \leftarrow \mathcal{A}(vk, \sigma_1, msg_1, \omega_1, \\ & \qquad\qquad \ldots, \sigma_n, msg_n, \omega_n) \end{array}\right]$$

For the above security notions,UF-CMA $\Rightarrow$ UF-XRMA $\Rightarrow$ UF-RMA holds. More precisely, for any signature scheme SIG, for any $\mathcal{A}'$ there exists $\mathcal{A}$ such that $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) \geq \mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{SIG},\mathcal{A}'}(\lambda)$, and for any $\mathcal{A}''$ there exists $\mathcal{A}'$ such that $\mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{SIG},\mathcal{A}'}(\lambda) \geq \mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{SIG},\mathcal{A}''}(\lambda)$.

3.3. *Partial One-Time and Tagged One-Time Signatures*

Partial one-time signatures, also known as two-tier signatures [12], are a variation of one-time signatures where only part of the public-key and secret-key must be updated for every signing, while the remaining part can be persistent.

Definition 12. (*Partial one-time signature scheme* [12]) A partial one-time signature scheme POS is a set of polynomial-time algorithms $\mathsf{POS}.\{\mathsf{Key}, \mathsf{Update}, \mathsf{Sign}, \mathsf{Vrf}\}$.

- $\mathsf{POS.Key}(gk)$ generates a long-term public-key pk and secret-key sk, and sets the associated message space to be $\mathcal{M}_o$ as defined by gk (Recall that we require that $\mathcal{M}_o$ be completely defined by gk).
- $\mathsf{POS.Update}(gk)$ takes gk as input, and outputs a one-time key pair (opk, osk). We denote the space for opk by $\mathcal{K}_{opk}$.
- $\mathsf{POS.Sign}(sk, msg, osk)$ outputs a signature σ on message msg based on sk and osk.
- $\mathsf{POS.Vrf}(pk, opk, msg, \sigma)$ outputs 1 for acceptance, or 0 for rejection.

Correctness requires that $1 = \mathsf{POS.Vrf}(pk, opk, msg, \sigma)$ holds except for negligible probability for any $gk,\ pk,\ opk,\ \sigma$, and $msg \in \mathcal{M}_o$, such that $gk \leftarrow \mathsf{Setup}(1^\lambda)$, $(pk, sk) \leftarrow \mathsf{POS.Key}(gk)$, $(opk, osk) \leftarrow \mathsf{POS.Update}(gk)$, $\sigma \leftarrow \mathsf{POS.Sign}(sk, msg, osk)$.

A tagged one-time signature scheme is a signature scheme whose signing function in addition to the long-term secret-key takes a tag as input. A tag is one-time, i.e., it must be different for every signing.

Definition 13. (*Tagged one-time signature scheme*) A tagged one-time signature scheme TOS is a set of polynomial-time algorithms $\mathsf{TOS}.\{\mathsf{Key}, \mathsf{Tag}, \mathsf{Sign}, \mathsf{Vrf}\}$.

- $\mathsf{TOS.Key}(gk)$ generates a long-term public-key pk and secret-key sk, and sets the associated message space to be $\mathcal{M}_t$ as defined by gk.
- $\mathsf{TOS.Tag}(gk)$ takes gk as input and outputs tag. By $\mathcal{T}$, we denote the space for tag.
- $\mathsf{TOS.Sign}(sk, msg, tag)$ outputs signature σ for message msg based on sk and tag.
- $\mathsf{TOS.Vrf}(pk, tag, msg, \sigma)$ outputs 1 for acceptance, or 0 for rejection.

Correctness requires that $1 = \mathsf{TOS.Vrf}(pk, tag, msg, \sigma)$ holds except for negligible probability for any $gk,\ pk,\ tag,\ \sigma$, and $msg \in \mathcal{M}_t$, such that $gk \leftarrow \mathsf{Setup}(1^\lambda)$, $(pk, sk) \leftarrow \mathsf{TOS.Key}(gk)$, $tag \leftarrow \mathsf{TOS.Tag}(gk)$, $\sigma \leftarrow \mathsf{TOS.Sign}(sk, msg, tag)$.

A TOS scheme is a POS scheme for which $tag = osk = opk$. We can thus give a security notion for POS schemes that also applies to TOS schemes by reading $\mathsf{Update} = \mathsf{Tag}$ and $tag = osk = opk$.

Definition 14. (*Unforgeability against one-time adaptive chosen message attacks*) A partial one-time signature scheme is unforgeable against one-time adaptive chosen message attacks (OT-CMA) if for any probabilistic polynomial-time oracle algorithms $\mathcal{A}$ the following advantage function $\mathrm{Adv}^{\mathsf{ot\text{-}cma}}_{\mathsf{POS},\mathcal{A}}$ is negligible in λ.

$$\mathrm{Adv}^{\mathsf{ot\text{-}cma}}_{\mathsf{POS},\mathcal{A}}(\lambda)$$
$$:= \Pr\left[\begin{array}{l|l} \exists (opk, msg, \sigma) \in Q_m \text{ s.t.} & gk \leftarrow \mathsf{Setup}(1^\lambda), \\ opk^\dagger = opk \ \wedge \ msg^\dagger \neq msg \ \wedge & (pk, sk) \leftarrow \mathsf{POS.Key}(gk), \\ 1 = \mathsf{POS.Vrf}(pk, opk^\dagger, \sigma^\dagger, msg^\dagger) & (opk^\dagger, \sigma^\dagger, msg^\dagger) \leftarrow \mathcal{A}^{\mathcal{O}_t, \mathcal{O}_s}(pk) \end{array}\right]$$

Q_m is initially an empty list. $\mathcal{O}_t$ is the one-time key generation oracle that on receiving a request invokes a fresh session j, performs $(opk_j, osk_j) \leftarrow \mathsf{POS.Update}(gk)$, and returns opk_j. $\mathcal{O}_s$ is the signing oracle that, on receiving a message msg_j for session j, performs $\sigma_j \leftarrow \mathsf{POS.Sign}(sk, msg_j, osk_j)$, returns σ_j to $\mathcal{A}$, and records (opk_j, msg_j, σ_j) to the list Q_m. $\mathcal{O}_s$ works only once for each session. Strong unforgeability is defined by replacing condition $msg^\dagger \neq msg$ with $(msg^\dagger, \sigma^\dagger) \neq (msg, \sigma)$.

We define a non-adaptive variant (OT-NACMA) of the above notion by integrating $\mathcal{O}_t$ into $\mathcal{O}_s$ so that opk_j and σ_j are returned to $\mathcal{A}$ at the same time. Namely, $\mathcal{A}$ must submit msg_j before seeing opk_j. If a scheme is secure in the sense of OT-CMA, the scheme is also secure in the sense of OT-NACMA. By $\mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{POS},\mathcal{A}}(\lambda)$ we denote the advantage of $\mathcal{A}$ in this non-adaptive case. For TOS, we use the same notation, OT-CMA and OT-NACMA, and define advantage functions $\mathrm{Adv}^{\mathsf{ot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}}$ and $\mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{TOS},\mathcal{A}}$ accordingly. We will also consider strong unforgeability, for which we use labels $\mathsf{sot\text{-}cma}$ and $\mathsf{sot\text{-}nacma}$. Recall that if a scheme is strongly unforgeable, it is unforgeable as well.

We define a condition that is relevant for coupling random message secure signature schemes with partial one-time and tagged one-time signature schemes in later sections.

Definition 15. (*Tag/one-time public-key uniformity*) A TOS is called uniform tag if $\mathsf{TOS.Tag}$ outputs tag that is uniformly distributed over tag space $\mathcal{T}$. Similarly, a POS is called uniform-key if $\mathsf{POS.Update}$ outputs opk that is uniformly distributed over key space $\mathcal{K}_{opk}$.

3.4. *Structure-Preserving Signatures*

A signature scheme is structure-preserving over a bilinear group Λ, if public-keys, signatures, and messages are all source group elements of Λ, and the verification only evaluates pairing product equations. Similarly, POS and TOS schemes are structure-preserving if their public-keys, signatures, messages, and tags or one-time public-keys consist of source group elements and the verification only evaluates pairing product equations.

4. Generic Constructions

4.1. $\mathsf{SIG1}$*: Combining Tagged One-Time and RMA-Secure Signatures*

Let rSIG be a signature scheme with message space $\mathcal{M}_\mathsf{r}$, and TOS be a tagged one-time signature scheme with tag space $\mathcal{T}$ such that $\mathcal{M}_\mathsf{r} = \mathcal{T}$ and both schemes use the same Setup. We construct a signature scheme $\mathsf{SIG1}$ from rSIG and TOS. Let gk be the global parameter generated by $\mathsf{Setup}(1^\lambda)$. It is assumed that a secret-key of rSIG includes gk.
[Generic Construction 1: $\mathsf{SIG1}$]

$\mathsf{SIG1.Key}(gk)$: Run $(pk_t, sk_t) \leftarrow \mathsf{TOS.Key}(gk)$, $(vk_r, sk_r) \leftarrow \mathsf{rSIG.Key}(gk)$. Output $vk := (pk_t, vk_r)$ and $sk := (sk_t, sk_r)$.
$\mathsf{SIG1.Sign}(sk, msg)$: Parse sk into (sk_t, sk_r) and take gk from sk_r. Run $tag \leftarrow \mathsf{TOS.Tag}(gk)$, $\sigma_t \leftarrow \mathsf{TOS.Sign}(sk_t, msg, tag)$, $\sigma_r \leftarrow \mathsf{rSIG.Sign}(sk_r, tag)$. Output $\sigma := (tag, \sigma_t, \sigma_r)$.
$\mathsf{SIG1.Vrf}(vk, msg, \sigma)$: Parse vk and σ accordingly. Output 1 if $1 = \mathsf{TOS.Vrf}(pk_t, tag, msg, \sigma_t)$ and $1 = \mathsf{rSIG.Vrf}(vk_r, tag, \sigma_r)$. Output 0 otherwise.

We prove that $\mathsf{SIG1}$ is secure by showing a reduction to the security of each component. As our reductions are efficient in their running time, we only relate success probabilities.

Theorem 1. $\mathsf{SIG1}$ *is UF-CMA if* TOS *is uniform tag and OT-NACMA, and* rSIG *is UF-RMA. In particular, for any p.p.t. algorithm $\mathcal{A}$ there exist p.p.t. algorithms $\mathcal{B}$ and $\mathcal{C}$ such that* $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG1},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{TOS},\mathcal{B}}(\lambda) + \mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{rSIG},\mathcal{C}}(\lambda)$.

Security against random message attacks is sufficient for rSIG as it is used only to sign uniformly chosen tags. To formally prove it, however, we use the important fact that the signing function of TOS does not require any secret behind the tags. Departing from the UF-CMA game for $\mathsf{SIG1}$, the security proof is done by evaluating two game transitions. The first transition is based on the OT-NACMA security of TOS. This part is rather simple as we can construct a simulator in a straightforward manner by following the key generation and signing of rSIG. The second transition is based on UF-RMA of rSIG. We construct a simulator that, given signatures of rSIG on uniformly chosen tags as messages, simulates signatures of $\mathsf{SIG1}$ for messages provided by the adversary. For this to be done, the simulator needs to compute one-time signatures of TOS for the given uniform tags. This, however, can be done without any problem since the simulator has legitimate signing keys that are sufficient to run the signing function of TOS with uniform tags.

Proof. Any signature that is accepted as a successful forgery must either reuse an existing tag, or sign a new tag. We show that the former case reduces to attacking TOS and the latter case reduces to attacking rSIG. Thus the success probability $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG1},\mathcal{A}}(\lambda)$ of an attacker on $\mathsf{SIG1}$ will be bounded by the sum of the success probabilities $\mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{TOS},\mathcal{B}}(\lambda)$ of an attacker on TOS and the success probability $\mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{rSIG},\mathcal{C}}(\lambda)$ of an attacker on rSIG.

Game 0: The actual unforgeability game. $\Pr[\textbf{Game0}] = \mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG1},\mathcal{A}}(\lambda)$.
Game 1: The real security game except that the winning condition is changed to no longer accept repetition of tags.

Lemma 1. $|\Pr[\textbf{Game 0}] - \Pr[\textbf{Game 1}]| \leq \mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{TOS},\mathcal{B}}(\lambda)$

Proof. Attacker $\mathcal{A}$ wins in Game 0, but loses in Game 1, iff it produces a forgery that reuses a tag from a signing query. We describe a reduction $\mathcal{B}$ that uses such an attacker to break the OT-NACMA-security of TOS. The reduction $\mathcal{B}$ receives gk and pk_t from the challenger of TOS, sets up vk_r and sk_r honestly by running $\mathsf{rSIG.Key}(gk)$, and provides gk and $vk = (vk_r, pk_t)$ to $\mathcal{A}$.

To answer a signing query, $\mathcal{B}$ uses the signing oracle of TOS to get tag and σ_t, signs tag using sk_r to produce σ_r, and returns $(tag, \sigma_t, \sigma_r)$. When $\mathcal{A}$ produces a forgery $(tag^\dagger, \sigma_t^\dagger, \sigma_r^\dagger)$ on message $msg^\dagger$, $\mathcal{B}$ outputs $(msg^\dagger, tag^\dagger, \sigma_t^\dagger)$ as a forgery for TOS.

Game 2: The fully idealized game. The winning condition is changed to reject all signatures.

Lemma 2. $|\Pr[\mathbf{Game\ 1}] - Pr[\mathbf{Game\ 2}]| \leq \mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{rSIG},\mathcal{C}}(\lambda)$

Proof. Attacker $\mathcal{A}$ wins in Game 1, iff it produces a forgery with a fresh tag. We describe a reduction algorithm $\mathcal{C}$ that uses $\mathcal{A}$ to break the UF-RMA security of rSIG. Algorithm $\mathcal{C}$ receives gk and vk_r, runs $(pk_t, sk_t) \leftarrow \mathsf{TOS.Key}(gk)$, and provides gk and $vk = (vk_r, pk_t)$ to $\mathcal{A}$.
To answer signing query on message msg, algorithm $\mathcal{C}$ consults $\mathcal{O}_s$ and receives random message $msg_r \leftarrow \mathcal{T}$ and signature σ_r. Algorithm $\mathcal{C}$ then uses msg_r as a tag, i.e., $tag = msg_r$, and creates signature σ_t on msg by running $\mathsf{TOS.Sign}(sk_t, msg, tag)$. It then returns $(tag, \sigma_t, \sigma_r)$. Note that for a uniform tag TOS scheme $\mathsf{TOS.Tag}(gk)$ would generate tags distributed uniformly over the tag space $\mathcal{T}$. Thus the reduction simulation is perfect. When $\mathcal{A}$ produces a forgery $(tag^\dagger, \sigma_t^\dagger, \sigma_r^\dagger)$ on $msg^\dagger$, algorithm $\mathcal{C}$ outputs $(tag^\dagger, \sigma_r^\dagger)$ as a forgery.

Thus $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG1},\mathcal{A}}(\lambda) = \Pr[\mathbf{Game\ 0}] \leq \mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{TOS},\mathcal{B}}(\lambda) + \mathrm{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{rSIG},\mathcal{C}}(\lambda)$ as claimed.
The following theorem is immediately obtained from the construction.

Theorem 2. *If* $\mathsf{TOS.Tag}$ *produces constant-size tags and signatures in the size of input messages, the resulting* $\mathsf{SIG1}$ *produces constant-size signatures as well. Furthermore, if* TOS *and* rSIG *are structure-preserving, so is* $\mathsf{SIG1}$.

4.2. $\mathsf{SIG2}$*: Combining Partial One-Time and XRMA-Secure Signatures*

Let xSIG be a signature scheme with message space $\mathcal{M}_{\mathsf{x}}$, and POS be a partial one-time signature scheme with one-time public-key space $\mathcal{K}_{opk}$ such that $\mathcal{M}_{\mathsf{x}} = \mathcal{K}_{opk}$ and both schemes use the same Setup. We construct a signature scheme $\mathsf{SIG2}$ from xSIG and POS. Let gk be a global parameter generated by $\mathsf{Setup}(1^\lambda)$. It is assumed that a secret-key for xSIG contains gk.
[Generic Construction 2: $\mathsf{SIG2}$]

$\mathsf{SIG2.Key}(gk)$: Run $(pk_p, sk_p) \leftarrow \mathsf{POS.Key}(gk)$, $(vk_x, sk_x) \leftarrow \mathsf{xSIG.Key}(gk)$. Output $vk := (pk_p, vk_x)$ and $sk := (sk_p, sk_x)$.
$\mathsf{SIG2.Sign}(sk, msg)$: Parse sk into (sk_p, sk_x) and take gk from sk_x. Run $(opk, osk) \leftarrow \mathsf{POS.Update}(gk)$, $\sigma_p \leftarrow \mathsf{POS.Sign}(sk_p, msg, osk)$, $\sigma_x \leftarrow \mathsf{xSIG.Sign}(sk_x, opk)$. Output $\sigma := (opk, \sigma_p, \sigma_x)$.
$\mathsf{SIG2.Vrf}(vk, msg, \sigma)$: Parse vk and σ accordingly. Output 1 if $1 = \mathsf{POS.Vrf}(pk_p, opk, msg, \sigma_p)$, and $1 = \mathsf{xSIG.Vrf}(vk_x, opk, \sigma_x)$. Output 0 otherwise.

Theorem 3. $\mathsf{SIG2}$ *is UF-CMA if* POS *is uniform-key and OT-NACMA, and* xSIG *is UF-XRMA relative to* $\mathsf{POS.Update}$ *as a message generator. In particular, for any p.p.t. algorithm* $\mathcal{A}$*, there exist p.p.t. algorithms* $\mathcal{B}$ *and* $\mathcal{C}$ *such that* $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG2},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{ot\text{-}nacma}}_{\mathsf{POS},\mathcal{B}}(\lambda) + \mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{xSIG},\mathcal{C}}(\lambda)$.

Proof. The proof is almost the same as that for Theorem 1. The only difference appears in constructing $\mathcal{C}$ in the second step. Since $\mathsf{POS.Update}$ is used as the extended random message generator, the pair (msg, ω) is in fact (opk, osk). Given (opk, osk), adversary $\mathcal{C}$ can run $\mathsf{POS.Sign}(sk, msg, osk)$ to yield legitimate signatures.

As for our first generic construction, the following theorem holds from the construction.

Theorem 4. *If* POS *produces constant-size one-time public-keys and signatures in the size of input messages, the resulting* SIG2 *produces constant-size signatures as well. Furthermore, if* POS *and* xSIG *are structure-preserving, so is* SIG2.

5. Instantiating SIG1

We instantiate the building blocks TOS and rSIG of our first generic construction to obtain our first SPS scheme. We do so in the Type-I bilinear group setting. The resulting SIG1 scheme is an efficient structure-preserving signature scheme based only on the DLIN assumption.

5.1. *Setup for Type-I Groups*

The following setup procedure is common for all instantiations in this section. The global parameter gk is given to all functions implicitly.

- $\mathsf{Setup}(1^\lambda)$: Run $\Lambda = (p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and pick random generators $(G, C, F, U) \leftarrow (\mathbb{G}^*)^4$. Output $gk := (\Lambda, G, C, F, U)$.

The parameter gk fixes the message space $\mathcal{M}_{\mathsf{r}} := \{(C^m, F^m, U^m) \in \mathbb{G}^3 \mid m \in \mathbb{Z}_p\}$ for the RMA-secure signature scheme presented in Sect. 5.3. For our generic framework to work, the tagged one-time signature schemes should have the same tag space.

5.2. *Tagged One-Time Signature Scheme*

Our scheme generates tags consisting of only one group element, C^t, which is optimally efficient in its size. However, as mentioned above, we need to adjust the tag space to match the message space of rSIG. We thus describe the scheme with a tag in the extended form of (C^t, F^t, U^t). The extended elements F^t and U^t can be dropped when unnecessary.

Our concrete construction of TOS can be seen as an adaptation of a one-time signature scheme in [7] so that it enjoys optimally short one-time public-key (i.e., a tag) with no corresponding one-time secret-key. We note that, given a TOS, one can construct a one-time signature scheme. But the reverse is not known in general.

