Signing on Elements in Bilinear Groups for Modular Protocol Design

Masayuki Abe* Kristiyan Haralambiev**§ Miyako Ohkubo*

* Information Sharing Platform Laboratories, NTT Corporation, Japan
{abe.masayuki,ookubo.miyako}@lab.ntt.co.jp

** Computer Science Department, New York University, U.S.A.
kkh@cs.nyu.edu

Abstract

This paper addresses the construction of signature schemes whose verification keys, messages, and signatures are group elements and the verification predicate is a conjunction of pairing product equations. We answer to the open problem of constructing constant-size signatures by presenting an efficient scheme. The security is proven in the standard model based on a novel non-interactive assumption called Simultaneous Flexible Pairing Assumption that can be justified and has an optimal bound in the generic bilinear group model. We also present efficient schemes with advanced properties including signing unbounded number of group elements, allowing simulation in the common reference string model, signing messages from mixed groups in the asymmetric bilinear group setting, and strong unforgeability. Among many applications, we show two examples; an adaptively secure round optimal blind signature scheme and a group signature scheme with efficient concurrent join. As a bi-product, several homomorphic trapdoor commitment schemes and one-time signature schemes are presented, too. In combination with the Groth-Sahai proof system, these schemes contribute to an efficient instantiation of modular constructions of cryptographic protocols.

Keywords: Digital Signatures, Modular Protocol Design, Groth-Sahai Proofs, Blind Signatures, Group Signatures

§Work done while visiting NTT Information Sharing Platform Laboratories.

Contents

1 Introduction

Background. Cryptographic protocols often allow modular constructions that combine general building blocks such as commitments, encryption, signatures, and zero-knowledge proofs. While modular design is useful to show feasibility of cryptographic tasks and also to illustrate a comprehensible framework, efficient instantiations are sometimes left as a next challenge. Some cryptographic tasks find "cleverly crafted" efficient solutions dedicated for their own purposes. Nevertheless, modular construction makes it easier and can be a good alternative for comparison when the building blocks have reasonable instantiations.

The combination of signatures and non-interactive zero-knowledge proofs of knowledge appears frequently in privacy-protecting cryptographic protocols such as blind signatures [24, 2], group signatures [4, 37, 6], anonymous credential systems [3], verifiably encrypted signatures [10, 45], non-interactive group encryption [22] and so on. An efficient non-interactive proof system in the standard model, however, has been absent until recently. In [35], Groth and Sahai presented the first (and currently the only) efficient non-interactive proof system based on bilinear mapping. Their proof system (GS proofs for short) exerts its full power as a proof of knowledge system when the proof statement is described as a conjunction of relations described by pairing product equations and when the witnesses consists of group elements [1].

Thus it is desired to have signature schemes with properties that (1) the verification keys, messages, and signatures are elements of bilinear groups, and (2) the verification predicate is a conjunction of pairing products. As a non-interactive proof of knowledge system is a very powerful tool in cryptographic protocol design, such a "GS-compatible" signature scheme apparently has numerous applications.

Related Works. Research on signature schemes that were compatible with GS proofs was initiated in [31]. While the design goal is clear and simple, it is not easy to achieve since the requirements proscribe the use of hash functions which usually play a central role to make signature schemes unforgeable against adaptive chosen message attacks, particularly when the messages are not from a specific space such as $\mathbb{Z}_p$. Furthermore, basing only on pairing products in the verification predicate often yields linear dependencies that can be useful in adaptive attacks. There are efficient signature schemes, e.g., [8, 18, 3, 16], whose all but one components are group elements and verification predicates are pairing product equations. These schemes are sufficient for specific purposes but would not for others. In [31], Groth first showed the feasibility by presenting a construction based on the decision linear assumption (DLIN) [9]. The size of a signature is $\mathcal{O}(k)$ where k is the number of group elements in a message. While it is remarkable that the security can be shown based on a simple standard assumption, the scheme is not practical due to a large constant factor. Based on the q-Hidden LRSW assumption on the asymmetric bilinear groups, Green and Hohenberger presented an efficient scheme that provides security against random message attacks [30]. Unfortunately, extension to the chosen message security is not known. In [25], Fuchsbauer presented a practical scheme based on (a variant of) the Double Hidden Strong Diffie-Hellman Assumption (DHSDH) from [26]. While their scheme is pretty efficient, it has limited generality as a message must be a single Diffie-Hellman pair. Recently, in [22], Cathalo, Libert and Yung showed a practical scheme based on a combination of the Hidden Strong Diffie-Hellman Assumption (HSDH), Flexible Diffie-Hellman Assumption, and the DLIN assumption. Their signature consists of $9k + 4$ group elements and it is left as an open problem to construct constant-size signatures.

[1] Disjunction can be handled in somewhat tricky way with extra computation and storage[15]. When the witness is a scalar, it is possible to preserve the proof of knowledge property, but it requires bit-wise treatment and results in proofs growing linearly in the security parameter.

Our Contribution. We present the first *constant-size* GS-compatible signature scheme for messages of general bilinear group elements. A signature consists only of 7 group elements regardless of the size of the message. For a message $(m_1, \ldots, m_k)$, a signature (z, r, s, t, u, v, w) fulfills the verification equations

$$A = e(g_z, z)\, e(g_r, r)\, e(s, t) \prod_{i=1}^{k} e(g_i, m_i), \text{ and}$$

$$B = e(h_z, z)\, e(h_u, u)\, e(v, w) \prod_{i=1}^{k} e(h_i, m_i)$$

determined by the verification key.