[Scheme TOS]

$\mathsf{TOS.Key}(gk)$: Parse $gk = (\Lambda, G, C, F, U)$. Choose $w_z, w_r, \mu_z, \mu_s, \tau$ uniformly from $\mathbb{Z}_p^*$ and compute $G_z := G^{w_z}$, $G_r := G^{w_r}$, $H_z := G^{\mu_z}$, $H_s := G^{\mu_s}$, $G_t := G^\tau$ and For $i = 1, \ldots, k$, uniformly choose $\chi_i, \gamma_i, \delta_i$ from $\mathbb{Z}_p$ and compute

$$G_i := G_z^{\chi_i} G_r^{\gamma_i}, \quad \text{and} \quad H_i := H_z^{\chi_i} H_s^{\delta_i}. \tag{1}$$

Output $pk := (G_z, G_r, H_z, H_s, G_t, G_1, \ldots, G_k, H_1, \ldots, H_k) \in \mathbb{G}^{2k+5}$ and $sk := (w_r, \mu_s, \tau, \chi_1, \gamma_1, \delta_1, \ldots, \chi_k, \gamma_k, \delta_k,) \in \mathbb{Z}_p^{3k+5}$.

$\mathsf{TOS.Tag}(gk)$: Choose $t \leftarrow \mathbb{Z}_p^*$, compute $T := C^t$. Output $tag := (T, T', T'') = (C^t, F^t, U^t) \in \mathbb{G}^3$.

$\mathsf{TOS.Sign}(sk, msg, tag)$: Parse msg as $(\tilde{M}_1, \ldots, \tilde{M}_k)$ and tag as (T, T', T''). Parse sk accordingly. Choose $\zeta \leftarrow \mathbb{Z}_p$ and output $\sigma := (\tilde{Z}, \tilde{R}, S) \in \mathbb{G}^3$ where

$$\tilde{Z} := G^{\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i}, \quad \tilde{R} := \left(T^{\tau} G_z^{-\zeta}\right)^{\frac{1}{w_r}} \prod_{i=1}^{k} \tilde{M}_i^{-\gamma_i}, \text{ and } S := \left(H_z^{-\zeta}\right)^{\frac{1}{\mu_s}} \prod_{i=1}^{k} \tilde{M}_i^{-\delta_i}.$$

$\mathsf{TOS.Vrf}(pk, tag, msg, \sigma)$: Parse σ as $(\tilde{Z}, \tilde{R}, S) \in \mathbb{G}^3$, msg as $(\tilde{M}_1, \ldots, \tilde{M}_k) \in \mathbb{G}^k$, and tag as (T, T', T''). Return 1 if the following equations hold. Return 0, otherwise.

$$e(T, G_t) = e\left(G_z, \tilde{Z}\right)\, e\left(G_r, \tilde{R}\right) \prod_{i=1}^{k} e(G_i, \tilde{M}_i) \tag{2}$$

$$1 = e\left(H_z, \tilde{Z}\right)\, e\left(H_s, S\right) \prod_{i=1}^{k} e\left(H_i, \tilde{M}_i\right) \tag{3}$$

Correctness is verified by inspecting the following relations.

$$\text{For (2): } e\left(G_z, G^{\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i}\right) e\left(G_r, \left(T^{\tau} G_z^{-\zeta}\right)^{\frac{1}{w_r}} \prod_{i=1}^{k} \tilde{M}_i^{-\gamma_i}\right) \prod_{i=1}^{k} e\left(G_z^{\chi_i} G_r^{\gamma_i}, \tilde{M}_i\right)$$
$$= e\left(G_z, G^{\zeta}\right)\, e\left(G, T^{\tau}\right) e\left(G, G_z^{-\zeta}\right) = e\left(G, T^{\tau}\right) = e\left(T, G_t\right)$$
$$\text{For (3): } e\left(H_z, G^{\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i}\right) e\left(H_s, \left(H_z^{-\zeta}\right)^{\frac{1}{\mu_s}} \prod_{i=1}^{k} \tilde{M}_i^{-\delta_i}\right) \prod_{i=1}^{k} e\left(H_z^{\chi_i} H_s^{\delta_i}, \tilde{M}_i\right)$$
$$= e\left(H_z, G^{\zeta}\right)\, e\left(G, H_z^{-\zeta}\right) = 1$$

We state the following theorems, of which the first one is immediate from the construction.

Theorem 5. *The above* TOS *is structure-preserving, and yields uniform tags and constant-size signatures.*

Theorem 6. *The above* TOS *is strongly unforgeable against one-time tag adaptive chosen message attacks (SOT-CMA) if the SDP assumption holds. In particular, for all p.p.t. algorithms $\mathcal{A}$, there exists p.p.t. algorithm $\mathcal{B}$ such that* $\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{B}}(\lambda) + 1/p(\lambda)$, *where $p(\lambda)$ is the size of the groups produced by $\mathcal{G}$. Moreover, the run-time overhead of the reduction $\mathcal{B}$ is a small number of multi-exponentiations per signing or tag query.*

Proof. Given successful forger $\mathcal{A}$ against TOS as a black-box, we construct $\mathcal{B}$ that breaks SDP as follows. Let $I_{\mathsf{sdp}} = (\Lambda, G_z, G_r, H_z, H_s)$ be an instance of SDP. Algorithm $\mathcal{B}$ simulates the attack game against TOS as follows. It first builds $gk := (\Lambda, G, C, F, U)$ by choosing G randomly from $\mathbb{G}^*$, choosing $c, f, u \leftarrow \mathbb{Z}_p$, and computing $C = G^c$, $F = G^f$, and $U = G^u$. This yields a gk in the same distribution as produced by Setup. Next $\mathcal{B}$ simulates $\mathsf{TOS.Key}$ by taking (G_z, G_r, H_z, H_s) from I_{sdp} and computing $G_t := H_s^\tau$ for random τ in $\mathbb{Z}_p^*$. It then generates G_i and H_i according to (1). This perfectly simulates $\mathsf{TOS.Key}$.

On receiving the jth query to $\mathcal{O}_t$, algorithm $\mathcal{B}$ computes

$$T := \left(G_z^\zeta G_r^\rho\right)^{\frac{1}{\tau}} \tag{4}$$

for $\zeta, \rho \leftarrow \mathbb{Z}_p^*$. If $T = 1$, $\mathcal{B}$ sets $Z^\star := H_s$, $S^\star := H_z^{-1}$, and $R^\star := (Z^\star)^{\rho/\zeta}$, outputs $(Z^\star, R^\star, S^\star)$ and stops. Otherwise, $\mathcal{B}$ stores (ζ, ρ) and returns $tag_j := (T, T^{f/c}, T^{u/c})$ to $\mathcal{A}$.

On receiving signing query $msg_j = (\tilde{M}_1, \ldots, \tilde{M}_k)$, algorithm $\mathcal{B}$ takes ζ and ρ used for computing tag_j (if tag_j is not yet defined, execute the above procedure for generating tag_j and take new ζ and ρ) and computes

$$\tilde{Z} := H_s^\zeta \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i}, \qquad \tilde{R} := H_s^\rho \prod_{i=1}^{k} \tilde{M}_i^{-\gamma_i}, \quad \text{and} \qquad S := H_z^{-\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\delta_i}. \tag{5}$$

Then $\mathcal{B}$ returns $\sigma_j := (Z, R, S)$ to $\mathcal{A}$ and records (tag_j, σ_j, msg_j).

When $\mathcal{A}$ outputs a forgery $(tag^\dagger, \sigma^\dagger, msg^\dagger)$, algorithm $\mathcal{B}$ searches the records for (tag, σ, msg) such that $tag^\dagger = tag$ and $(msg^\dagger, \sigma^\dagger) \neq (msg, \sigma)$. If no such entry exists, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ computes

$$\tilde{Z}^\star := \frac{\tilde{Z}^\dagger}{\tilde{Z}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^\dagger}{\tilde{M}_i}\right)^{\chi_i}, \quad \tilde{R}^\star := \frac{\tilde{R}^\dagger}{\tilde{R}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^\dagger}{\tilde{M}_i}\right)^{\gamma_i}, \text{ and } S^\star := \frac{S^\dagger}{S} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^\dagger}{\tilde{M}_i}\right)^{\delta_i}$$

where $(\tilde{Z}, \tilde{R}, S)$, $(\tilde{M}_1, \ldots, \tilde{M}_k)$ and their dagger counterparts are taken from (σ, msg) and $(\sigma^\dagger, msg^\dagger)$, respectively. $\mathcal{B}$ finally outputs $(\tilde{Z}^\star, \tilde{R}^\star, S^\star)$ and stops. This completes the description of $\mathcal{B}$.

We claim that $\mathcal{B}$ solves the problem by itself or the view of $\mathcal{A}$ is perfectly simulated. The correctness of key generation has been already inspected. In the simulation of $\mathcal{O}_t$, there is a case of $T = 1$ that happens with probability $1/p$. If it happens, $\mathcal{B}$ outputs a correct answer to I_{sdp}, which is clear by observing $G_z = G_r^{-\rho/\zeta}$, $Z^\star = H_s \neq 1$, $e(G_z, Z^\star)e(G_r, R^\star) = e(G_r^{-\rho/\zeta}, Z^\star)e(G_r, (Z^\star)^{\rho/\zeta}) = 1$ and $e(H_z, Z^\star)e(H_s, S^\star) = e(H_z, H_s)e(H_s, H_z^{-1}) = 1$. Otherwise, tag T is uniformly distributed over $\mathbb{G}^*$ and the simulation is perfect.

Oracle $\mathcal{O}_s$ is simulated perfectly as well. Correctness of simulated $\sigma_j = (\tilde{Z}, \tilde{R}, S)$ can be verified by inspecting the following relations.

$$\begin{aligned}
\text{(Right-hand of (2))} &= e\left(G_z, H_s^{\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i}\right) e\left(G_r, H_s^{\rho} \prod_{i=1}^{k} \tilde{M}_i^{-\gamma_i}\right) \\
&\quad \times \prod_{i=1}^{k} e\left(G_z^{\chi_i} G_r^{\gamma_i}, \tilde{M}_i\right) \\
&= e\left(G_z^{\zeta} G_r^{\rho}, H_s\right) = e\left(\left(G_z^{\zeta} G_r^{\rho}\right)^{\frac{1}{\tau}}, H_s^{\tau}\right) = e(T_1, G_t) \\
\text{(Right-hand of (3))} &= e\left(H_z, H_s^{\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i}\right) e\left(H_s, H_z^{-\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\delta_i}\right) \prod_{i=1}^{k} e(H_z^{\chi_i} H_s^{\delta_i}, \tilde{M}_i) \\
&= e\left(H_z, H_s^{\zeta}\right) \ e\left(H_s, H_z^{-\zeta}\right) \ = 1
\end{aligned}$$

Every $\tilde{Z}$ is uniformly distributed over $\mathbb{G}$ due to the uniform choice of ζ. Then $\tilde{R}$ and S are uniquely determined by following the distribution of $\tilde{Z}$.

Accordingly, $\mathcal{A}$ outputs a successful forgery with non-negligible probability and $\mathcal{B}$ finds a corresponding record (tag, σ, msg). We show that output $(\tilde{Z}^{\star}, \tilde{R}^{\star}, S^{\star})$ from $\mathcal{B}$ is a valid solution to I_{sdp}. First, Eq. (2) is satisfied because

$$\begin{aligned}
1 &= e\left(G_z, \frac{\tilde{Z}^{\dagger}}{\tilde{Z}}\right) e\left(G_r, \frac{\tilde{R}^{\dagger}}{\tilde{R}}\right) \prod_{i=1}^{k} e\left(G_z^{\chi_i} G_r^{\gamma_i}, \frac{\tilde{M}_i^{\dagger}}{\tilde{M}_i}\right) \\
&= e\left(G_z, \frac{\tilde{Z}^{\dagger}}{\tilde{Z}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^{\dagger}}{M_i}\right)^{\chi_i}\right) e\left(G_r, \frac{\tilde{R}^{\dagger}}{\tilde{R}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^{\dagger}}{M_i}\right)^{\gamma_i}\right) \\
&= e\left(G_z, \tilde{Z}^{\star}\right) \ e\left(G_r, \tilde{R}^{\star}\right),
\end{aligned}$$

holds. Equation (3) can be verified similarly.

It remains to prove that $\tilde{Z}^{\star} \neq 1$. Note that, if $msg = msg^{\dagger}$ but this is still a valid forgery then it must be the case that $(\tilde{Z}, \tilde{R}) \neq (\tilde{Z}^{\dagger}, \tilde{R}^{\dagger})$. Since $\tilde{R}$ (resp. $\tilde{R}^{\dagger}$) is uniquely determined by $\tilde{Z}$ and msg (resp. $\tilde{Z}^{\dagger}, msg^{\dagger}$), that would mean that $\tilde{Z}^{\star} \neq 1$. Alternatively, if $msg^{\dagger} \neq msg$, then there exists $\ell \in \{1, \ldots, k\}$ such that $\tilde{M}_{\ell}^{\dagger}/M_{\ell} \neq 1$. We claim that parameters $\chi_1, \ldots, \chi_k$ are independent of the view of $\mathcal{A}$. We prove it by showing that, for every possible assignment to $\chi_1, \ldots, \chi_k$, there exists an assignment to other coins, i.e., $(\gamma_1, \ldots, \gamma_k, \delta_1, \ldots, \delta_k)$ and $(\zeta^{(1)}, \rho^{(1)}, \ldots, \zeta^{(q_s)}, \rho^{(q_s)})$ for q_s queries, that is consistent with the view of $\mathcal{A}$ (By $\zeta^{(j)}$, we denote ζ with respect to the jth query. We follow this convention hereafter. Without loss of generality, we assume that $\mathcal{A}$ makes q_s tag queries and the same number of signing queries). Observe that the view of $\mathcal{A}$ consists of independent group elements $(G, G_z, G_r, H_z, H_s, G_t, G_1, H_1, \ldots, G_k, H_k)$ and $(T_1^{(j)}, \tilde{Z}^{(j)}, \tilde{M}_1^{(j)}, \ldots, \tilde{M}_k^{(j)})$ for $j = 1, \ldots, q_s$ (Note that we omit $\tilde{R}^{(j)}$ and $S^{(j)}$ from the view since they are uniquely determined by the other components). We represent the view by the discrete logarithms of these group elements with respect to base G. Namely, the view is represented by $(1, w_z, w_r, \mu_z, \mu_s, \tau, w_1, \mu_1, \ldots, w_k, \mu_k)$ and $(t^{(j)}, z^{(j)}, m_1^{(j)}, \ldots, m_k^{(j)})$ for $j = 1, \ldots, q_s$. The view and the random coins follow relations from (1), (4), and (5), which translate to

$$w_i = w_z \chi_i + w_r \gamma_i, \quad \mu_i = \mu_z \chi_i + \mu_s \delta_i \quad \text{for } i = 1, \ldots, k, \tag{6}$$

$$\tau t^{(j)} = w_z \zeta^{(j)} + w_r \rho^{(j)}, \text{ and} \tag{7}$$

$$z^{(j)} = \mu_s\, \zeta^{(j)} - \sum_{i=1}^{k} m_i^{(j)} \chi_i \quad \text{for } j = 1, \ldots, q_s. \tag{8}$$

For any $\ell \in \{1, \ldots, k\}$, fix $\chi_1, \ldots, \chi_{\ell-1}, \chi_{\ell+1}, \ldots, \chi_k$, and consider χ_ℓ. For every value of χ_ℓ in $\mathbb{Z}_p$, the linear equations in (6) determine γ_ℓ and δ_ℓ. Then, if $m_\ell^{(j)} \neq 0$, equation (8) determines $\zeta^{(j)}$, and $\rho^{(j)}$ follows from equation (7). If $m_\ell^{(j)} = 0$, then $\zeta^{(j)},\ \rho^{(j)}$ can be assigned independently from χ_ℓ. The above holds for every ℓ in $\{1, \ldots, k\}$. Thus, if $(\chi_1, \ldots, \chi_k)$ is distributed uniformly over $\mathbb{Z}_p^k$, then other coins are distributed uniformly as well and the view of $\mathcal{A}$ is still consistent.

Now we see that given $\mathcal{A}$'s view, $\left(M_\ell^\dagger / M_\ell\right)^{\chi_\ell}$ is distributed uniformly over $\mathbb{G}$ and independent of the other $\{\chi_i\}_{i \neq \ell}$. Therefore $Z^\star = 1$ happens only with probability $1/p$. Thus, $\mathcal{B}$ outputs a valid $(Z^\star, R^\star, S^\star)$ with probability $\mathrm{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{B}} = 1/p + (1 - 1/p)(1 - 1/p)\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}}$, which leads to $\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{TOS},\mathcal{A}} \leq \mathrm{Adv}^{\mathsf{sdp}}_{\mathcal{G},\mathcal{B}} + 1/p$ as claimed. □

Remark 1. The above TOS does not trivially work in the Type-III setting since computing R from T in signing, simulating T using G_r in the reduction, and computing pairing $e(G_r, R)$ in the verification cannot be consistent. In a very recent paper [AGOT14], it is claimed that it can work if some extra group elements are given in public-keys and the underlying assumption, though the resulting scheme would be slightly less efficient than our dedicated construction in the Type-III setting.

Remark 2. The TOS can be used to sign messages of unbounded length by chaining the signatures. Every message block except for the last one is followed by a tag used to sign the next block. The signature consists of all internal signatures and tags. The initial tag is considered as the tag for the entire signature. For a message consisting of m group elements, it repeats $\tau := 1 + \max(0, \lceil \frac{m-k}{k-1} \rceil)$ times and the resulting signature consists of $4\tau - 1$ elements.

5.3. *RMA-Secure Signature Scheme*

To sign random group elements, we will use a construction based on the dual system signature scheme of Waters [44]. For readers unfamiliar with Waters' scheme we recall it in "Appendix." Our intuition for making the original scheme structure-preserving is as follows. While the original scheme is CMA-secure under the DLIN assumption, the security proof makes use of a trapdoor commitment to elements in $\mathbb{Z}_p$ and consequently messages are elements in $\mathbb{Z}_p$ rather than $\mathbb{G}$. Our construction below resorts to RMA-security and removes this commitment to allow messages to be a sequence of random group elements satisfying a particular relation. Concretely, the message space $\mathcal{M}_{\mathsf{x}} := \{(C^m, F^m, U^m) \in \mathbb{G}^3 \mid m \in \mathbb{Z}_p\}$ is defined by generators (C, F, U) in gk. Moreover, the tag elements of Waters' scheme are removed in our RMA-secure scheme as they were primarily required for (*adaptive*) CMA-security.

Other minor modifications are needed for the structure-preserving property. We modify the verification algorithm. Our verification algorithm is deterministic and uses five verification equations. Two equations are for signature elements that are not related to the message part—this is a consequence of deterministic verification. Three equations are for the (extended) message part. We also slightly modify the verification key. One element in $\mathbb{G}_T$ is divided into two elements of $\mathbb{G}$ via randomization due to the requirement of SPS.

[Scheme rSIG]

$\mathsf{rSIG.Key}(gk)$: Given $gk := (\Lambda, G, C, F, U)$ as input, uniformly select V, V_1, V_2, H from $\mathbb{G}^*$ and a_1, a_2, b, α, and ρ from $\mathbb{Z}_p^*$. Then compute and output $vk := (B, A_1, A_2, B_1, B_2, R_1, R_2, W_1, W_2, H, X_1, X_2)$ and $sk := (vk, K_1, K_2, V, V_1, V_2)$ where

$$\begin{array}{lllll} B := G^b, & A_1 := G^{a_1}, & A_2 := G^{a_2}, & B_1 := G^{b\cdot a_1}, & B_2 := G^{b\cdot a_2} \\ R_1 := V V_1^{a_1}, & R_2 := V V_2^{a_2}, & W_1 := R_1^b, & W_2 := R_2^b, & \\ X_1 := G^{\rho}, & X_2 := G^{\alpha\cdot a_1\cdot b/\rho}, & K_1 := G^{\alpha}, & K_2 := G^{\alpha\cdot a_1}. & \end{array}$$

$\mathsf{rSIG.Sign}(sk, msg)$: Parse msg into (M_1, M_2, M_3). Pick random $r_1, r_2, z_1, z_2 \in \mathbb{Z}_p$. Let $r = r_1 + r_2$. Compute and output signature $\sigma := (S_0, S_1, \ldots S_7)$ where

$$\begin{array}{llll} S_0 := (M_3 H)^{r_1}, & S_1 := K_2 V^r, & S_2 := K_1^{-1} V_1^r G^{z_1}, & S_3 := B^{-z_1}, \\ S_4 := V_2^r G^{z_2}, & S_5 := B^{-z_2}, & S_6 := B^{r_2}, & S_7 := G^{r_1}. \end{array}$$

$\mathsf{rSIG.Vrf}(vk, \sigma, msg)$: Parse msg into (M_1, M_2, M_3) and σ into $(S_0, S_1, \ldots, S_7)$. Also parse vk accordingly. Verify the following pairing product equations:

$$e(S_1, B)\, e(S_2, B_1)\, e(S_3, A_1) = e(S_6, R_1)\, e(S_7, W_1), \tag{9}$$

$$e(S_1, B)\, e(S_4, B_2)\, e(S_5, A_2) = e(S_6, R_2)\, e(S_7, W_2)\, e(X_1, X_2), \tag{10}$$

$$e(S_7, M_3 H) = e(G, S_0), \tag{11}$$

$$e(F, M_1) = e(C, M_2), \tag{12}$$

$$e(U, M_1) = e(C, M_3). \tag{13}$$

The scheme is structure-preserving by construction, and the correctness is easily verified as follows.

$$\begin{aligned} (\text{Left-hand of (9)}) &= e\left(G^{\alpha a_1} V^r, G^b\right) e\left(G^{-\alpha} V_1^r G^{z_1}, G^{b a_1}\right) e\left(G^{-b z_1}, G^{a_1}\right) \\ &= e\,(G, V)^{br}\, e\,(G, V_1)^{b a_1 r} \\ &= e\,(G, V)^{b(r_1+r_2)}\, e\,(G, V_1)^{b a_1 (r_1+r_2)} \\ &= e\left(G^{b r_2}, V V_1^{a_1}\right) e\left(G^{r_1}, V^b V_1^{b a_1}\right) \end{aligned}$$

$$\begin{aligned}
&= (\text{Right-hand of (9)}) \\
(\text{Left-hand of (10)}) &= e\left(G^{\alpha a_1} V^r, G^b\right) e\left(V_2^r G^{z_2}, G^{ba_2}\right) e\left(G^{-bz_2}, G^{a_2}\right) \\
&= e(G, G)^{\alpha b a_1} e(G, V)^{br} e(G, V_2)^{ba_2 r} \\
&= e(G, V)^{b(r_1+r_2)} e(G, V_2)^{ba_2(r_1+r_2)} e(G, G)^{\alpha b a_1} \\
&= e\left(G^{br_2}, V V_2^{a_2}\right) e\left(G^{r_1}, V^b V_2^{ba_2}\right) e\left(G^{\rho}, G^{\alpha b a_1/\rho}\right) \\
&= (\text{Right-hand of (10)})
\end{aligned}$$

Equations (9) and (10) hold since $r = r_1 + r_2$. The followings also hold.

$$\begin{aligned}
&(\text{Left-hand of (11)}) = e(G^{r_1}, U^m H) = e(G, U^m H)^{r_1} = e(G, (U^m H)^{r_1}) \\
&\quad = (\text{Right-hand of (11)}), \\
&(\text{Left-hand of (12)}) = e(F, C^m) = e(F, C)^m = e(C, F^m) = (\text{Right-hand of (12)}), \\
&(\text{Left-hand of (13)}) = e(U, C^m) = e(U, C)^m = e(C, U^m) = (\text{Right-hand of (13)}).
\end{aligned}$$

Theorem 7. *The above* rSIG *scheme is UF-RMA under the DLIN assumption. In particular, for any p.p.t. algorithm $\mathcal{A}$ against* rSIG *that makes at most $q_s(\lambda)$ signing queries, there exists p.p.t. algorithm $\mathcal{B}$ for DLIN such that* $\text{Adv}^{\mathsf{uf\text{-}rma}}_{\mathsf{rSIG},\mathcal{A}}(\lambda) \leq (q_s(\lambda) + 2) \cdot \text{Adv}^{\mathsf{dlin}}_{\mathcal{G},\mathcal{B}}(\lambda)$.

Proof. We refer to the signatures output by the signing algorithm as *normal signatures*. In the proof we will consider an additional type of signatures which we refer to as *simulation-type signatures*; these will be computationally indistinguishable but easier to simulate. For $\gamma \in \mathbb{Z}_p$, simulation-type signatures are of the form $\sigma = (S_0, S_1' = S_1 \cdot G^{-a_1 a_2 \gamma}, S_2' = S_2 \cdot G^{a_2 \gamma}, S_3, S_4' = S_4 \cdot G^{a_1 \gamma}, S_5, \ldots, S_7)$ where $(S_0, \ldots, S_7)$ is a normal signature. We give the outline of the proof using some lemmas. Proofs for the lemmas are given after the outline.

Lemma 3. *Any signature that is accepted by the verification algorithm must be either a normal-type signature or a simulation-type signature.*

To prove this lemma, we introduced two verification equations for signature elements that are not related to a message. We consider a sequence of games. Let p_i be the probability that the adversary succeeds in **Game i**, and $p_i^{\text{norm}}(\lambda)$ and $p_i^{\text{sim}}(\lambda)$ that he succeeds with a normal-type, or simulation-type forgery, respectively. Then by Lemma 3, $p_i(\lambda) = p_i^{\text{norm}}(\lambda) + p_i^{\text{sim}}(\lambda)$ for all i.

Game 0: The actual unforgeability under random message attacks game.

Lemma 4. *There exists an adversary $\mathcal{B}_1$ such that $p_0^{sim}(\lambda) \leq \text{Adv}^{\mathsf{dlin}}_{\mathcal{G},\mathcal{B}_1}(\lambda)$.*

Game i: The real security game except that the first i signatures that are given by the oracle are simulation-type signatures.

Lemma 5. *There exists an adversary $\mathcal{B}_2$ such that* $|p_{i-1}^{norm}(\lambda) - p_i^{norm}(\lambda)| \le \mathsf{Adv}_{\mathcal{G},\mathcal{B}_2}^{\mathsf{dlin}}(\lambda)$.

Game q: All signatures given by the oracle are simulation-type signatures.

Lemma 6. *There exists an adversary $\mathcal{B}_3$ such that* $p_q^{norm}(\lambda) \le \mathsf{Adv}_{\mathcal{G},\mathcal{B}_3}^{\mathsf{cdh}}(\lambda)$.

We have shown that in **Game q**, $\mathcal{A}$ can output a normal-type forgery with at most negligible probability. Thus, by Lemma 5 we can conclude that the same is true in **Game 0** and it holds that

$$\begin{aligned}\mathsf{Adv}_{\mathsf{rSIG},\mathcal{A}}^{\mathsf{uf\text{-}rma}}(\lambda) &= p_0(\lambda) = p_0^{\mathrm{sim}}(\lambda) + p_0^{\mathrm{norm}}(\lambda) \le p_0^{\mathrm{sim}}(\lambda) + \sum_{i=1}^{q} |p_{i-1}^{\mathrm{norm}}(\lambda) - p_i^{\mathrm{norm}}(\lambda)| \\ &\quad + p_q^{\mathrm{norm}}(\lambda) \\ &\le \mathsf{Adv}_{\mathcal{G},\mathcal{B}_1}^{\mathsf{dlin}}(\lambda) + q\mathsf{Adv}_{\mathcal{G},\mathcal{B}_2}^{\mathsf{dlin}}(\lambda) + \mathsf{Adv}_{\mathcal{G},\mathcal{B}_3}^{\mathsf{cdh}}(\lambda) \le (q+2)\cdot \mathsf{Adv}_{\mathcal{G},\mathcal{B}}^{\mathsf{dlin}}(\lambda).\end{aligned}$$

Proof of Lemma 3. We have to show that only normal and simulation-type signatures can fulfil these equations. We ignore verification equations (12) and (13) that establish that *msg* is well formed. A signature has four random exponents, r_1, r_2, z_1, z_2. A simulation-type signature has additional exponent γ.

We interpret S_7 as G^{r_1}, and it follows from verification equation (11) that S_0 is $(M_3 H)^{r_1}$. We interpret S_3 as G^{-bz_1}, S_5 as G^{-bz_2}, and S_6 as $G^{r_2 b}$. Now we have fixed all exponents of a normal signature. The remaining two verification equations tell us that

$$\begin{aligned} e\left(G^b, S_1\right)\cdot e\left(G^{ba_1}, S_2\right) &= e\left(VV_1^{a_1}, G^{r_2 b}\right)\cdot e\left(\left(VV_1^{a_1}\right)^b, G^{r_1}\right)\cdot e\left(G^{a_1}, G^{bz_1}\right),\\ e\left(G^b, S_1\right)\cdot e\left(G^{ba_2}, S_4\right) &= e\left(VV_2^{a_2}, G^{r_2 b}\right)\cdot e\left(\left(VV_2^{a_2}\right)^b, G^{r_1}\right)\cdot e\left(G^{a_2}, G^{bz_2}\right)\\ &\quad \cdot e\left(G, G\right)^{\alpha a_1 b}.\end{aligned}$$

We interpret S_1 as $G^{\alpha\cdot a_1} V^r G^{-a_1 a_2 \gamma}$. Now we have two equations and two unknowns that fix S_2 to $G^{-\alpha} V_1^r G^{z_1} G^{a_2\gamma}$ and S_4 to $V_2^r G^{z_2} G^{a_1\gamma}$, respectively. If $\gamma = 0$ we have a normal signature, otherwise we have a simulation-type signature.

Proof of Lemma 4. Suppose for contradiction that there is an adversary $\mathcal{A}$, which, when playing **Game 0** (and thus receiving only normal signatures), produces forgeries which are formed like simulation-type signatures. Then we can construct an adversary $\mathcal{B}_1$ for DLIN as follows.

Let $I_{\mathsf{dlin}} = (\Lambda, G_1, G_2, G_3, X, Y, Z)$ be an instance of DLIN where $\Lambda = (p, \mathbb{G}, \mathbb{G}_T, e)$ is a Type-I bilinear group setting and G_1, G_2, G_3 are randomly taken from $\mathbb{G}^*$ and there exist random $x, y, z \in \mathbb{Z}_p$ such that $X = G_1^x$, $Y = G_2^y$ and $Z = G_3^z$ or G_3^{x+y}. Given I_{dlin}, adversary $\mathcal{B}_1$ works as follows. It first sets $G := G_3$ and chooses C, F, U at random from $\mathbb{G}^*$, and then sets them into gk. Next, it chooses $v, v_1, v_2 \in \mathbb{Z}_p^*$ and computes $V := G_3^v$, $V_1 := G_3^{v_1}$, and $V_2 := G_3^{v_2}$ (This way we know the discrete log of these values w.r.t. G_3). Then it chooses random $H \in \mathbb{G}^*$, $b, \alpha, \rho \in \mathbb{Z}_p^*$ and compute:

$$\begin{array}{llll} B := G_3^b, & & & \\ A_1 := G_1, & A_2 := G_2, & B_1 := G_1^b, & B_2 := G_2^b \\ R_1 := VV_1^{a_1} = G_3^v G_1^{v_1}, & R_2 := VV_2^{a_2} = G_3^v G_2^{v_2}, & W_1 := R_1^b = (G_3^v G_1^{v_1})^b, & W_2 := R_2^b = (G_3^v G_2^{v_2})^b, \\ X_1 := G_3^{\rho}, & X_2 := G^{\alpha \cdot a_1 \cdot b/\rho} = G_1^{\alpha b/\rho}, & K_1 := G_3^{\alpha}, & K_2 := G^{\alpha \cdot a_1} = G_1^{\alpha}. \end{array}$$

and sets them into vk and sk, accordingly. Note that both the distribution of the public- and secret-keys are statistically close to that in the real DLIN game. Moreover, to sign random messages, $\mathcal{B}_1$ can follow the real signing algorithm by using sk.

Suppose that $\mathcal{A}$ produces a valid forgery $\sigma^{\dagger}$ and $msg^{\dagger}$. Then $\mathcal{B}_1$ proceeds as follows. It parses $\sigma^{\dagger}$ as $(S_0, \ldots, S_7)$. By Lemma 3, it is shown that if the verification equations hold, then it must hold that $S_1 = G^{\alpha a_1} V^r G^{-a_1 a_2 \gamma}$, $S_2 = G^{-\alpha} V_1^r G^{z_1} G^{a_2 \gamma}$, and $S_4 = V_2^r G^{z_2} G^{a_1 \gamma}$. If this is a simulation-type signature, it holds that $\gamma \neq 0$. According to our choice of public-key, we can rewrite $S_1 = G_1^{\alpha} V^r G_2^{-f\gamma}$, $S_2 = G_3^{-\alpha} V_1^r G_3^{z_1} G_2^{\gamma}$, and $S_4 = V_2^r G_3^{z_2} G_1^{\gamma}$, where f is the discrete log of G_1 w.r.t. G_3. Thus, if $\mathcal{B}_1$ can extract $G_2^{-f\gamma}, G_2^{\gamma}, G_1^{\gamma}$, it can easily break the DLIN instance by testing whether $1 = e(Z, G_2^{-f\gamma}) \cdot e(G_2^{\gamma}, X) e(G_1^{\gamma}, Y)$. $\mathcal{B}_1$ can extract such values because the signature includes $S_3 = G_3^{-bz_1}$, $S_5 = G_3^{-bz_2}$, $S_6 = G_3^{br_2}$, and $S_7 = G_3^{r_1}$, and it has b, α and the discrete logarithms of V, V_1, V_2 w.r.t. G_3. Thus, it will be straightforward to extract the above values.

Proof of Lemma 5. Suppose for contradiction that there exists an adversary $\mathcal{A}$ such that the probabilities that $\mathcal{A}$ outputs a normal-type forgery in Game i and Game $i+1$ differ by a non-negligible amount. Then we will use $\mathcal{A}$ to construct an algorithm $\mathcal{B}_2$ that breaks the DLIN assumption.

$\mathcal{B}_2$ is given an instance of DLIN; $I_{\mathsf{dlin}} = (\Lambda, G_1, G_2, G_3, X, Y, Z)$. Note that determining whether a signature is of normal type or simulation type naturally corresponds to a DLIN problem: each signature contains $S_7 = G^{r_1}$, $S_6 = (G^b)^{r_2}$, and S_1 which will include $V^{r_1+r_2}$ or $V^{r_1+r_2} G^{-a_1 a_2 \gamma}$ depending on whether this is a normal- or simulation-type signature (Recall that we define $r = r_1 + r_2$). If $\mathcal{B}_2$ sets $G = G_2$, $G^b = G_1$, and $V = G_3$, then it seems fairly straightforward to argue based on the DLIN assumption that it will be impossible for the adversary to distinguish normal and simulation-type signatures. However, $\mathcal{B}_2$ cannot tell whether $\mathcal{A}$'s forgery is normal type or simulation type in this simulation. Thus, there will be no way for $\mathcal{B}_2$ to take advantage of a change in $\mathcal{A}$'s success probability to solve the DLIN challenge.

The solution is to set things up so that, with high probability $\mathcal{B}_2$ can take S_0 from the adversary's forgery and extract something that looks like $G_3^{r_1}$ (which will allow $\mathcal{B}_2$ to distinguish DLIN tuples and consequently detect simulation-type signatures), but at the same time it is guaranteed that for the ith message, the G_3 component of S_0 will cancel out, leaving only an $G_2^{r_1}$ component which will not allow the challenger itself to know whether a simulated signature is normal type or simulation type.

More specifically, the idea will be to choose some secret values ξ, β, χ, η and embed them in the parameters so that for message (C^w, F^w, U^w) we get $U^w H = G_2^{\chi w + \eta} G_3^{\xi w + \beta}$. Then $S_0 = (U^w H)^{r_1} = G_2^{(\chi w + \eta) r_1} G_3^{(\xi w + \beta) r_1}$. If $\xi w + \beta \neq 0$, this gives useful information on $G_3^{r_1}$ (in particular it will allow $\mathcal{B}_2$ to test candidate values), while if $\xi w + \beta = 0$, this has no G_3 component and thus doesn't help at all with finding $G_3^{r_1}$. $\mathcal{B}_2$ chooses

ξ, β so that $\xi w + \beta = 0$ for the w used to generate the ith message. Furthermore, it will be guaranteed that ξ, β are information theoretically hidden even given w, so the adversary has only negligible chance of producing another message with U^{w^*} such that $\xi w^* + \beta = 0$ as well.

Now we show details of the algorithm for $\mathcal{B}_2$. First of all, $\mathcal{B}_2$ sets up the message space and generates the public-key in the following manner. $\mathcal{B}_2$ sets (C, F), used to define message space $\mathcal{M}$, to (G_1^{φ}, G_3) by choosing random $\varphi \leftarrow \mathbb{Z}_p^*$. It chooses random $\xi, \beta, \chi, \eta \leftarrow \mathbb{Z}_p^*$, and computes $U := G_2^{\chi} G_3^{\xi}$, and $H := G_2^{\eta} G_3^{\beta}$. These values will be uniformly distributed, and independent of ξ, β. $\mathcal{B}_2$ then sets

$$gk = (\Lambda, G, C, F, U) := \left(\Lambda, G_2, G_1^{\varphi}, G_3, G_2^{\chi} G_3^{\xi}\right)$$

$\mathcal{B}_2$ also sets $B := G_1$, and chooses V, V_1, V_2. It must choose these values carefully so that it can compute both W_i and W_i^b, and at the same time so that the component V^r of a signature value S_1 gives $\mathcal{B}_2$ some useful information (in particular it will allow $\mathcal{B}_2$ to derive G_3^r). It does this by choosing $v_1, v_2, \delta \leftarrow \mathbb{Z}_p^*$, and computing $V := G_3^{-a_1 a_2 \delta}$, $V_1 := G_2^{v_1} G_3^{a_2 \delta}$, and $V_2 := G_2^{v_2} G_3^{a_1 \delta}$.

Next, $\mathcal{B}_2$ chooses $a_1, a_2, \alpha, \rho \leftarrow \mathbb{Z}_p^*$ and computes

$$\begin{array}{llll}
B := G_1, & & & \\
A_1 := G_2^{a_1}, & A_2 := G_2^{a_2}, & B_1 := G_1^{a_1}, & B_2 := G_1^{a_2} \\
R_1 := V V_1^{a_1} = G_2^{a_1 v_1}, & R_2 := V V_2^{a_2} = G_2^{a_2 v_2}, & W_1 := R_1^b = G_1^{a_1 v_1}, & W_2 := R_2^b = G_1^{a_2 v_2}, \\
X_1 := G_2^{\rho}, & X_2 := G_1^{\alpha a_1/\rho}, & K_1 := G_2^{\alpha}, & K_2 := G_2^{\alpha a_1},
\end{array}$$

and sets them into vk and sk, accordingly. Note that both of these tuples are distributed statistically close to those produced by Setup and rSIG.Key.

Next $\mathcal{B}_2$ simulates signatures for the jth random message as follows.

Case $j < i$: It chooses w_j at random and computes $(M_1, M_2, M_3) = (C^{w_j}, F^{w_j}, U^{w_j})$. It can compute a simulation-type signatures for this message since it has sk and $G^{a_1 a_2} = G_2^{a_1 a_2}$.

Case $j = i$: It chooses w such that $\xi w + \beta = 0$ and computes $(M_1, M_2, M_3) = (C^w, F^w, U^w)$. Note that since no information about ξ, β is revealed this message will look appropriately random to the adversary. It will implicitly hold that $r_1 = y$ and $r_2 = x$. $\mathcal{B}_2$ computes $S_6 = G^{b r_2} = G_1^x = X$ and $S_7 = G^{r_1} = G_2^y = Y$. Recall that it chose U, H such that $U^w H = G_2^{\chi w + \eta}$. Thus, $\mathcal{B}_2$ can compute $S_0 = (M_3 H)^{r_1} = Y^{\chi w + \eta}$.

What remains is to compute S_1, S_2, S_4. Note that this involves computing V^r, V_1^r, and V_2^r, respectively. This is where $\mathcal{B}_2$ will embed its challenge. Recall that $V = G_3^{-a_1 a_2 \delta}$. Thus, it will compute $V^r = (G_3^{r_1 + r_2})^{-a_1 a_2 \delta}$ as $Z^{-a_1 a_2 \delta}$. If $Z = G_3^{x+y}$ this will be correct; if $Z = G_3^z$ for random z, then there will be an extra factor of $G_3^{-a_1 a_2 \delta (z - (x+y))}$. If $\mathcal{B}_2$ lets $G^{\gamma} = G_3^{\delta(z-(x+y))}$ (which is uniformly random from the adversary's point of view), then this is distributed exactly as it should be in a simulation-type signature. Thus, $\mathcal{B}_2$ computes S_1 which should be either $G^{\alpha a_1} V^r$ or $G^{\alpha a_1} V^r G^{-a_1 a_2 \gamma}$ as $G_2^{\alpha a_1} Z^{-a_1 a_2 \delta}$.

$\mathcal{B}_2$ can try to apply the same approach to compute V_1^r to get S_2. However, recall that $\mathcal{B}_2$ sets $V_1 = G_2^{v_1} G_3^{a_2\delta}$. Thus, computing V_1^r involves computing G_2^r, which $\mathcal{B}_2$ cannot do (If it could it could use that to break the DLIN assumption). To get around this, $\mathcal{B}_2$ uses z_1, z_2. It chooses random s_1, s_2 and implicitly sets $G^{z_1} = G_2^{-v_1 r_2 + s_1}$ and $G^{z_2} = G_2^{-v_2 r_2 + s_2}$. While it cannot compute these values, it can compute $G^{-z_1 b} = G_1^{v_1 r_2 - s_1} = X^{v_1} G_1^{-s_1}$ and $G^{-z_2 b} = X^{v_2} G_1^{-s_2}$. Then to generate S_2, $\mathcal{B}_2$ can compute

$$\begin{aligned} G_2^{-\alpha} Y^{v_1} Z^{a_2\delta} G_2^{s_1} &= G^{-\alpha} G_2^{r_1 v_1} Z^{a_2\delta} G_2^{s_1} G_2^{r_2 v_1} G_2^{-r_2 v_1} \\ &= G^{-\alpha} G_2^{(r_1+r_2)v_1} Z^{a_2\delta} G_2^{s_1 - r_2 v_1} \\ &= G^{-\alpha} G_2^{r v_1} Z^{a_2\delta} G^{z_1}. \end{aligned}$$

If $Z = G_3^{x+y} = G_3^r$, then this will be

$$\begin{aligned} G^{-\alpha} G_2^{r v_1} G_3^{r a_2\delta} G^{z_1} &= G^{-\alpha} \left(G_2^{v_1} G_3^{a_2\delta} \right)^r G^{z_1} \\ &= G^{-\alpha} V_1^r G^{z_1}. \end{aligned}$$

If $Z = G_3^{z \neq x+y}$, then this will be

$$\begin{aligned} G^{-\alpha} G_2^{r v_1} G_3^{z a_2\delta} G^{z_1} &= G^{-\alpha} G_2^{r v_1} G_3^{r a_2\delta} G_3^{a_2\delta(z-(x+y))} G^{z_1} \\ &= G^{-\alpha} G_2^{r v_1} G_3^{r a_2\delta} G^{a_2\gamma} G^{z_1} \\ &= G^{-\alpha} V_1^r G^{a_2\gamma} G^{z_1} \end{aligned}$$

where the second to last equality follows from our choice of γ above. By a similar argument, $\mathcal{B}_2$ computes S_4 as $Y^{v_2} Z^{a_1\delta} G_2^{s_2}$ and this will be either $V_2^r G^{z_2}$ or $V_2^r G^{z_2} G^{a_1\gamma}$ as desired. $\mathcal{B}_2$ sets $S := (S_0, S_1, S_2, S_3, S_4, S_5, S_6, S_7)$ where

$$\begin{array}{lll} S_0 = Y^{\chi w_i + \eta} & S_1 = G_2^{\alpha a_1} Z^{-a_1 a_2 \delta} & S_2 = G_2^{-\alpha} Y^{v_1} Z^{a_2\delta} G_2^{s_1} \\ S_3 = X^{v_1} G_1^{-s_1} & S_4 = Y^{v_2} Z^{a_1\delta} G_2^{s_2} & S_5 = X^{v_2} G_1^{-s_2} \\ S_6 = X & S_7 = Y. & \end{array}$$

Case $j > i$: It chooses w and computes $m_j = (M_1, M_2, M_3) = (C^w, F^w, U^w)$ and a signature σ according to $\mathsf{rSIG.Sign}(sk, m_j)$. It outputs σ, m_j.

On receiving forgery $S = (S_0, S_1, \ldots, S_7)$ and $(M_1, M_2, M_3) = (C^w, F^w, U^w)$ for some message w, $\mathcal{B}_2$ outputs 1 if and only if

$$e\left(S_0, G_1\right) \cdot e\left(M_2^{\xi} G_3^{\beta}, S_6\right)$$
$$= e\left(\left(S_1 G_2^{-\alpha a_1}\right)^{1/(-a_1 a_2 \delta)}, \left(M_1^{1/\varphi}\right)^{\xi} G_1^{\beta}\right) \cdot e\left(S_7, \left(M_1^{1/\varphi}\right)^{\chi} G_1^{\eta}\right).$$

By Lemma 3, we are guaranteed that if the signature S verifies, then there must exist w, r_1, r_2, γ such that $S_0 = (U^w H)^{r_1}$, $S_1 = G^{\alpha a_1} V^r G^{-a_1 a_2 \gamma}$, $S_6 = G^{b r_2}$, and $S_7 = G^{r_1}$ where $r = r_1 + r_2$. We are also guaranteed that $M_1 = (G_1^{\varphi})^w$ and $M_2 = G_3^w$.

Rephrased in terms of our parameters, this means

$$S_0 = (G_2^{\chi w + \eta} G_3^{\xi w + \beta})^{r_1} \qquad S_1 = G_2^{\alpha a_1} G_3^{-a_1 a_2 \delta r} G_2^{-a_1 a_2 \gamma}$$
$$S_6 = G_1^{r_2} \qquad S_7 = G_2^{r_1}.$$

Plugging this into the above computation we get that $\mathcal{B}_2$ will output 1 if and only if

$$e\left(\left(G_2^{\chi w+\eta} G_3^{\xi w+\beta}\right)^{r_1}, G_1\right) \cdot e\left(\left(G_3^{w}\right)^{\xi} G_3^{\beta}, G_1^{r_2}\right)$$
$$= e\left(\left(G_2^{\alpha a_1} G_3^{-a_1 a_2 \delta r} G_2^{-a_1 a_2 \gamma} G_2^{-\alpha a_1}\right)^{1/(-a_1 a_2 \delta)}, \left(G_1^{w}\right)^{\xi} G_1^{\beta}\right) \cdot e\left(G_2^{r_1}, (G_1^{w})^{\chi} G_1^{\eta}\right).$$

Simplifying the left side to

$$e\left(\left(G_2^{\chi w+\eta} G_3^{\xi w+\beta}\right)^{r_1}, G_1\right) \cdot e\left(G_3^{\xi w+\beta}, G_1^{r_2}\right)$$
$$= e\left(G_2, G_1\right)^{(\chi w+\eta) r_1} \cdot e(G_3, G_1)^{(\xi w+\beta) r_1} \cdot e(G_3, G_1)^{(\xi w+\beta) r_2}$$
$$= e(G_2, G_1)^{(\chi w+\eta) r_1} \cdot e(G_3, G_1)^{(\xi w+\beta) r}$$

and the right side to

$$e\left(\left(G_3^{-a_1 a_2 \delta r} G_2^{-a_1 a_2 \gamma}\right)^{1/(-a_1 a_2 \delta)}, G_1^{\xi w+\beta}\right) \cdot e\left(G_2^{r_1}, G_1^{\chi w+\eta}\right)$$
$$= e\left(G_3^{r} G_2^{\gamma/\delta}, G_1^{\xi w+\beta}\right) \cdot e\left(G_2^{r_1}, G_1^{\chi w+\eta}\right)$$
$$= e(G_2, G_1)^{(\chi w+\eta) r_1} \cdot e(G_3, G_1)^{(\xi w+\beta) r} \cdot e(G_2, G_1)^{(\gamma/\delta)(\xi w+\beta)}$$

and by dividing out all the pairings of the left side we obtain the simplified equation

$$1 = e(G_2, G_1)^{(\gamma/\delta)(\xi w+\beta)}$$

which is true if and only if either $\xi w + \beta = 0$ or $\gamma = 0$. Since $\xi w_i + \beta$ is a pairwise-independent function, we are guaranteed that $\xi w + \beta = 0$ happens with negligible probability. Thus, we conclude that $\mathcal{B}_2$ outputs 1 iff $\gamma = 0$, and this was a normal-type signature, and $\mathcal{B}_2$ outputs 0 iff $\gamma \neq 0$ and this was a simulation-type signature.

Proof of Lemma 6. Suppose that there exists an adversary $\mathcal{A}$ that outputs normal-type forgeries with non-negligible probability in **Game** q. Then we construct an adversary $\mathcal{B}_3$ for the CDH problem as follows.

$\mathcal{B}_3$ is given $X = G^x,\ Y = G^y$ and must compute G^{xy}. $\mathcal{B}_3$ will proceed as follows.

Message space setup and key generation: $\mathcal{B}_3$ will implicitly set $\alpha := xy$ and $a_2 := y$. It chooses b, a_1 at random from $\mathbb{Z}_p^*$. $\mathcal{B}_3$ needs to be able to compute $V_2^{a_2}$, so it chooses random $v_2 \in \mathbb{Z}_p^*$ and sets $V_2 := G^{v_2}$. It also wants to have the discrete logarithm of V_1, so it will choose random $v_1 \in \mathbb{Z}_p^*$ and set $V_1 := G^{v_1}$. $\mathcal{B}_3$ chooses $U, C, F \in \mathbb{G}$ and $H, V \in \mathbb{G}^*$ at random, sets $G^{a_2} := Y$, and computes $V V_2^{a_2} = V Y^{v_2}$. It chooses random $\rho' \in \mathbb{Z}_p^*$ and sets $X_1 := X^{\rho'}$ and $X_2 := Y^{a_1 b/\rho'}$. The rest of the parameters can be constructed honestly.

Signature queries: On a signature query, $\mathcal{B}_3$ chooses w at random, computes $(M_1, M_2, M_3) = (C^w, F^w, U^w)$, and generates a simulation-type signature as follows. It chooses random $r_1, r_2, z_1, z_2 \in \mathbb{Z}_p$, and random $s \in \mathbb{Z}_p$ and implicitly sets $\gamma := (x - s)$. $\mathcal{B}_3$ computes

$$\begin{aligned}
S_1 &:= Y^{sa_1} V^r = G^{ysa_1} V^r = G^{ysa_1 + xya_1 - xya_1} V^r = G^{xya_1} V^r G^{(s-x)ya_1} \\
&\quad = G^{\alpha a_1} V^r G^{-\gamma a_2 a_1}, \\
S_2 &:= Y^{-s} V_1^r G^{z_1} = G^{-ys} V_1^r G^{z_1} = G^{-ys+xy-xy} V_1^r G^{z_1} = G^{-xy} V_1^r G^{z_1} G^{(x-s)y} \\
&\quad = G^{-\alpha} V_1^r G^{z_1} G^{\gamma a_2}, \\
S_4 &:= V_2^r G^{z_2} X^{a_1} G^{-sa_1} = V_2^r G^{z_2} G^{xa_1} G^{-sa_1} = V_2^r G^{z_2} G^{(x-s)a_1} = V_2^r G^{z_2} G^{a_1 \gamma}.
\end{aligned}$$

The rest of the signature can be computed honestly.

Adversary's forgery: When the adversary outputs a normal-type forgery, there exists r_1, r_2, z_1 such that $S_2 = G^{-\alpha} V_1^{r_1+r_2} G^{z_1}$, $S_3 = (G^b)^{-z_1}$, $S_6 = G^{r_2 b}$, and $S_7 = G^{r_1}$. Thus, $\mathcal{B}_3$ can compute

$$\begin{aligned}
S_2^{-1} \cdot S_7^{v_1} S_6^{v_1/b} S_3^{-1/b} &= G^{\alpha} V_1^{-(r_1+r_2)} G^{-z_1} \cdot (G^{r_1})^{v_1} (G^{r_2 b})^{v_1/b} ((G^b)^{-z_1})^{-1/b} \\
&= G^{\alpha} V_1^{-r_1-r_2} G^{-z_1} \cdot (G^{v_1})^{r_1} (G^{v_1})^{r_2} G^{z_1} \\
&= G^{\alpha} V_1^{-r_1-r_2} G^{-z_1} \cdot V_1^{r_1} V_1^{r_2} G^{z_1} \\
&= G^{\alpha}.
\end{aligned}$$

$\mathcal{B}_3$ will output this value. By our choice of parameters, recall that $\alpha = xy$, so it holds that $G^{\alpha} = G^{xy}$ as desired.

That is, $\mathcal{B}_3$ can solve the CDH problem.

Let MSGGen be an extended random message generator that first chooses $\omega = m$ randomly from $\mathbb{Z}_p$ and then computes $msg = (C^m, F^m, U^m)$. Note that this is what the reduction algorithm does in the proof of Theorem 7. Therefore, the same reduction algorithm works for the case of extended random message attacks with respect to message generator MSGGen. We thus have the following.

Table 1. Efficiency comparison of constant-size SPS over symmetric bilinear groups.

Scheme	$\lvert msg\rvert$	$\lvert gk\rvert + \lvert vk\rvert$	$\lvert\sigma\rvert$	#(PPE)	Assumption
[4]	k	$2k + 13$	7	2	q-SFP
SIG1	k	$2k + 21$	14	7	DLIN

Corollary 1. *Under the DLIN assumption,* rSIG *scheme is UF-XRMA w.r.t. the message generator that provides $\omega = m$ for every message $msg = (C^m, F^m, U^m)$. In particular, for any p.p.t. algorithm $\mathcal{A}$ against* rSIG *that is given at most $q_s(\lambda)$ signatures, there exists p.p.t. algorithm $\mathcal{B}$ such that* $\mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{rSIG},\mathcal{A}}(\lambda) \le (q_s(\lambda) + 2) \cdot \mathrm{Adv}^{\mathsf{dlin}}_{\mathcal{G},\mathcal{B}}(\lambda)$.

5.4. *Security and Efficiency of Resulting* SIG1

Let SIG1 be the signature scheme obtained from TOS and rSIG by following the first generic construction in Sect. 4. From Theorems 1, 2, 6 and 7, the following is immediate.

Theorem 8. SIG1 *is a structure-preserving signature scheme that yields constant-size signatures, and is UF-CMA under the DLIN assumption. In particular, for any p.p.t. algorithm $\mathcal{A}$ for* SIG1 *making at most $q_s(\lambda)$ signing queries, there exists p.p.t. algorithm $\mathcal{B}$ such that* $\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG1},\mathcal{A}}(\lambda) \le (q_s(\lambda) + 3) \cdot \mathrm{Adv}^{\mathsf{dlin}}_{\mathcal{G},\mathcal{B}}(\lambda) + 1/p(\lambda)$*, where $p(\lambda)$ is the size of the groups produced by $\mathcal{G}$.*

The efficiency is summarized in Table 1. It is compared to an existing efficient structure-preserving scheme in [4, Section 5.2] (The original scheme is presented over asymmetric bilinear groups. It is translated to the symmetric setting for our purpose). We measure the efficiency by counting the number of group elements and the number of pairing product equations for verifying a signature.

In Table 2, we also assess the cost of proving possession of valid signatures and messages by using Groth–Sahai NIWI and NIZK proof system. Columns "σ" indicate the case where a witness is a valid signature (regarding the signature scheme from [4], we optimize by putting randomizable parts of a signature in the clear). The message is put in the clear. Similarly, columns "(σ, msg)" show the case where a witness consists of a valid signature and a message. Details of each assessment are as follows.

For NIWI, the cost of proving valid σ is counted by

$$|\mathrm{NIWI}(\sigma)| = |com| \times |\sigma_{\mathrm{wit}}| + |\sigma_{\mathrm{rnd}}| + |\pi_{NL}| \times \#(\mathrm{NLPPE}) + |\pi_L| \times \#(\mathrm{LPPE}) \quad (14)$$

and the cost of proving valid (σ, msg) is counted by

$$\begin{aligned} |\mathrm{NIWI}(\sigma, msg)| = {} & |com| \times (|\sigma_{\mathrm{wit}}| + |msg|) + |\sigma_{\mathrm{rnd}}| \\ & + |\pi_{NL}| \times \#(\mathrm{NLPPE}) + |\pi_L| \times \#(\mathrm{LPPE}) \end{aligned} \quad (15)$$

where $|\pi_{L/NL}|$, $|\sigma_{\mathrm{rnd}}|$, $|\sigma_{\mathrm{wit}}|$, $|com|$ are the size of a proof for a linear/nonlinear relation, randomizable parts of a signature, rest of the parts in the signature, and commitment per witness, respectively. Also, LPPE and NLPPE denotes the linear and nonlinear PPEs in the verification predicate of the signature scheme.

To achieve zero-knowledge, extra procedure is needed. For every pairing of constants in the target PPEs, either of the input constants is turned into a witness, and correctness of its commitment is proven using a multiscalar multiplication equation. Let #(CONST) denote the minimum number of constants that covers all constant pairings. For instance, if there are three pairings $e(A, B)$, $e(A, C)$, and $e(A, D)$ with constants A, B, C, D, only A need to be turned into a witness and hence #(CONST) $= 1$ in this example. Let $|\pi_{MS}|$ denote the size of the proof for correct commitment to A. Let σ_{var} denote elements in σ_{rnd} that are included in CONST. Using these parameters, the cost for proving ones possession of a correct signature in zero-knowledge is estimated as

$$\begin{aligned}|\text{NIZK}(\sigma)| =& |com| \times (|\sigma_{\text{wit}}| + \#(\text{CONST})) + (|\sigma_{\text{rnd}}| - |\sigma_{\text{var}}|) + |\pi_{NL}| \times \#(\text{NLPPE}) \\ &+ |\pi_L| \times \#(\text{LPPE}) + |\pi_{MS}| \times \#(\text{CONST})\end{aligned} \tag{16}$$

and the cost of proving valid (σ, msg) is counted by

$$\begin{aligned}|\text{NIZK}(\sigma, msg)| =& |com| \times (|\sigma_{\text{wit}}| + |msg| + \#(\text{CONST})) + (|\sigma_{\text{rnd}}| - |\sigma_{\text{var}}|) \\ &+ |\pi_{NL}| \times \#(\text{NLPPE}) + |\pi_L| \times \#(\text{LPPE}) + |\pi_{MS}| \times \#(\text{CONST}).\end{aligned} \tag{17}$$

According to [33], we have $(|com|, |\pi_L|, |\pi_{NL}|) = (3, 3, 9)$ in $\mathbb{G}$, and $|\pi_{MS}| = 3$ in $\mathbb{Z}_p$. Proof π_{MS} can consist of elements in $\mathbb{G}$ by describing the relation of correct commitment of public value with a pairing product equation. It turns entire proof to be structure-preserving with increased proof size.

For [4], we have $|\sigma_{\text{wit}}| = 3$, $|\sigma_{\text{rnd}}| = 4$. Since the verification consists of two nonlinear equations, we have #(NLPPE) $= 0$ and #(LPPE) $= 2$. This results in $|\text{NIWI}(\sigma)| = 3{\cdot}3{+}4{+}9{\cdot}0{+}3{\cdot}2 = 19$ and $|\text{NIWI}(\sigma, msg)| = 3{\cdot}(3{+}k){+}4{+}9{\cdot}0{+}3{\cdot}2 = 3k{+}19$. For NIZK$(\sigma)$, turning #(CONST) $= k + 4 + 2$ constants into witnesses eliminates constant pairings in the signature verification. In detail, k comes from the pairings that involve the message, 4 is from the parings that only involves the public-key, and $2 = |\sigma_{\text{var}}|$ is from the pairings that involves the randomizable part of the signature. Thus $3 \cdot (6{+}k) {-} 2$ group elements and $3 \cdot (6 + k)$ Zp elements are needed on top of NIWI(σ). For NIZK(σ, msg), on the other hand, the message is a part of the witness. Thus we can set #(CONST) $= 6$ and the additional cost on NIWI(σ, msg) is $3{\cdot}6{-}2$ group elements and $3{\cdot}6$ $\mathbb{Z}_p$ elements. This results in $(3k + 35, 3k + 18)$ and $(3k + 35, 18)$ elements as shown in Table 2.

Regarding to SIG1, whole signature is considered as a witness. Thus we have $|\sigma_{\text{wit}}| = 14$ and $|\sigma_{\text{rnd}}| = 0$. And the verification consists of 6 linear equations and 1 nonlinear equation; #(NLPPE) $= 1$ and #(LPPE) $= 6$. We thus have $|\text{NIWI}(\sigma)| = 3 \cdot 14 + 0 + 9 \cdot 1 + 3 \cdot 6 = 69$ and $|\text{NIWI}(\sigma, msg)| = 3 \cdot (14 + k) + 0 + 9 \cdot 1 + 3 \cdot 6 = 3k + 69$. For NIZK$(\sigma)$, we have #(CONST) $= 1 + k$ constant pairings in the signature verification, which results in adding $3 + 3k$ elements in both $\mathbb{G}$ and $\mathbb{Z}_p$ to NIWI(σ). Finally, for NIZK(σ, msg), we have #(CONST) $= 1$, which adds three elements in $\mathbb{G}$ and $\mathbb{Z}_p$ to NIWI(σ, msg).

Table 2. Size of GS proofs and commitments for proving possession of a valid signature and message in WI or ZK.

Scheme	NIWI(σ)	NIWI(σ, msg)	NIZK(σ)	NIZK(σ, msg)
[4]	19	$3k + 19$	$(3k + 35, 3k + 18)$	$(3k + 35, 18)$
SIG1	69	$3k + 69$	$(3k + 72, 3k + 3)$	$(3k + 72, 3)$

Numbers count elements in $\mathbb{G}$. For ZK, (x, y) denotes x elements in $\mathbb{G}$ and y elements in $\mathbb{Z}_p$

6. Instantiating SIG2

We instantiate the POS and xSIG building blocks of our second generic construction to obtain our second SPS scheme. Here we choose the Type-III bilinear group setting. The resulting SIG2 scheme is an efficient structure-preserving signature scheme based on SXDH and XDLIN.

6.1. *Setup for Type-III Groups*

The following setup procedure is common for all building blocks in this section. The global parameter gk is given to all functions implicitly.

- Setup(1^λ): Run $\Lambda = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda)$ and choose generators $G \in \mathbb{G}_1^*$ and $\hat{G} \in \mathbb{G}_2^*$. Also choose u, f_1, f_2 randomly from $\mathbb{Z}_p^*$, compute $F_1 := G^{f_1}$, $\hat{F}_1 := \hat{G}^{f_1}$, $F_2 := G^{f_2}$, $\hat{F}_2 := \hat{G}^{f_2}$, $U := G^u$, $\hat{U} := \hat{G}^u$, and output $gk := (\Lambda, G, \hat{G}, F_1, \hat{F}_1, F_2, \hat{F}_2, U, \hat{U})$.

A gk defines a message space $\mathcal{M}_{\mathsf{x}} = \{(\hat{F}_1^m, \hat{F}_2^m, \hat{U}^m) \in (\mathbb{G}_2^*)^3 \mid m \in \mathbb{Z}_p\}$ for the XRMA-secure signature scheme in this section. For our generic construction to work, the partial one-time signature scheme must have the same key space.

6.2. *Partial One-Time Signatures for Unilateral Messages*

We first construct a partial one-time signature scheme, POSu2, for messages in $\mathbb{G}_2^k$ for $k > 0$. The suffix "u2" indicates that the scheme is unilateral and messages are taken from $\mathbb{G}_2$. Correspondingly, POSu1 refers to the scheme whose messages belong to $\mathbb{G}_1$, which is obtained by swapping $\mathbb{G}_2$ and $\mathbb{G}_1$ in the following description. In the following section we will show how to combine POSu2 and POSu1 to obtain signatures on bilateral messages consisting of elements from both $\mathbb{G}_1$ and $\mathbb{G}_2$.

Our POSu2 scheme is a minor refinement of the one-time signature scheme introduced in [7]. It comes, however, with a security proof for the new security model. Basically, a one-time public-key in our scheme consists of one element in the source group $\mathbb{G}_1$, the opposite group from the one to which the messages belong. This property is very useful when we move on to construct a POS scheme for signing bilateral messages.

Like the tags in the TOS of Sect. 5.2, the one-time public-keys of POSu2 will have to be in an extended form, (F_1^a, F_2^a, U^a), to meet the constraint from xSIG presented in the sequel. The extended part (F_1^a, F_2^a) can be dropped if unnecessary.

[Scheme $\mathsf{POSu2}$**]**

$\mathsf{POSu2.Key}(gk)$: Take generators U and $\hat{U}$ from gk. Choose w_r uniformly from $\mathbb{Z}_p^*$ and compute $G_r := U^{w_r}$. For $i = 1, \ldots, k$, uniformly choose χ_i and γ_i from $\mathbb{Z}_p$ and compute $G_i := U^{\chi_i} G_r^{\gamma_i}$. Output $pk := (G_r, G_1, \ldots, G_k) \in \mathbb{G}_1^{k+1}$ and $sk := (\chi_1, \gamma_1, \ldots, \chi_k, \gamma_k, w_r)$.

$\mathsf{POSu2.Update}(gk)$: Take F_1, F_2, U from gk. Choose $a \leftarrow \mathbb{Z}_p$ and output $opk := (F_1^a, F_2^a, U^a) \in \mathbb{G}_1^3$ and $osk := a$.

$\mathsf{POSu2.Sign}(sk, msg, osk)$: Parse msg into $(\tilde{M}_1, \ldots, \tilde{M}_k) \in \mathbb{G}_2^k$. Take a and w_r from osk and sk, respectively. Choose ρ randomly from $\mathbb{Z}_p$ and compute $\zeta := a - \rho\, w_r \bmod p$. Then compute and output $\sigma := (\tilde{Z}, \tilde{R}) \in \mathbb{G}_2^2$ as the signature, where

$$\tilde{Z} := \hat{U}^{\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i} \quad \text{and} \quad \tilde{R} := \hat{U}^{\rho} \prod_{i=1}^{k} \tilde{M}_i^{-\gamma_i}. \tag{18}$$

$\mathsf{POSu2.Vrf}(pk, opk, msg, \sigma)$: Parse σ as $(\tilde{Z}, \tilde{R}) \in \mathbb{G}_2^2$, msg as $(\tilde{M}_1, \ldots, \tilde{M}_k) \in \mathbb{G}_2^k$, and opk as (A_1, A_2, A). Return 1, if

$$e(A, \hat{U}) = e(U, \tilde{Z})\, e(G_r, \tilde{R}) \prod_{i=1}^{k} e(G_i, \tilde{M}_i) \tag{19}$$

holds. Return 0, otherwise.

Scheme $\mathsf{POSu2}$ is structure-preserving and has uniform one-time public-keys by construction. It is correct as the following relation holds for the verification equation and the computed signatures:

$$\begin{aligned} & e(U, \tilde{Z})\, e(G_r, \tilde{R}) \prod_{i=1}^{k} e(G_i, \tilde{M}_i) \\ &= e\left(U, \hat{U}^{\zeta} \prod_{i=1}^{k} \tilde{M}_i^{-\chi_i}\right) e\left(G_r, \hat{U}^{\rho} \prod_{i=1}^{k} \tilde{M}_i^{-\gamma_i}\right) \prod_{i=1}^{k} e(U^{\chi_i} G_r^{\gamma_i}, \tilde{M}_i) \\ &= e\left(U, \hat{U}^{\zeta}\right)\, e\left(U^{w_r}, \hat{U}^{\rho}\right) = e\left(U^{\zeta + w_r \rho}, \hat{U}\right) = e(A, \hat{U}). \end{aligned}$$

Theorem 9. $\mathsf{POSu2}$ *is strongly unforgeable against OT-CMA if DBP*$_1$ *holds. In particular, for all p.p.t. algorithms* $\mathcal{A}$ *there exists a p.p.t. algorithm* $\mathcal{B}$ *such that* $\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{POSu2},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{dbp1}}_{\mathcal{G},\mathcal{B}}(\lambda) + 1/p(\lambda)$, *where* $p(\lambda)$ *is the size of the groups produced by* $\mathcal{G}$. *Moreover, the run-time overhead of the reduction* $\mathcal{B}$ *is a small number of multi-exponentiations per signing or key query.*

Proof. Using a successful forger $\mathcal{A}$ against $\mathsf{POSu2}$ as a black-box, we construct $\mathcal{B}$ that is successful in breaking DBP$_1$. Given instance $I_{\mathsf{dbp1}} = (\Lambda, G_z, G_r)$ of DBP$_1$, algorithm $\mathcal{B}$ simulates the attack game against $\mathsf{POSu2}$ as follows.

Key Generation: Set $U := G_z$, $\hat{U} \leftarrow \mathbb{G}_2^*$, and $gk := (\Lambda, U^g, \hat{U}^g, U^{f_1'}, \hat{U}^{f_1'}, U^{f_2'}, \hat{U}^{f_2'}, U, \hat{U})$ for $g, f_1', f_2' \leftarrow \mathbb{Z}_p^*$. Then generate pk by following $\mathsf{POSu2.Key}(gk)$ except that G_r is taken from I_{dbp1}.

One-time key query to $\mathcal{O}_t$: On receiving a one-time key query, generate $\zeta, \rho \leftarrow \mathbb{Z}_p$, compute $A := U^\zeta G_r^\rho$, $A_1 := A^{f_1'}$, $A_2 := A^{f_2'}$ with f_1' and f_2' generated in Setup, and return $opk := (A_1, A_2, A)$.

Signature query to $\mathcal{O}_s$: On receiving a signing query, $msg^{(j)}$, compute $\tilde{Z}$ and $\tilde{R}$ as described in (18) taking χ_i and γ_i from those used in key generation and ζ and ρ from those used in simulating $\mathcal{O}_t$. Then output $\sigma := (\tilde{Z}, \tilde{R})$. For each signing, transcript (opk, σ, msg) is recorded.

When $\mathcal{A}$ outputs a forgery $(opk^\dagger, \sigma^\dagger, msg^\dagger)$, algorithm $\mathcal{B}$ searches the records for (opk, σ, msg) such that $opk^\dagger = opk$ and $(msg^\dagger, \sigma^\dagger) \neq (msg, \sigma)$. If no such entry exists, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ computes

$$\tilde{Z}^\star := \frac{\tilde{Z}^\dagger}{\tilde{Z}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^\dagger}{\tilde{M}_i} \right)^{\chi_i}, \quad \text{and} \quad \tilde{R}^\star := \frac{\tilde{R}^\dagger}{\tilde{R}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^\dagger}{\tilde{M}_i} \right)^{\gamma_i}, \tag{20}$$

where $(\tilde{Z}, \tilde{R}, \tilde{M}_1, \ldots, \tilde{M}_k)$ and its dagger counterpart are taken from (σ, msg) and $(\sigma^\dagger, msg^\dagger)$, respectively. $\mathcal{B}$ finally outputs $(\tilde{Z}^\star, \tilde{R}^\star)$. This completes the description of $\mathcal{B}$.

We first claim that the simulation by $\mathcal{B}$ is perfect; keys distribute uniformly due to the randomness of G_z and G_r in the given instance, and signatures are computed following the legitimate procedure. It is noted that $f_1'g$ and $f_2'g$ corresponds to f_1 and f_2 in the real execution. Accordingly, $\mathcal{A}$ outputs a successful forgery with noticeable probability and $\mathcal{B}$ finds a corresponding record (opk, σ, msg).

We next claim that each χ_i is independent of the view of $\mathcal{A}$. Concretely, we show that, if coins $\chi_1, \ldots, \chi_k$ are distributed uniformly over $(\mathbb{Z}_p)^k$, other coins $\gamma_1, \ldots, \gamma_k, \zeta^{(1)}, \rho^{(1)}, \ldots, \zeta^{(q_s)}, \rho^{(q_s)}$ are distributed uniformly and $\mathcal{A}$'s view is consistent. Observe that the view of $\mathcal{A}$ making q signing queries consists of independent group elements $(U, \hat{U})$, (G, F_1, F_2), $(G_r, G_1, \ldots, G_k)$ and $(A^{(j)}, \tilde{Z}^{(j)}, \tilde{M}_1^{(j)}, \ldots, \tilde{M}_k^{(j)})$ for $j = 1, \ldots, q_s$ (Note that $\hat{G}$, $\hat{F}_1$, $\hat{F}_2$, and $A_1^{(j)}$, $A_2^{(j)}$, and $\tilde{R}^{(j)}$ for all j are uniquely determined by the other group elements). We represent the view by the discrete logarithms of these group elements with respect to bases U and $\hat{U}$ in each group. Namely, the view is represented by $(g, f_1', f_2', w_r, w_1, \ldots, w_k)$ and $(a^{(j)}, z^{(j)}, m_1^{(j)}, \ldots, m_k^{(j)})$ for $j = 1, \ldots, q_s$. To be consistent, the view and the coins must satisfy the following relations:

$$w_i = \chi_i + w_r \gamma_i \quad \text{for } i = 1, \ldots, k, \text{ and} \tag{21}$$

$$a^{(j)} = \zeta^{(j)} + w_r \rho^{(j)}, \quad \text{and} \quad z^{(j)} = \zeta^{(j)} - \sum_{i=1}^{k} m_i^{(j)} \chi_i \quad \text{for } j = 1, \ldots, q_s. \tag{22}$$

From relation (21), $(\gamma_1, \ldots, \gamma_k)$ is distributed uniformly according to the uniform choice of $(\chi_1, \ldots, \chi_k)$. From the second relation in (22) for every j, if $(m_1, \ldots, m_k) \neq (0, \ldots, 0)$ then $\zeta^{(j)}$ is distributed uniformly according to the uniform distribution of

$(\chi_1, \ldots, \chi_k)$. Then, from the first relation of (22), $\rho^{(j)}$ is distributed uniformly, too. If $(m_1, \ldots, m_k) = (0, \ldots, 0)$, then $\zeta^{(j)}$ and $\rho^{(j)}$ are independent of $(\chi_1, \ldots, \chi_k)$ and can be uniformly assigned by following the first relation in (22).

Finally, we claim that $(\tilde{Z}^\star, \tilde{R}^\star)$ is a valid solution to the given instance of DBP$_1$. Since both forged and recorded signatures fulfill the verification equation, dividing the equations results in

$$\begin{aligned} 1 &= e\left(U, \frac{\tilde{Z}^\dagger}{\tilde{Z}}\right) e\left(G_r, \frac{\tilde{R}^\dagger}{\tilde{R}}\right) \prod_{i=1}^{k} e\left(U^{\chi_i} G_r^{\gamma_i}, \frac{\tilde{M}_i^\dagger}{\tilde{M}_i}\right) \\ &= e\left(U, \frac{\tilde{Z}^\dagger}{\tilde{Z}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^\dagger}{M_i}\right)^{\chi_i}\right) e\left(G_r, \frac{\tilde{R}^\dagger}{\tilde{R}} \prod_{i=1}^{k} \left(\frac{\tilde{M}_i^\dagger}{M_i}\right)^{\gamma_i}\right) \\ &= e\left(U, \tilde{Z}^\star\right) e\left(G_r, \tilde{R}^\star\right). \end{aligned}$$

What remains is to prove that $\tilde{Z}^\star \neq 1$. If $msg^\dagger \neq msg^{(j)}$, there exists $\ell \in \{1, \ldots, k\}$ such that $\frac{\tilde{M}_\ell^\dagger}{M_\ell} \neq 1$. As already proven, χ_ℓ is independent of the view of $\mathcal{A}$ and of the other χ_i values. Thus $\left(\frac{M_\ell^\dagger}{M_\ell}\right)^{\chi_\ell}$ is distributed uniformly over $\mathbb{G}_2$ and so is $\tilde{Z}^\star$. Accordingly, $Z^\star = 1$ holds only if $Z^\dagger = \tilde{Z} \prod (M_i^\dagger / M_i)^{-\chi_i}$, which happens only with probability $1/p$ over the choice of χ_ℓ. Otherwise, if $msg^\dagger = msg^{(j)}$ and $(Z^\dagger, R^\dagger) \neq (Z, R)$, then, we have $Z^\dagger = Z$ to fulfil $Z^\star = 1$. However, if $Z^\dagger = Z$, then $R^\dagger = R$ holds since the verification equation uniquely determines such $R^\dagger$ and R. Thus $msg^\dagger = msg^{(j)}$ and $(Z^\dagger, R^\dagger) \neq (Z, R)$ can never happen. We thus have $\mathrm{Adv}^{\textsf{sot-cma}}_{\textsf{POSu2},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\textsf{dbp1}}_{\mathcal{G},\mathcal{B}}(\lambda) + 1/p$ as stated.

6.3. *Partial One-Time Signatures for Bilateral Messages*

Using POSu1 for $msg \in \mathbb{G}_1^{k_1+1}$ and POSu2 for $msg \in \mathbb{G}_2^{k_2}$, we construct a POSb scheme for signing bilateral messages $(msg_1, msg_2) \in \mathbb{G}_1^{k_1} \times \mathbb{G}_2^{k_2}$. The scheme is a simple two-story construction where msg_2 is signed by POSu2 with one-time secret-key $osk_2 \in \mathbb{G}_1$ and then the one-time public-key opk_2 is attached to msg_1 and signed by POSu1. Public-key opk_2 is included in the signature, and opk_1 is output as a one-time public-key for POSb.

[Scheme POSb**]**

POSb.Key(gk): Run $(pk_1, sk_1) \leftarrow$ POSu1.Key(gk) for message size $k_1 + 1$ and $(pk_2, sk_2) \leftarrow$ POSu2.Key(gk) for message size k_2. Set $pk := (pk_1, pk_2)$ and $sk := (sk_1, sk_2)$, and output (pk, sk).

POSb.Update(gk): Run $(opk, osk) \leftarrow$ POSu1.Update(gk) and output (opk, osk).

POSb.Sign(sk, msg, osk): Parse msg into $(msg_1, msg_2) \in \mathbb{G}_1^{k_1} \times \mathbb{G}_2^{k_2}$, and sk into (sk_1, sk_2). Run $(opk_2, osk_2) \leftarrow$ POSu2.Update(gk), and compute $\sigma_2 \leftarrow$ POSu2.Sign(sk_2, msg_2, osk_2) and $\sigma_1 \leftarrow$ POSu1.Sign$(sk_1, (msg_1, opk_2), osk)$. Output $\sigma := (\sigma_1, \sigma_2, opk_2)$.

$\mathsf{POSb.Vrf}(pk, opk, msg, \sigma)$: Parse msg into $(msg_1, msg_2) \in \mathbb{G}_1^{k_1} \times \mathbb{G}_2^{k_2}$, and σ into $(\sigma_1, \sigma_2, opk_2)$. If $1 = \mathsf{POSu1.Vrf}(pk_1, opk, (msg_1, opk_2), \sigma_1) = \mathsf{POSu2.Vrf}(pk_2, opk_2, msg_2, \sigma_2)$, output 1. Otherwise, output 0.

We consider dropping unnecessary extended part from opk_2 so that it consists of only one group element. Then, for a message in $\mathbb{G}_1^{k_1} \times \mathbb{G}_2^{k_2}$, the above POSb uses a public-key of size (k_2+1, k_1+2), yields a one-time public-key of size $(0, 3)$, and a signature of size $(3, 2)$. Verification requires 2 pairing product equations. A one-time public-key, which is treated as a message to xSIG in this section, is of the form $opk = (\hat{F}_1^a, \hat{F}_2^a, \hat{U}^a) \in \mathbb{G}_2^3$. The structure preservation and uniform public-key properties carry over from the underlying $\mathsf{POSu1}$ and $\mathsf{POSu2}$.

Theorem 10. *Scheme* POSb *is strongly unforgeable against OT-CMA if SXDH holds. In particular, for all p.p.t. algorithms* $\mathcal{A}$ *there exists a p.p.t. algorithm* $\mathcal{B}$ *such that* $\mathrm{Adv}^{\mathsf{sot\text{-}cma}}_{\mathsf{POSb},\mathcal{A}}(\lambda) \le \mathrm{Adv}^{\mathsf{sxdh}}_{\mathcal{G},\mathcal{B}}(\lambda) + 2/p(\lambda)$, *where* $p(\lambda)$ *is the size of the groups produced by* $\mathcal{G}$. *Moreover, the run-time overhead of the reduction* $\mathcal{B}$ *is a small number of multi-exponentiations per signing or key query.*

Proof. Suppose an adversary $\mathcal{A}$ outputs a forgery $(opk^\dagger, \sigma^\dagger, msg^\dagger)$. Then there exists a triple (σ, opk, msg) observed by the signing oracle such that $opk^\dagger = opk$ and $(msg^\dagger, \sigma^\dagger) \neq (msg, \sigma)$. Let $msg^\dagger = (msg_1^\dagger, msg_2^\dagger)$ and $\sigma^\dagger = (\sigma_1^\dagger, \sigma_2^\dagger, opk_2^\dagger)$. Similarly, let $msg = (msg_1, msg_2)$ and $\sigma = (\sigma_1, \sigma_2, opk_2)$. Then there are two cases; either $((msg_1, opk_2), \sigma_1) \neq ((msg_1^\dagger, opk_2^\dagger), \sigma_1^\dagger)$, or $opk_2 = opk_2^\dagger$ and $(msg_2, \sigma_2) \neq (msg_2^\dagger, \sigma_2^\dagger)$. In the first case we break the strong unforgeability of $\mathsf{POSu1}$ and contradict the DBP_2 assumption; in the second case we break the strong unforgeability of $\mathsf{POSu2}$ and contradict the DBP_1 assumption.

Accordingly, we have $\mathrm{Adv}^{\mathsf{ot\text{-}cma}}_{\mathsf{POSb},\mathcal{A}}(\lambda) \le \mathrm{Adv}^{\mathsf{dbp1}}_{\mathcal{G},\mathcal{A}}(\lambda) + 1/p + \mathrm{Adv}^{\mathsf{dbp2}}_{\mathcal{G},\mathcal{B}}(\lambda) + 1/p \le \mathrm{Adv}^{\mathsf{sxdh}}_{\mathcal{G},\mathcal{B}}(\lambda) + 2/p$.

6.4. *XRMA-Secure Signature Scheme*

An intuition behind our XRMA-secure scheme is the same as that of RMA-secure scheme in the previous section. Recall that $gk = (\Lambda, G, \hat{G}, F_1, \hat{F}_1, F_2, \hat{F}_2, U, \hat{U})$ with $\Lambda = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ is generated by $\mathsf{Setup}(1^\lambda)$ in advance (see Sect. 6.1).

[Scheme xSIG**]**

$\mathsf{xSIG.Gen}(gk)$: Given gk as input, uniformly select generators $V, V' \leftarrow \mathbb{G}_1^*$, $\hat{V}, \hat{V}' \in \mathbb{G}_2^*$ such that $V \sim \hat{V}$, $V' \sim \hat{V}'$, $\tilde{H} \leftarrow \mathbb{G}_2^*$, and exponents $a, b, \alpha, \rho \leftarrow \mathbb{Z}_p^*$. Then compute and output $vk := (gk, \tilde{B}, \tilde{A}, \tilde{B}_a, \tilde{R}, \tilde{W}, \tilde{H}, X_1, \tilde{X}_2)$ and $sk := (vk, K_1, K_2, V, V')$ where

$$\begin{array}{lllll} \tilde{B} := \hat{G}^b, & \tilde{A} := \hat{G}^a, & \tilde{B}_a := \hat{G}^{ba}, & \tilde{R} := \hat{V}(\hat{V}')^a, & \tilde{W} := \tilde{R}^b \\ X_1 := G^\rho, & \tilde{X}_2 := \hat{G}^{\alpha \cdot b/\rho}, & K_1 := G^\alpha, & K_2 := G^b. & \end{array}$$

$\mathsf{xSIG.Sign}(sk, msg)$: Parse msg into $(\tilde{M}_1, \tilde{M}_2, \tilde{M}_3) = (\hat{F}_1^m, \hat{F}_2^m, \hat{U}^m) \in \mathbb{G}_2^3$ ($m \in \mathbb{Z}_p$). Pick random $r_1, r_2, z \leftarrow \mathbb{Z}_p$. Let $r := r_1 + r_2$. Compute and output signature $\sigma := (\tilde{S}_0, S_1, \ldots, S_5)$ where

$$\tilde{S}_0 := (\tilde{M}_3\tilde{H})^{r_1}, \quad S_1 := K_1 V^r, \quad S_2 := (V')^r G^{-z}, \quad S_3 := K_2^z, \quad S_4 := K_2^{r_2}, \quad S_5 := G^{r_1}.$$

$\mathsf{xSIG.Vrfy}(vk, msg, \sigma)$: Parse msg into $(\tilde{M}_1, \tilde{M}_2, \tilde{M}_3)$ and σ into $(\tilde{S}_0, S_1, \ldots, S_5)$. Also parse vk accordingly. Verify the following pairing product equations:

$$e(S_1, \tilde{B})e(S_2, \tilde{B}_a)e(S_3, \tilde{A}) = e(S_4, \tilde{R})e(S_5, \tilde{W})e(X_1, \tilde{X}_2), \tag{23}$$
$$e(S_5, \tilde{M}_3\tilde{H}) = e(G, \tilde{S}_0), \tag{24}$$
$$e(F_1, \tilde{M}_3) = e(U, \tilde{M}_1), \tag{25}$$
$$e(F_2, \tilde{M}_3) = e(U, \tilde{M}_2). \tag{26}$$

The scheme is structure-preserving by construction. We can easily verify the correctness as follows.

$$\begin{aligned}
\text{(Left-hand of (23))} &= e\left(G^\alpha V^r, \hat{G}^b\right) e\left(\left(V'\right)^r G^{-z}, \hat{G}^{ba}\right) e\left(G^{bz}, \hat{G}^a\right) \\
&= e\left(G, \hat{G}\right)^{\alpha b} e\left(V, \hat{G}\right)^{br} e\left(V', \hat{G}\right)^{abr} \\
&= e\left(G, \hat{V}\right)^{b(r_1+r_2)} e\left(G, \hat{V}'\right)^{ab(r_1+r_2)} e\left(G, \hat{G}\right)^{\alpha b} \\
&= e\left(G^{br_2}, \hat{V}\left(\hat{V}'\right)^a\right) e\left(G^{r_1}, \hat{V}^b\left(\hat{V}'\right)^{ba}\right) e\left(G, \hat{G}\right)^{\alpha b} \\
&= \text{(Right-hand of (23))}
\end{aligned}$$

Equation (23) holds since $r = r_1 + r_2$, $V \sim \hat{V}$, and $V' \sim \hat{V}'$. The followings also hold.

$$\begin{aligned}
&\text{(Left-hand of (24))} = e(G^{r_1}, \hat{U}^m\tilde{H}) = e(G, \hat{U}^m\tilde{H})^{r_1} = e(G, (\hat{U}^m\tilde{H})^{r_1}) \\
&\quad = \text{(Right-hand of (24))}, \\
&\text{(Left-hand of (25))} = e(F_1, \hat{U}^m) = e(F_1, \hat{U})^m = e(U, \hat{F}_1^m) = \text{(Right-hand of (25))}, \\
&\text{(Left-hand of (26))} = e(F_2, \hat{U}^m) = e(F_2, \hat{U})^m = e(U, \hat{F}_2^m) = \text{(Right-hand of (26))}.
\end{aligned}$$

Theorem 11. *The above* xSIG *scheme is UF-XRMA with respect to the message generator that returns $\omega = m$ for every random message $msg = (\hat{F}_1^m, \hat{F}_2^m, \hat{U}^m)$ under the DDH$_2$ and XDLIN$_1$ assumptions. In particular for any p.p.t. algorithm $\mathcal{A}$ for* xSIG *making at most $q(\lambda)$ signing queries, there exist p.p.t. algorithms $\mathcal{B}_1, \mathcal{B}$ such that* $\mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{xSIG},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{ddh2}}_{\mathcal{G},\mathcal{B}_1}(\lambda) + (q(\lambda)+1)\mathrm{Adv}^{\mathsf{xdlin1}}_{\mathcal{G},\mathcal{B}}(\lambda)$.

Proof. In this scheme, simulation-type signatures are of the form $\sigma = (\tilde{S}_0, S_1' = S_1 \cdot G^{-a\gamma}, S_2' = S_2 \cdot G^\gamma, S_3, S_4, S_5)$ for $\gamma \in \mathbb{Z}_p$. The outline of the proof follows that of

Water's dual signature scheme and is quite similar to the proof of Theorem 7. We start with the following lemma.

Lemma 7. *Any signature that is accepted by the verification algorithm must be either a normal-type signature or a simulation-type signature.*

Proof of Lemma 7. We ignore the last row of verification equations that establish that *msg* is well formed. A signature has three random exponents, r_1, r_2, z. A simulation-type signature has an additional exponent γ. We interpret S_5 as G^{r_1}, so the first verification equation implies that $\tilde{S_0} = (\hat{U}^m \tilde{H})^{r_1}$. For fixed $b \in \mathbb{Z}_p$ ($\hat{G}^b$ is included in vk), there exists $r_2, z \in \mathbb{Z}_p$ such that $S_3 = G^{bz}$, $S_4 = G^{br_2}$. If we fix $S_1 = G^{\alpha} V^r G^{-a\gamma}$, then a remaining unknown value is S_2. The verification equation is

$$e\left(S_1, \hat{G}^b\right) e\left(S_2, \hat{G}^{ba}\right) e\left(S_3, \hat{G}^a\right) = e\left(S_4, \tilde{R}\right) e\left(S_5, \tilde{R}^b\right) e\left(G, \hat{G}\right)^{\alpha b}$$

so we can fix $S_2 = (V')^r G^{-z} G^{\gamma}$.

Based on the notion of simulation-type signatures, we consider a sequence of games. Let p_i be the probability that the adversary succeeds in **Game i**, and $p_i^{\mathrm{norm}}(\lambda)$ and $p_i^{\mathrm{sim}}(\lambda)$ be the probability that he succeeds with a normal-type or simulation-type forgery, respectively. Then by Lemma 7, $p_i(\lambda) = p_i^{\mathrm{norm}}(\lambda) + p_i^{\mathrm{sim}}(\lambda)$ for all i.

Game 0: The actual Unforgeability under Extended Random Message Attacks game.

Lemma 8. *There exists an adversary $\mathcal{B}_1$ such that $p_0^{sim}(\lambda) \le \mathrm{Adv}_{\mathcal{G},\mathcal{B}_1}^{\mathsf{ddh2}}(\lambda)$.*

Game i: The real security game except that the first i signatures that are given by the oracle are simulation-type signatures.

Lemma 9. *There exists an adversary $\mathcal{B}_2$ such that $|p_{i-1}^{norm}(\lambda) - p_i^{norm}(\lambda)| \le \mathrm{Adv}_{\mathcal{G},\mathcal{B}_2}^{\mathsf{xdlin1}}(\lambda)$.*

Game q: All signatures given by the oracle are simulation-type signatures.

Lemma 10. *There exists an adversary $\mathcal{B}_3$ such that $p_q^{norm}(\lambda) \le \mathrm{Adv}_{\mathcal{G},\mathcal{B}_3}^{\mathsf{co\text{-}cdh}}(\lambda)$.*

We have shown that in **Game q**, $\mathcal{A}$ can output a normal-type forgery with at most negligible probability. Thus, by Lemma 9 we can conclude that the same is true in **Game 0**. Since we have already shown that in **Game 0** the adversary can output simulation-type forgeries only with negligible probability, and that any signature that is accepted by the verification algorithm is either normal type or simulation type, we conclude that the adversary can produce valid forgeries with only negligible probability

$$\begin{aligned}\mathrm{Adv}_{\mathsf{xSIG},\mathcal{A}}^{\mathsf{uf\text{-}xrma}}(\lambda) = \; & p_0(\lambda) = p_0^{\mathrm{sim}}(\lambda) + p_0^{\mathrm{norm}}(\lambda) \\ & \le p_0^{\mathrm{sim}}(\lambda) + \sum_{i=1}^{q} |p_{i-1}^{\mathrm{norm}}(\lambda) - p_i^{\mathrm{norm}}(\lambda)| + p_q^{\mathrm{norm}}(\lambda)\end{aligned}$$

$$\leq \mathrm{Adv}^{\mathsf{ddh2}}_{\mathcal{G},\mathcal{B}_1}(\lambda) + q\mathrm{Adv}^{\mathsf{xdlin1}}_{\mathcal{G},\mathcal{B}_2}(\lambda) + \mathrm{Adv}^{\mathsf{co\text{-}cdh}}_{\mathcal{G},\mathcal{B}_3}(\lambda)$$
$$\leq \mathrm{Adv}^{\mathsf{ddh2}}_{\mathcal{G},\mathcal{B}_1}(\lambda) + (q+1)\mathrm{Adv}^{\mathsf{xdlin1}}_{\mathcal{G},\mathcal{B}}(\lambda)$$

as stated. The last inequality holds since the CDH$_1$ assumption is implied by the XDLIN$_1$ assumption.

Proof of Lemma 8. We show that, if the adversary outputs a simulation-type forgery, then we can construct algorithm $\mathcal{B}_1$ that solves the DDH$_2$ problem. Algorithm $\mathcal{B}_1$ is given instance $(\Lambda, \hat{G}, \hat{G}^s, \hat{G}^a, \tilde{Z} \in \mathbb{G}_2)$ of DDH$_2$, and simulates the verification key and the signing oracle for the signature scheme ($\mathcal{B}_1$ does not have the values a, s).

$\mathcal{B}_1$ generates gk and vk as follows. It selects $G \leftarrow \mathbb{G}_1$, and exponents $u, f_1, f_2 \leftarrow \mathbb{Z}_p^*$, computes $F_1 := G^{f_1}$, $\hat{F}_1 := \hat{G}^{f_1}$, $F_2 := G^{f_2}$, $\hat{F}_2 := \hat{G}^{f_2}$, $U := G^u$, $\hat{U} := \hat{G}^u$, and sets them into gk. It also selects exponents $v, v' \leftarrow \mathbb{Z}_p^*$, computes $V := G^v$, $V' := G^{v'}$, $\hat{V} := \hat{G}^v$, $\hat{V}' := \hat{G}^{v'}$. Next, it selects exponents $b, \alpha, h, \rho \leftarrow \mathbb{Z}_p^*$, computes $\tilde{H} := \hat{G}^h$, and

$$\begin{aligned}
&\tilde{B} := \hat{G}^b, \quad \tilde{A} := \hat{G}^a, \quad \tilde{B}_a := (\hat{G}^a)^b, \quad \tilde{R} := \hat{V}(\hat{V}')^a = \hat{G}^v(\hat{G}^a)^v,\\
&\quad \tilde{W} := \tilde{R}^b = \hat{G}^{bv}(\hat{G}^a)^{bv}\\
&X_1 := G^\rho, \quad \tilde{X}_2 := \hat{G}^{\alpha b/\rho}, \quad K_1 := G^\alpha, \quad K_2 := G^b,
\end{aligned}$$

and sets them into vk and sk, accordingly.

$\mathcal{B}_1$ can generate normal-type signatures by using the (normal) signing algorithm since $\mathcal{B}_1$ has α, b and V, V'. For ith signature, $\mathcal{B}_1$ randomly selects $m_i \in \mathbb{Z}_p$, generates normal-type signature σ_i for message $(\hat{F}_1^{m_i}, \hat{F}_2^{m_i}, \hat{U}^{m_i})$, and gives $((\hat{F}_1^{m_i}, \hat{F}_2^{m_i}, \hat{U}^{m_i}), \sigma_i, m_i)$ to $\mathcal{A}$.

If adversary $\mathcal{A}$ outputs a simulation-type forgery $S_1 := (G^\alpha V^r) \cdot G^{-a\gamma}$, $S_2 := ((V')^r G^{-z}) \cdot G^\gamma$, $S_3 := (G^b)^{-z}$, $S_4 := (G^b)^{r_2}$, $S_5 := G^{r_1}$, and $S_0 := (\tilde{M}_3\tilde{H})^{r_1}$, for some $r_1, r_2, z, \gamma \in \mathbb{Z}_p$ $(r = r_1 + r_2)$ for message $msg = (\hat{F}_1^m, \hat{F}_2^m, \hat{U}^m)$, then $\mathcal{B}_1$ can compute $(G^{a\gamma}, G^\gamma)$ from S_1, S_2, respectively. The reason is as follows:

$\mathcal{B}_1$ has b, so it can compute G^z, G^{r_1}, G^{r_2} from $S_3 = G^{bz}$, $S_5 = G^{r_1}$, $S_4 = G^{br_2}$, respectively and obtain $G^r = G^{r_1+r_2}$, $V^r = G^{rv}$, $(V')^r = G^{rv'}$ ($\mathcal{B}_1$ has v, v'). Thus, $\mathcal{B}_1$ can extract $(G^{a\gamma}, G^\gamma)$ from S_1 and S_2 since it has α. $\mathcal{B}_1$ can solve the DDH$_2$ problem by checking whether

$$e(G^\gamma, \tilde{Z}) = e\left(G^{a\gamma}, \hat{G}^s\right)$$

or not because $e(G^{a\gamma}, \hat{G}^s) = e(G, \hat{G})^{as\gamma} = e(G^\gamma, \hat{G}^{as})$. If $\hat{Z} = \hat{G}^{as}$ (DDH tuple), then the equation holds. Thus, $\mathcal{B}_1$ solves the DDH$_2$ problem whenever the adversary outputs a valid simulation-type forgery, i.e., $p_0^{\mathrm{sim}}(\lambda) \leq \mathrm{Adv}^{\mathsf{ddh2}}_{\mathcal{G},\mathcal{B}_1}(\lambda)$ as claimed.

Proof of Lemma 9. Given access to $\mathcal{A}$ playing $p_{i-1}^{\mathrm{norm}}(\lambda)$ and $p_i^{\mathrm{norm}}(\lambda)$, we construct algorithm $\mathcal{B}_2$ that solves the XDLIN$_1$ problem with advantage $|p_{i-1}^{\mathrm{norm}}(\lambda) - p_i^{\mathrm{norm}}(\lambda)|$.

$\mathcal{B}_2$ is given instance $(\Lambda, G_1, G_2, G_3, \hat{G}_1, \hat{G}_2, \hat{G}_3, X, Y, \hat{X}, \hat{Y}, Z \in \mathbb{G}_1)$ of the XDLIN$_1$ problem. It implicitly holds that $G_1 = G_2^b$, $\hat{G}_1 = \hat{G}_2^b$, $X = G_1^x$, $Y = G_2^y$, $\hat{X} = \hat{G}_1^x$, $\hat{Y} = \hat{G}_2^y$ for some $b, x, y \in \mathbb{Z}_p$. $\mathcal{B}_2$ generates the group elements in gk and vk as follows: It selects exponents $\xi, \beta, \chi_1, \chi_2, \varphi \leftarrow \mathbb{Z}_p^*$ such that $\xi m + \beta = 0$ where $m \in \mathbb{Z}_p$ is the exponent of the ith random message (If $\xi m + \beta = 0$, then it holds that $(\hat{U}^m \tilde{H}) = \hat{G}_2^{m\chi_1+\chi_2} \hat{G}_3^{\xi m+\beta} = \hat{G}_2^{m\chi_1+\chi_2}$. Note that ξ and β are information theoretically hidden even given m, so the adversary has only negligible chance of producing another message $\hat{U}^{m^*}$ such that $\xi m^* + \beta = 0$). It then computes $G := G_2$, $\hat{G} := \hat{G}_2$, $F_1 := G_1^{\varphi}$, $\hat{F}_1 := \hat{G}_1^{\varphi}$, $F_2 := G_3$, $\hat{F}_2 := \hat{G}_3$, $U := G_2^{\chi_1} G_3^{\xi}$, $\hat{U} := \hat{G}_2^{\chi_1} \hat{G}_3^{\xi}$, sets into gk, and then compute $\tilde{H} := \hat{G}_2^{\chi_2} \hat{G}_3^{\beta}$. It also chooses $a, \delta, v' \leftarrow \mathbb{Z}_p^*$ and computes $V := G_3^{-a\delta}$, $V' := G_3^{\delta} G_2^{v'}$, $\hat{V} := \hat{G}_3^{-a\delta}$, $\hat{V}' := \hat{G}_3^{\delta} \hat{G}_2^{v'}$. Next it chooses $\alpha, \rho \leftarrow \mathbb{Z}_p^*$, computes

$$\begin{aligned}
&\tilde{B} := \hat{G}_1, \quad \tilde{A} := \hat{G}_2^a, \quad \tilde{B}_a := \hat{G}_1^a, \quad \tilde{R} := \hat{V}(\hat{V}')^a = \hat{G}_2^{v'a}, \\
&\tilde{W} := (\hat{V}(\hat{V}')^a)^b = \hat{G}_1^{v'a}, \\
&X_1 := G_2^{\rho}, \quad \tilde{X}_2 := (\hat{G}_1)^{\alpha/\rho}, \quad K_1 := G_2^{\alpha}, \quad K_2 := G_2^b = G_1,
\end{aligned}$$

and them sets them into vk and sk, accordingly.

Since $\mathcal{B}_2$ has a, it can compute G^a and further generate simulation-type signatures. Now $\mathcal{B}_2$ simulates signatures for jth random message as follows.

Case $j > i$: $\mathcal{B}_2$ randomly selects $m_j \in \mathbb{Z}_p$, generates normal-type signature σ_j for message $(\hat{F}_1^{m_j}, \hat{F}_2^{m_j}, \hat{U}^{m_j})$ by using $sk = (vk, G_2^{\alpha}, G_2^b, V, V')$, and gives $((\hat{F}_1^{m_j}, \hat{F}_2^{m_j}, \hat{U}^{m_j}), \sigma_j, m_j)$ to $\mathcal{A}$.

Case $j = i$: $\mathcal{B}_2$ embeds the instance as follows. For the ith randomly chosen message $msg = (\hat{F}_1^m, \hat{F}_2^m, \hat{U}^m) \in \mathbb{G}_2^3$, $\mathcal{B}_2$ implicitly sets $r_1 := y, r_2 := x$ and computes $S_4 := G^{br_2} = G_1^x$, $S_5 := G^{r_1} = G_2^y$. $\mathcal{B}_2$ can compute $\tilde{S}_0 := (\hat{G}_2^y)^{m\chi_1+\chi_2} = (\hat{U}^m \tilde{H})^{r_1}$. Next, in order to compute V^r and $(V')^r$, $\mathcal{B}_2$ computes $(G_3^{r_1+r_2})^{-a\delta}$ as $Z^{-a\delta}$. If $Z = G_3^{x+y}$, then this will be correct. If $Z = G_3^{\zeta}$ for $\zeta \leftarrow \mathbb{Z}_p$, then we let $G^{\gamma} := G_3^{\delta(\zeta-(x+y))}$ and this will be a simulation-type signature. $\mathcal{B}_2$ chooses $s \leftarrow \mathbb{Z}_p$ and implicitly sets $G^{-z} := G_2^{-v'r_2+s}$. These value are not computable but $\mathcal{B}_2$ can compute $G^{zb} = G_1^{xv'-s}$. $S_2 := (G_2^y)^{v'} Z^{\delta} G_2^s = G_2^{r_1 v'+r_2 v'} Z^{\delta} G_2^{s-r_2 v'} = G_2^{rv'} Z^{\delta} G^{-z}$. $\mathcal{B}_2$ generates a signature $\sigma := (\tilde{S}_0, \ldots, S_5)$ as follows:

$$\begin{array}{lll}
\tilde{S}_0 := (\hat{G}_2^y)^{m\chi_1+\chi_2} & S_1 := G_2^{\alpha} Z^{-a\delta} & S_2 := (G_2^y)^{v'} Z^{\delta} G_2^s \\
S_3 := (G_1^x)^{v'} G_1^{-s} & S_4 := G_1^x & S_5 := G_2^y.
\end{array}$$

$\mathcal{B}_2$ can generate S_0 correctly since $\mathcal{B}_2$ set $\xi m+\beta=0$. $\mathcal{B}_2$ gives $((\hat{F}_1^m, \hat{F}_2^m, \hat{U}^m), \sigma, m)$ to $\mathcal{A}$.

- If $Z = G_3^{x+y} \in \mathbb{G}_1$, the above signature is a normal-type signature with $Z = G_3^r$, $S_1 = G_2^{\alpha} G_3^{-a\delta r} = G_2^{\alpha} V^r$, and $S_2 = (G_2^{v'} G_3^{\delta})^r G^{-z} = (V')^r G^{-z}$.

- If $Z \leftarrow \mathbb{G}_1$, the above signature is a simulation-type signature since $Z = G_3^{\zeta}$ for some $\zeta \leftarrow \mathbb{Z}_p$, $S_1 = G_2^{\alpha} G_3^{-a\delta r} G_3^{-a\delta\zeta} G_3^{a\delta r} = G_2^{\alpha} V^r G_3^{-a\delta(\zeta-(x+y))} = G^{\alpha} V^r G^{-a\gamma}$ since $G_3^{\delta(\zeta-(x+y))} = G^{\gamma}$, and $S_2 = G_2^{rv'} G_3^{r\delta} G_3^{\delta(\zeta-(x+y))} G^{-z} = (V')^r G^{\gamma} G^{-z}$.

Case $j < i$: $\mathcal{B}_2$ randomly selects $m_j \in \mathbb{Z}_p$, generates simulation-type signature σ_j for message $(\hat{F}_1^{m_j}, \hat{F}_2^{m_j}, \hat{U}^{m_j})$ by using sk and G_2^a, and gives $((\hat{F}_1^{m_j}, \hat{F}_2^{m_j}, \hat{U}^{m_j}), \sigma_j, m_j)$ to $\mathcal{A}$.

If $Z = G_3^{x+y}$ (linear), then $\mathcal{A}$ is in $p_{i-1}^{\text{norm}}(\lambda)$, otherwise $\mathcal{A}$ is in $p_i^{\text{norm}}(\lambda)$. For all messages, $\mathcal{B}_2$ can return $\mu(M_i) = m_i$.

At some point, $\mathcal{A}$ outputs forgery $(\tilde{S_0^*}, S_1^*, \ldots, S_5^*)$ and message $msg^* = (\tilde{Q}_1, \tilde{Q}_2, \tilde{Q}_3) = (\hat{F}_1^{m^*}, \hat{F}_2^{m^*}, \hat{U}^{m^*})$. $\mathcal{B}_2$ outputs 1 if and only if

$$e\left(G_1, \tilde{S_0}\right) \cdot e\left(S_4, \tilde{Q}_2^{\xi} \hat{G}_3^{\beta}\right) = e\left(\left(S_1 G_2^{-\alpha a_1}\right)^{1/(-a\delta)}, \left(\tilde{Q}_1^{1/\varphi}\right)^{\xi} \hat{G}_1^{\beta}\right) \cdot e\left(S_5, \left(\tilde{Q}_1^{1/\varphi}\right)^{\chi_1} \hat{G}_1^{\chi_2}\right).$$

By Lemma 7, there exist $m^*, r_1^*, r_2^*, \gamma^*,\ r^* = r_1^* + r_2^*$ such that $\tilde{S_0} = (\hat{U}^{m^*} \tilde{H})^{r_1^*},\ S_1 = G_2^{\alpha} V^{r^*} G_2^{-a\gamma^*},\ S_4 = G_1^{r_2^*},\ S_5 = G_2^{r_1^*},\ \hat{Q}_1 = (\hat{G}_1^{\varphi})^{m^*},\ \hat{Q}_2 = \hat{G}_3^{m^*}$. Rephrased in terms of our parameters, this means

$$\tilde{S_0} = \left(\hat{G}_2^{m^*\chi_1+\chi_2} \hat{G}_3^{\xi m^*+\beta}\right)^{r_1^*} \qquad S_1 = G_2^{\alpha} {G_3}^{-a\delta r^*} G_2^{-a\gamma^*}$$
$$S_4 = {G_1}^{r_2^*} \qquad S_5 = G_2^{r_1^*}.$$

Plugging this into the above computation, we have the left-hand side is

$$\begin{aligned} e\left(G_1, \tilde{S_0}\right) \cdot e\left(S_4, \hat{Q}_2^{\xi} \hat{G}_3^{\beta}\right) &= e\left(G_1, \left(\hat{G}_2^{m^*\chi_1+\chi_2} \hat{G}_3^{\xi m^*+\beta}\right)^{r_1^*}\right) \cdot e\left(G_1^{r_2^*}, \left(\hat{G}_3^{m^*}\right)^{\xi} \hat{G}_3^{\beta}\right) \\ &= e\left(G_1, \hat{G}_2\right)^{(m^*\chi_1+\chi_2)r_1^*} e\left(G_1, \hat{G}_3\right)^{(\xi m^*+\beta)r_1^*} \\ &\quad \times e\left(G_1, \hat{G}_3\right)^{(\xi m^*+\beta)r_2^*} \end{aligned}$$

and the right-hand side is

$$\begin{aligned} & e\left(\left(S_1 G_2^{-\alpha}\right)^{1/(-a\delta)}, \left(\hat{Q}_1^{1/\varphi}\right)^{\xi} \hat{G}_1^{\beta}\right) \cdot e\left(S_5, \left(\hat{Q}_1^{1/\varphi}\right)^{\chi_1} \hat{G}_1^{\chi_2}\right) \\ &= e\left(G_3^{r^*} G_2^{\gamma^*/\delta}, \hat{G}_1^{\xi m^*+\beta}\right) \cdot e\left(G_2^{r_1^*}, \hat{G}_1^{m^*\chi_1+\chi_2}\right) \\ &= e\left(G_3, \hat{G}_1\right)^{(\xi m^*+\beta)r^*} e\left(G_2, \hat{G}_1\right)^{\gamma^*/\delta(\xi m^*+\beta)} e\left(G_2, \hat{G}_1\right)^{(m^*\chi_1+\chi_2)r_1^*}. \end{aligned}$$

A simplified equation is $1 = e(G_2, \hat{G}_1)^{\gamma^*/\delta(\xi m^* + \beta)}$.

Thus, the difference of $\mathcal{A}$'s advantage in two games gives the advantage of $\mathcal{B}_2$ in solving the XDLIN$_1$ problem as stated.

Proof of Lemma 10. Observe that, in $p_q^{\mathrm{norm}}(\lambda)$, $\mathcal{A}$ is given simulation-type signatures only. We show that if $\mathcal{A}$ outputs a normal-type forgery in $p_q^{\mathrm{norm}}(\lambda)$ then we can construct algorithm $\mathcal{B}_3$ that solves the co-CDH problem.

$\mathcal{B}_3$ is given instance $(\Lambda, G, \hat{G}, G^x, G^y, \hat{G}^x, \hat{G}^y)$ of the co-CDH problem. $\mathcal{B}_3$ generates the verification key as follows: $\mathcal{B}_3$ selects exponents $u, h, f_1, f_2 \leftarrow \mathbb{Z}_p^*$, computes $F_1 := G^{f_1}$, $\hat{F}_1 := \hat{G}^{f_1}$, $F_2 := G^{f_2}$, $\hat{F}_2 := \hat{G}^{f_2}$, $U := G^u$, $\hat{U} := \hat{G}^u$, and sets them into gk. $\mathcal{B}_3$ also selects exponents $v, v' \leftarrow \mathbb{Z}_p^*$, computes $V := G^v$, $V' := G^{v'}$, $\hat{V} := \hat{G}^v$, $\hat{V}' := \hat{G}^{v'}$. Next, it also selects exponents $h, b, \rho' \leftarrow \mathbb{Z}_p^*$, computes $\tilde{H} := \hat{G}^h$ and

$$\tilde{B} := \hat{G}^b, \quad \tilde{A} := \hat{G}^y, \quad \tilde{B}_a := (\hat{G}^y)^b, \quad \tilde{R} := \hat{V}(\hat{V}')^a = \hat{V}(\hat{G}^y)^{v'},$$
$$\tilde{W} := \tilde{R}^b = (\hat{V}(\hat{G}^y)^{v'})^b$$
$$X_1 := (G^x)^{\rho'}, \quad \tilde{X}_2 := (\hat{G}^y)^{b/\rho'}, \quad K_2 := G^b,$$

and sets them into vk and sk, accordingly. Note that it means implicitly $\rho = \rho' x$ and $\alpha = xy$ though $\mathcal{B}_3$ does not have α, ρ. Therefore $\mathcal{B}_3$ does not have $K_1 = G^\alpha = G^{xy}$, and cannot compute normal-type signatures. For ith message, $\mathcal{B}_3$ randomly select $m_i \in \mathbb{Z}_p$ and outputs simulation-type signatures for each random message $msg_i = (\hat{F}_1^{m_i}, \hat{F}_2^{m_i}, \hat{U}^{m_i})$ as follows:

$\mathcal{B}_3$ selects $r_1, r_2, z, \gamma' \leftarrow \mathbb{Z}_p$, sets $r := r_1 + r_2$ (we want to set $\gamma := x + \gamma'$), and computes:

$$S_1 := (G^y)^{-\gamma'} \cdot V^r = (G^\alpha V^r) \cdot G^{-a\gamma} \ (a = y, xy = \alpha)$$
$$S_2 := G^{\gamma'} G^x (V')^r G^{-z} = ((V')^r G^{-z}) \cdot G^\gamma$$
$$S_3 := (G^b)^z \qquad S_4 := G^{r_2 b} \qquad S_5 := G^{r_1} \qquad \tilde{S}_0 := (\hat{U}^{m_i} \tilde{H})^{r_1}.$$

$\mathcal{B}_3$ gives $((\hat{F}_1^{m_i}, \hat{F}_2^{m_i}, \hat{U}^{m_i}), \sigma_i, m_i)$ where $\sigma_i := (\tilde{S}_0, S_1, \ldots, S_5)$ to $\mathcal{A}$.

At some point, $\mathcal{A}$ outputs a normal-type forgery, $S_1^* = G^\alpha V^{r^*}$, $S_2^* = (V')^{r^*} G^{-z^*}$, $S_3^* = (G^b)^{z^*}$, $S_4^* = G^{r_2^* b}$, $S_5^* = G^{r_1^*}$, and $\tilde{S}_0^* = (\hat{U}^{m^*} \tilde{H})^{r_1^*}$, for some $r_1^*, r_2^*, z^*, \in \mathbb{Z}_p$ for message $msg^* = (\hat{F}_1^{m^*}, \hat{F}_2^{m^*}, \hat{U}^{m^*})$.

By using these values, $\mathcal{B}_3$ can compute $G^{r_2^*} = (S_4^*)^{1/b}$, $G^{r_1^*} = S_5^*$, $G^{z^*} = (S_3^*)^{1/b}$, $V^{r^*} = (G^{r_1^*} \cdot G^{r_2^*})^v$ since $V = G^v$. Thus, $\mathcal{B}_3$ can compute $S_1^*/V^{r^*} = G^\alpha = G^{xy}$. That is, $\mathcal{B}_3$ can solve the co-CDH problem and it holds that $p_q^{\mathrm{norm}}(\lambda) \leq \mathrm{Adv}_{\mathcal{G},\mathcal{B}_3}^{\textsf{co-cdh}}(\lambda)$ as claimed.

Remark 3. It is difficult to modify $\textsf{xSIG}$ so as to rely on the DDH$_1$ and DDH$_2$ assumption, that is, only on the SXDH assumption because we are not given instances in group $\mathbb{G}_2$ and cannot simulate verification keys in group $\mathbb{G}_2$ under the DDH$_1$ assumption when we prove a similar statement to Lemma 9 by using DDH$_1$. Constructing XRMA-secure

SPS scheme only from the SXDH assumption is an important open problem since it will save on the number of group elements in a signature and a verification key. Moreover, it is non-trivial to modify xSIG so as to rely on the DDH_1 and XDLIN_1 because if we use assumptions only over $\mathbb{G}_1$, then all elements in a signature must be in $\mathbb{G}_1$. It means that a message must consist of elements in both $\mathbb{G}_1$ and $\mathbb{G}_2$, which we would like to avoid.

6.5. *Security and Efficiency of Resulting* SIG2

Let $\mathsf{SIG2}$ be the scheme obtained from POSb and xSIG. $\mathsf{SIG2}$ is structure-preserving as vk, σ, and msg consist of group elements from $\mathbb{G}_1$ and $\mathbb{G}_2$, and $\mathsf{SIG2.Vrf}$ evaluates pairing product equations. From Theorems 3, 10 and 11, we obtain the following theorem.

Theorem 12. $\mathsf{SIG2}$ *is a structure-preserving signature scheme that is unforgeable against adaptive chosen message attacks if SXDH and* XDLIN_1 *hold for* $\mathcal{G}$. *In particular, for any p.p.t. algorithm* $\mathcal{A}$ *for* $\mathsf{SIG2}$ *making at most* $q_s(\lambda)$ *signing queries, there exist p.p.t. algorithms* $\mathcal{B}, \mathcal{C}$ *such that* $\mathrm{Adv}^{\mathsf{uf\text{-}}cma}_{\mathsf{SIG2},\mathcal{A}}(\lambda) \leq (q_s(\lambda)+1) \cdot \mathrm{Adv}^{\mathsf{dlin}}_{\mathcal{G},\mathcal{B}}(\lambda) + 2 \cdot \mathrm{Adv}^{\mathsf{sxdh}}_{\mathcal{G},\mathcal{C}}(\lambda) + 2/p(\lambda)$, *where* $p(\lambda)$ *is the size of the groups produced by* $\mathcal{G}$.

Table 3 summarizes the efficiency of $\mathsf{SIG2}$ for both unilateral messages consisting of k elements and bilateral messages consisting of k_1 and k_2 elements in $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. We count the number of group elements in public components of $\mathsf{SIG2}$. Note that the default generators in gk is not included in the count. For comparison, we also evaluate the efficiency of the schemes in [4, Section 5.2] and [5, Section 5.2]. For bilateral messages, the scheme from [4] is combined with POSb from Sect. 6.3. Since the scheme in [4] can sign a single group element, extended part of one-time verification key from $\mathsf{POSb.Update}$ can be dropped and gk need to include only one generator for each $\mathbb{G}_1$ and $\mathbb{G}_2$.

In Tables 4 and 5, we assess the size of proofs for showing ones possession of a valid signature and message of $\mathsf{SIG2}$ by using the GS proof system as NIWI or NIZK. The general formulas are the same as those in (14)–(17) except that witnesses and linear equations in $\mathbb{G}_1$ and $\mathbb{G}_2$ are considered separately (We say that an equation is linear in $\mathbb{G}_1$ if all variables in the equation are in $\mathbb{G}_1$). By (x, y), we denote x and y elements in

Table 3. Efficiency of SIG2 and comparison to other schemes with constant-size signatures.

Scheme	$\lvert msg \rvert$	$\lvert gk \rvert + \lvert vk \rvert$	$\lvert \sigma \rvert$	#(PPE)	Assumptions
[4]	$(k_1, 0)$	$(5, 2k_1+9)$	$(5, 2)$	2	q-SFP
[5]	$(k_1, 0)$	$(1, k_1+4)$	$(3, 1)$	2	q-type
SIG2 : POSu1 + xSIG	$(k_1, 0)$	$(5, k_1+12)$	$(7, 4)$	5	SXDH, XDLIN_1
POSb + [4]	(k_1, k_2)	(k_2+12, k_1+7)	$(8, 5)$	4	q-SFP
[5]	(k_1, k_2)	(k_2+3, k_1+4)	$(3, 3)$	2	q-type
SIG2 : POSb + xSIG	(k_1, k_2)	(k_2+6, k_1+13)	$(8, 6)$	6	SXDH, XDLIN_1

The upper half is for unilateral messages and the lower half is for bilateral messages. Notation (x, y) represents x elements in $\mathbb{G}_1$ and y in $\mathbb{G}_2$

Table 4. Costs of WI proofs with the GS proof system of valid signature of SIG2 for unilateral and bilateral messages.

SIG2	$\lvert\mathrm{NIWI}(\sigma)\rvert$	$\lvert\mathrm{NIWI}(\sigma, msg)\rvert$
Unilateral	$(26, 18, 0)$	$(2k_1 + 26, 18, 0)$
Bilateral	$(32, 26, 0)$	$(2k_1 + 32, 2k_2 + 26, 0)$

Entry (x, y, z) denotes x, y, and z elements in $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{Z}_p$, respectively

Table 5. Costs for proving valid signature of SIG2 for unilateral and bilateral messages in ZK with the GS proof system.

SIG2	$\lvert\mathrm{NIZK}(\sigma)\rvert$	$\lvert\mathrm{NIZK}(\sigma, msg)\rvert$
Unilateral	$(2k_1 + 28, 18, 2k_1 + 2)$	$(2k_1 + 28, 18, 2)$
Bilateral	$(2k_1 + 34, 2k_2 + 26, 2k_1 + 2k_2 + 2)$	$(2k_1 + 34, 2k_2 + 26, 2)$

$\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Similarly, by (x, y, z), we denote additional element z in $\mathbb{Z}_p$. In this asymmetric setting, we have $|com| = (2, 0, 0)$ for committing to $\mathbb{G}_1$ elements, and $|com| = (0, 2, 0)$ for $\mathbb{G}_2$. Proof size for linear equation in $\mathbb{G}_1$ and $\mathbb{G}_2$ is $|\pi_L| = (0, 2, 0)$ and $(2, 0, 0)$, respectively. We also have $|\pi_{NL}| = (4, 4, 0)$ and $|\pi_{MS}| = (0, 0, 2)$.

We first consider the cases of NIWI shown in Table 4. For unilateral messages, we have $|\sigma_{\mathrm{wit}}| = (7, 4)$ group elements and $|\sigma_{\mathrm{rnd}}| = (0, 0)$. Verifying POSu1 consists of one nonlinear relation (19), and verifying xSIG consists of one linear equation in $\mathbb{G}_1$ (23), two linear equations in $\mathbb{G}_2$ (25, 26) and one nonlinear equation (24). Thus, $|\mathrm{NIWI}(\sigma)| = ((2, 0, 0) \times 7 + (0, 2, 0) \times 4) + 0 + (4, 4, 0) \times 2 + ((0, 2, 0) \times 1 + (2, 0, 0) \times 2) = (26, 18, 0)$. For bilateral messages, we have $|\sigma_{\mathrm{wit}}| = (8, 6)$ group elements and $|\sigma_{\mathrm{rnd}}| = (0, 0)$. Verifying POSb consists of verification for POSu1 and POSu2, which are two nonlinear relations in total (They are nonlinear since one-time public-key A is in $\mathbb{G}_1$ whereas signature $\tilde{Z}$, $\tilde{R}$ are in $\mathbb{G}_2$). Equations for xSIG are the same as above. Thus $|\mathrm{NIWI}(\sigma)| = ((2, 0, 0) \times 8 + (0, 2, 0) \times 6) + 0 + (4, 4, 0) \times 3 + ((0, 2, 0) \times 1 + (2, 0, 0) \times 2) = (32, 26, 0)$. For $\mathrm{NIWI}(\sigma, msg)$, we add $(2k_1, 0)$ and $(2k_1, 2k_2)$ elements for the commitment of the message in unilateral and bilateral case, respectively. Hence $|\mathrm{NIWI}(\sigma, msg)| = (2k_1 + 26, 18, 0)$ for unilateral case, and $|\mathrm{NIWI}(\sigma, msg)| = (2k_1 + 32, 2k_2 + 26, 0)$ for bilateral case.

We next consider the cases of NIZK. Additional elements comes from public constants to commit to, and the proof of their correct commitment. For $\mathrm{NIZK}(\sigma)$, every element in a message are regarded as public constants that are input to constant pairings. And xSIG involves one constant pairing $e(X_1, \tilde{X}_2)$ where we commit to X_1 so that (23) remains a linear equation. We thus have $k_1 + 1$ constants to commit to in $\mathbb{G}_1$ for the unilateral case, and $k_1 + 1$ and k_2 constants to commit to in $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively in the bilateral case. By wrapping up, we have $|\mathrm{NIZK}(\sigma)| = |\mathrm{NIWI}(\sigma)| + (2, 0, 0) \times (k_1 + 1) + (0, 0, 2) \times (k_1 + 1) = (2k_1 + 28, 18, 2k_1 + 2)$ for the unilateral case, and $|\mathrm{NIZK}(\sigma)| = |\mathrm{NIWI}(\sigma)| + (2, 0, 0) \times (k_1 + 1) + (0, 2, 0) \times k_2 + (0, 0, 2) \times (k_1 + k_2 + 1) = (32, 26, 0) + (2k_1 + 2, 0, 0) + (0, 2k_2, 0) + (0, 0, 2k_1 + 2k_2 + 2) = (2k_1 + 34, 2k_2 + 26, 2k_1 + 2k_2 + 2)$ for the bilateral case. For $\mathrm{NIZK}(\sigma, msg)$ where messages are already committed, additional elements are from committing to X_1 compared to the case of $\text{NIWI}(\sigma, msg)$. We thus have $|\text{NIZK}(\sigma, msg)| = |\text{NIWI}(\sigma, msg)| + (2, 0, 0) \times 1 + (0, 0, 2) \times 1 = (2k_1 + 28, 18, 2)$ for unilateral case, and $|\text{NIZK}(\sigma, msg)| = |\text{NIWI}(\sigma, msg)| + (2, 0, 0) \times 1 + (0, 0, 2) \times 1 = (2k_1 + 34, 2k_2 + 26, 2)$ for bilateral case.

7. Applications

We list a few recent examples of applications of SPS that benefit from our results.

- *Group Signatures with Efficient Revocation and Compact Verifiable Shuffles.* Using our SIG1 scheme from Sect. 5 both the construction of a group signature scheme with efficient revocation by Libert et al. [36] and the construction of compact verifiable shuffles by Chase et al. [18] can be proven purely under the DLIN assumption. All other building blocks already have efficient instantiations based on DLIN.
- *Tightly-secure Structure-preserving Signatures.* Hofheinz and Jager [34] construct a tightly-secure one-time signature scheme and use it to construct s tightly-secure tree-based SPS scheme, say tSIG. Instead, we propose to use our partial one-time scheme to construct tSIG. As the resulting tSIG is secure against non-adaptive chosen message attacks, it is secure against extended random message attacks as well. We then combine the POSb scheme and the new tSIG scheme according to our second generic construction. The resulting signature scheme is significantly more efficient than [34] and is a SPS scheme with a tight security reduction to SXDH. As shown in [3], the same is possible in Type-I groups by using the tagged one-time signature scheme in Sect. 5.2 whose security tightly reduced to DLIN.
- *Simulation-sound and Simulation-extractable NIZK.* In [3], we also show how to construct more efficient simulation-sound and simulation-extractable non-interactive zero-knowledge (SS-NIZK & SE-NIZK) proof systems. While in [3] we were primarily interested in tightly-secure NIZK and thus used the tree-based tSIG scheme, RMA-security suffices for constructing unbounded SS-NIZK and SE-NIZK schemes. Our rSIG and xSIG schemes can thus be used directly to construct even more efficient unbounded SE-NIZK if one lifts the requirement of a tight reduction.
- *Tightly-secure Structure-preserving CCA-secure Public-key Encryption.* Following the approach of [34] and [3], tightly-secure SE-NIZK enables tightly-secure and structure-preserving CCA-secure public-key encryption under standard decisional assumptions.
- *Efficient Adaptive Oblivious Transfer.* Hohenberger and Green proposed a universally composable (UC) adaptive oblivious transfer (AOT) protocol by using an SPS scheme based on a q-type assumption [30]. Thus their protocol relies on a q-type assumptions and constructing an efficient UC AOT protocol from only standard assumptions was an open problem. As a corollary of our result, we can obtain a UC AOT protocol based on only standard assumptions by replacing their SPS scheme with ours.

  As an application of our schemes, Abe, Camenisch, Dubovitskaya, and Nishimaki proposed a UC AOT with hidden access control protocol from standard assumptions by using our schemes [1]. Moreover, they proposed an XRMA-secure SPS scheme only from the SXDH assumption based on another (non-structure-preserving) signature scheme by Chen et al. [19]. However, their scheme is less efficient than ours since their construction technique is different from ours and their message space is large.

8. Conclusions and Open Questions

We showed generic framework for constructing SPS by refining the Even–Goldreich–Micali framework with novel notions and primitives such as extended random message attacks and tagged one-time signature schemes. By instantiating them, we presented constant-size SPS consisting of only 11–14 group elements based on simple assumptions such as DLIN for symmetric pairings and analogues of DDH and XDLIN for asymmetric pairings. Our approach is modular and divides the problem into the need to construct constant-size RMA/XRMA-secure SPS and constant-size structure-preserving one-time signatures. This is in line with the promise of [7] that SPS enable modular protocol design. Indeed this modularity facilitates applications in which one can cherry pick primitives according to requirements.

A tight bound for the size of SPS under simple assumptions is an important open question, and would shed light on the overhead of such a modular approach. It is also still an open question to construct efficient RMA/XRMA-secure SPS schemes from only the SXDH assumption. Similarly, constructing (X)RMA-secure schemes with a message space that is a simple Cartesian product of groups without sacrificing efficiency and constructing more efficient RMA-secure schemes, which may not necessarily be XRMA-secure are interesting open problems. All RMA-secure signature schemes developed in this paper are in fact XRMA-secure.

Finally, it is also an interesting open problem to design a constant-size SPS scheme with tight security under simple assumptions. For the hybrid argument in the security proof, our concrete constructions suffer the security loss in the number of sining queries.

Appendix: Waters' Dual System Signature Scheme

We review Waters' dual system signature scheme [44] in this section.

[Scheme WdSIG]

WdSIG.Key(gk): Given $gk := (\Lambda, G)$ as input, sample V, V_1, V_2, H, I, U uniformly from $\mathbb{G}^*$ and a_1, a_2, b, and α from $\mathbb{Z}_p^*$. Then compute

$$\begin{array}{lllll} B := G^b, & A_1 := G^{a_1}, & A_2 := G^{a_2}, & B_1 := G^{b\cdot a_1}, & B_2 := G^{b\cdot a_2} \\ R_1 := V V_1^{a_1}, & R_2 := V V_2^{a_2}, & W_1 := R_1^b, & W_2 := R_2^b, & \\ T := e(G,G)^{\alpha\cdot a_1\cdot b} & K_1 := G^{\alpha}, & K_2 := G^{\alpha\cdot a_1}, & & \end{array}$$

and output $vk := (B, A_1, A_2, B_1, B_2, R_1, R_2, W_1, W_2, H, I, U, T)$ and $sk := (vk, K_1, K_2, V, V_1, V_2)$.

$\mathsf{WdSIG.Sign}(sk, msg)$: Parse sk into $(vk, K_1, K_2, V, V_1, V_2)$. Also parse vk accordingly. For $msg \in \mathbb{Z}_p$, pick random $r_1, r_2, z_1, z_2, \mathsf{tag}_k \in \mathbb{Z}_p$. Let $r = r_1 + r_2$. Compute and output signature $\sigma := (S_1, \ldots S_7, S_0, \mathsf{tag}_k)$ where

$$\begin{array}{llll} S_1 := K_2 V^r, & S_2 := K_1^{-1} V_1^r G^{z_1}, & S_3 := B^{-z_1}, & S_4 := V_2^r G^{z_2}, \\ S_5 := B^{-z_2}, & S_6 := B^{r_2}, & S_7 := G^{r_1}, & S_0 := (U^{msg} I^{\mathsf{tag}_k} H)^{r_1}. \end{array}$$

$\mathsf{WdSIG.Vrf}(vk, \sigma, msg)$: Parse σ into $(S_1, \ldots, S_7, S_0, \mathsf{tag}_k)$. Also parse vk accordingly. Pick random s_1, s_2, t and tag_c from $\mathbb{Z}_p$, compute

$$C_1 := B^{s_1+s_2}, \quad C_2 := B_1^{s_1}, \quad C_3 := A_1^{s_1}, \quad C_4 := B_2^{s_2},$$
$$C_5 := A_2^{s_2}, \quad C_6 := R_1^{s_1} R_2^{s_2}, \quad C_7 := W_1^{s_1} W_2^{s_2}, \quad E_1 := (U^{msg} I^{\mathsf{tag}_c} H)^{r_1},$$
$$E_2 := G^t,$$

and if $\mathsf{tag}_c - \mathsf{tag}_k \neq 0$, verify

$$\begin{aligned} & e(C_1, S_1) \cdot e(C_2, S_2) \cdot e(C_3, S_3) \cdot e(C_4, S_4) \cdot e(C_5, S_5), \\ & \quad = e(C_6, S_6) \cdot e(C_7, S_7) \cdot (e(E_1, S_7)/e(E_2, S_0))^{1/(\mathsf{tag}_c - \mathsf{tag}_k)} \cdot T^{s_2}. \end{aligned}$$

References

[1] M. Abe, J. Camenisch, M. Dubovitskaya, R. Nishimaki, Universally composable adaptive oblivious transfer (with access control) from standard assumptions, in *DIM'13, Proceedings of the 2013 ACM Workshop on Digital Identity Management, Berlin, Germany* (ACM, 2013), pp. 1–12

[2] M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, M. Ohkubo, Constant-size structure-preserving signatures generic constructions and simple assumptions, in *Advances in Cryptology—ASIACRYPT 2012*, volume 7658 of LNCS, ed. by X. Wang, K. Sako (Springer, Berlin, 2012), pp. 4–12,

[3] M. Abe, B. David, M. Kohlweiss, R. Nishimaki, M. Ohkubo, Tagged one-time signatures: tight security and optimal tag size, in *Public-Key Cryptology—PKC 2013*, volume 7778 of LNCS, ed. by K. Kurosawa, G. Hanaoka (Springer, Berlin, 2013), pp. 312–331

[4] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. Ohkubo, Structure-preserving signatures and commitments to group elements. *J. Cryptol.*, (2015). doi:10.1007/s00145-014-9196-7

[5] M. Abe, J. Groth, K. Haralambiev, M. Ohkubo, Optimal structure-preserving signatures in asymmetric bilinear groups, in *Advances in Cryptology—CRYPTO '11*. LNCS (Springer, Berlin, 2011)

[6] M. Abe, J. Groth, M. Ohkubo, Separating short structure preserving signatures from non-interactive assumptions, in *Advances in Cryptology—ASIACRYPT 2011*, volume 7073 of LNCS, ed. by D. H. Lee, X. Wang (Springer, Berlin, 2011), pp. 628–646

[7] M. Abe, K. Haralambiev, M. Ohkubo, Signing on group elements for modular protocol designs. IACR ePrint Archive, Report 2010/133, 2010. http://eprint.iacr.org

[8] M. Abe, M. Ohkubo, A framework for universally composable non-committing blind signatures. IJACT, 2(3), 229–249 (2012).

[9] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, H. Shacham, Randomizable proofs and delegatable anonymous credentials, in *Advances in Cryptology—CRYPTO 2009*, volume 5677 of LNCS, ed. by S. Halevi (Springer, Berlin, 2009), pp. 108–125

[10] M. Bellare, D. Micciancio, B. Warinschi, Foundations of group signatures: Formal definitions, simplified requirements and a construction based on general assumptions, in *Advances in Cryptology—EUROCRYPT 2013*, volume 2656 of LNCS, ed. by E. Biham (Springer, Berlin, 2003), pp. 614–629

[11] M. Bellare, H. Shi, C. Zhang, Foundations of group signatures: the case of dynamic groups, in *Topics in Cryptology—CT-RSA 2005,* volume 3376 of LNCS, ed. by A. Menezes (Springer, Berlin, 2005), pp. 136–154. Full version available at IACR e-print 2004/077

[12] M. Bellare, S. Shoup, Two-tier signatures, strongly unforgeable signatures, and Fiat–Shamir without random oracles, in *Public-Key Cryptology—PKC 2007*, volume 4450 of LNCS, ed. by T. Okamoto, X. Wang (Springer, Berlin, 2007), pp. 201–216

[13] D. Boneh, X. Boyen, H. Shacham, Short group signatures, in *Advances in Cryptology—CRYPTO 2004*, volume 3152 of LNCS, ed. by M. Franklin (Springer, Berlin, 2004), pp. 41–55

[14] D. Boneh, C. Gentry, B. Lynn, H. Shacham, Aggregate and verifiably encrypted signatures from bilinear maps, in *Advances in Cryptology—EUROCRYPT 2003*, volume 2656 of LNCS, ed. by E. Biham (Springer, Berlin, 2003), pp. 416–432

[15] J. Camenisch, M. Dubovitskaya, K. Haralambiev, Efficient structure-preserving signature scheme from standard assumptions, in *Security and Cryptography for Networks—SCN 2012*, volume 7485 of LNCS, ed. by I. Visconti, R. De Prisco (Springer, Berlin, 2012), pp. 76–94

[16] J. Cathalo, B. Libert, M. Yung, Group encryption: Non-interactive realization in the standard model, in *Advances in Cryptology—ASIACRYPT 2009*, volume 5912 of LNCS, ed. by M. Matsui (2009), pp. 179–196

[17] M. Chase, M. Kohlweiss, A new hash-and-sign approach and structure-preserving signatures from DLIN, in *Security and Cryptography for Networks-SCN 2012*, volume 7485 of LNCS, ed. by I. Visconti, R. De Prisco (Springer, Berlin, 2012), pp. 131–148

[18] M. Chase, M. Kohlweiss, A. Lysyanskaya, S. Meiklejohn, Malleable proof systems and applications, in *Advances in Cryptology—EUROCRYPT 2012*, volume 7237 of LNCS, ed. by D. Pointcheval, T. Johansson (Springer, Berlin, 2012), pp. 281–300

[19] J. Chen, H. W. Lim, S. Ling, H. Wang, H. Wee, Shorter identity-based encryption via asymmetric pairings. *Des. Codes Cryptogr.*, **73**(3), 911–947 (2014)

[20] D. Dolev, C. Dwork, M. Naor, Nonmalleable cryptography. SIAM J. Comput., 30(2), 391–437 (2000).

[21] C. Dwork, M. Naor, An efficient existentially unforgeable signature scheme and its applications. *J. Cryptol.*, **11**(3), 187–208 (1998)

[22] S. Even, O. Goldreich, S. Micali, On-line/off-line digital signatures. *J. Cryptol.*, **9**(1), 35–67 (1996)

[23] M. Fischlin, Round-optimal composable blind signatures in the common reference model, in *Advances in Cryptology—CRYPTO 2006*, volume 4117 of LNCS, ed. by C. Dwork (Springer, Berlin, 2006), pp. 60–77

[24] G. Fuchsbauer, Commuting signatures and verifiable encryption, in *Advances in Cryptology—EUROCRYPT 2011*, volume 6632 of LNCS, ed. by K. G. Paterson (Springer, Berlin, 2011), pp. 224–245

[25] G. Fuchsbauer, D. Pointcheval, Anonymous proxy signatures, in *Security and Cryptography for Networks—SCN 2008*, volume 5229 of LNCS, ed. by R. Ostrovsky, R. De Prisco, I. Visconti (Springer, Berlin, 2008), pp. 201–217

[26] G. Fuchsbauer, D. Pointcheval, D. Vergnaud, Transferable constant-size fair e-cash, in *Cryptology and Network Security—CANS 2009*, volume 5888 of LNCS, ed. by J.A. Garay, A. Miyaji, A. Otsuka (Springer, Berlin, 2009), pp. 226–247

[27] G. Fuchsbauer, D. Vergnaud, Fair blind signatures without random oracles, in *Progress in Cryptology—AFRICACRYPT 2010*, volume 6055 of LNCS, ed.by D. J. Bernstein, T. Lange (Springer, Berlin, 2010), pp. 16–33

[28] S.D. Galbraith, K.G. Peterson, N.P. Smart, Pairings for cryptographers. *Discrete Appl. Math.*, **156**(16), 3113–3121 (2008)

[29] S. Goldwasser, S. Micali, R. Rivest, A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, **17**(2), 281–308 (1988)

[30] M. Green, S. Hohenberger, Universally composable adaptive oblivious transfer, in *Advances in Cryptology—ASIACRYPT 2008*, volume 5350 of LNCS, ed. by J. Pieprzyk (Springer, Berlin, 2008), pp. 179–197

[31] M. Green, S. Hohenberger, Practical adaptive oblivious transfer from simple assumptions, in *Theory of Cryptography—TCC 2011*, volume 6597 of LNCS, ed. by Y. Ishai (Springer, Berlin, 2011), pp. 347–363

[32] J. Groth, Simulation-sound NIZK proofs for a practical language and constant size group signatures, in *Advances in Cryptology—ASIACRYPT 2006*, volume 4284 of LNCS, ed. by X. Lai, K. Chen (Springer, Berlin, 2006), pp. 444–459

[33] J. Groth, A. Sahai, Efficient noninteractive proof systems for bilinear groups. SIAM J. Comput., 41(5), 1193–1232 (2012).

[34] D. Hofheinz, T. Jager, Tightly secure signatures and public-key encryption, in *Advances in Cryptology—CRYPTO 2012*, volume 7417 of LNCS, ed. by R. Naini, R. Canetti (Springer, Berlin, 2012), pp. 590–607

[35] A. Kiayias, M. Yung, Group signatures with efficient concurrent join, in *Advances in Cryptology—EUROCRYPT 2005*, volume 3494 of LNCS, ed. by R. Cramer (Springer, Berlin, 2005), pp. 198–214

[36] B. Libert, T. Peters, M. Yung, Scalable group signatures with revocation, in *Advances in Cryptology—EUROCRYPT 2012*, volume 7237 of LNCS, ed. by D. Pointcheval, T. Johansson (Springer,Berlin, 2012), pp. 609–627

[37] Y. Lindell, A simpler construction of CCA2-secure public-key encryption under general assumptions. *J. Cryptol.*, **19**(3), 359–377 (2006)

[38] M. Naor, M. Yung, Public-key cryptosystems provably secure against chosen ciphertext attacks, in *Symposium on Theory of Computing(STOC) 1990*, ed. by H. Ortiz (ACM, NY, 1990), pp. 427–437

[39] M. Rückert, D. Schröder, Security of verifiably encrypted signatures and a construction without random oracles, in *Pairing-Based Cryptography—Pairing 2009*, volume 5671 of LNCS, ed. by H. Shacham, B. Waters (Springer, Berlin, 2009), pp. 17–34

[40] A. Sahai, Non-malleable non-interactive zero-knowledge and chosen-ciphertext security, in *Foundations of Computer Science(FOCS) 1999* (IEEE Computer Society, Washington, DC, 1999) pp. 543–553

[41] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, A. Sahai. Robust non-interactive zero knowledge. in *Advances in Cryptology—CRYPTO 2001*, volume 2139 of LNCS, ed. by J. Kilian (Springer, Berlin, 2001), pp. 566–598

[42] A. Shamir, Y. Tauman, Improved online/offline signature schemes, in *Advances in Cryptology—CRYPTO 2001*, volume 2139 of LNCS, ed. by J. Kilian (Springer, Berlin, 2001), pp. 355–367

[43] V. Shoup, Lower bounds for discrete logarithms and related problems, in *Advances in Cryptology—EUROCRYPT 1997*, volume 1233 of LNCS, ed. by W. Fumy (Springer, Berlin, 1997), pp. 256–266

[44] B. Waters, Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions, in *Advances in Cryptology—CRYPTO 2009*, volume 5677 of LNCS, ed. by S. Halevi (Springer, Berlin, 2009), pp. 619–636