The unforgeability against adaptive chosen message attacks is proven in the standard model based on a novel non-interactive assumption called the Simultaneous Flexible Pairing Assumption (SFP). It is a strong, i.e., so-called "q-type" assumption like the popular Strong Diffie-Hellman Assumption (SDH) [8]. On the positive side, SFP is a rare strong assumption that achieves the optimal quadratic security bound when analyzed in the generic group model [49] while SDH and its variations suffer from a cubic bound. (We refer to [23] and [41] for a risk and discussion about non-optimality in the generic model.) Another positive point is that SFP implies the Simultaneous Double Pairing Assumption (SDP), a simple assumption implied by DLIN and that allows to build useful commitment schemes that could be smoothly integrated into constructions based on SFP. On the negative side, SFP is more complex than (H)SDH. Nevertheless, we enlighten the bright side and hope that SFP be considered as a reasonable alternative for primitive designs when only group elements are involved.

We then explore variations and a few applications. In Section 5, based on the observation that the constant-size signatures allow unbounded "signature chaining", we present a signature scheme that signs *unbounded-size* messages. Since the message space of the resulting scheme covers the verification key space, this extension gives an automorphic signature scheme [25], which has number of interesting high-level applications coupled with GS proofs. In Section 6, we addresses *simulatability* in the common reference string (CRS) model. With simulatable signatures, a simulator can create signatures for arbitrary messages by using the trapdoor for the CRS. Such a property is useful in building adaptively secure protocols where a simulator has to have correct signatures without having help from a corrupted signer [2]. The resulting scheme gives an efficient instantiation to (adaptively secure variant of) Fischlin's round-optimal blind signature framework [24, 2] which we present in Section 9.1. It has been an open problem since Crypto'06 and considered as difficult [40]. In Section 7, we present a scheme that works in the asymmetric setting where the symmetric external Diffie-Hellman (SXDH) assumption holds. This setting is of interest as the GS proof system can provide better efficiency and some protocols may demand such a setting. We stress that it is not trivial to sign a message consisting of elements from both $\mathbb{G}_1$ and $\mathbb{G}_2$ since there are no efficient mappings between both groups, and straightforward independent signing allows a forgery. Finally, in Section 8, we show a variation that provides strong unforgeability with constant-size signatures.

As a bi-product, we present several homomorphic trapdoor commitment schemes that are useful in coupling with our signature schemes and the GS proofs. One of them is fully GS-compatible, i.e., its commitment-key, message, commitment, and decommitment are in $\mathbb{G}_1$ and $\mathbb{G}_2$ and the verification predicate consists of pairing product equations. It is the first such scheme that binds multiple messages at once.

Organization. After introducing necessary notations and notions in Section 2, the main constant-size signature scheme is presented in Section 4. It is followed by variations in Section 5, Section 6,

and Section 7. We show outline of high-level applications in Section 9. Some of the formal proofs are shown in Appendix. The commitment schemes are presented in Appendix B.

2 Preliminaries

2.1 Common Setup with Bilinear Groups

Let $\Lambda := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \tilde{g})$ be a description of groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of prime order p equipped with efficient bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. It also includes a random generator g of $\mathbb{G}_1$ and $\tilde{g}$ of $\mathbb{G}_2$. By $\mathbb{G}_1^*$ we denote $\mathbb{G}_1 \setminus \{1_{\mathbb{G}_1}\}$, and the same for $\mathbb{G}_2^*$ and $\mathbb{G}_T^*$. By Λ_{sym} we denote a special case of Λ where $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$. Similarly, Λ_{xdh} denotes a case where the Decision Diffie-Hellman (DDH) assumption holds for $\mathbb{G}_1$ ($\mathrm{DDH}_{\mathbb{G}_1}$ in short). This setting implies that there is no efficiently computable mapping $\mathbb{G}_1 \to \mathbb{G}_2$. And Λ_{sxdh} denotes a case where the DDH assumption holds for both $\mathbb{G}_1$ and $\mathbb{G}_2$. This means that no efficient mapping is available for either direction. The Λ_{xdh} and Λ_{sxdh} settings are usually referred to as the (Symmetric) External Diffie-Hellman Assumption [48, 9, 28, 50]. For differences of these settings in practice, we refer to [27]. A scheme (or an assumption or a proof) designed and proven in one setting may not necessarily go through in a different setting. In particular, if the scheme is for Λ_{sym} and uses the homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$, it does not work, or not known to be secure when used with Λ_{xdh} or Λ_{sxdh}. We treat Λ as a common parameter implicitly given to all algorithms of interest. However, we present our constructions with care so that it is clear in which setting they work and are secure.

2.2 Digital Signatures

Definition 1 (Digital Signature Scheme). A digital signature scheme SIG is a set of algorithms ($\mathsf{SIG.Key}$, $\mathsf{SIG.Sign}$, $\mathsf{SIG.Vrf}$) such that:

$\mathsf{SIG.Key}(1^\lambda)$: A key generation algorithm that takes security parameter 1^λ and generates a verification key vk and a signing key sk. Message space $\mathcal{M}$ is associated to vk.

$\mathsf{SIG.Sign}(sk, m)$: A signature generation algorithm that computes a signature σ for input message m by using signing key sk.

$\mathsf{SIG.Vrf}(vk, m, \sigma)$: A verification algorithm that outputs 1 for acceptance or 0 for rejection according to the input.

A signature scheme must provide correctness in the sense that if the key pair and a signature on a message are generated legitimately, $\mathsf{SIG.Vrf}$ returns 1. In this paper, algorihtms works over common bilinear setting Λ. The security parameter 1^λ allows $\mathsf{SIG.Key}$ to implicitly select Λ of appropreate size. $\mathsf{SIG.Key}$ may also take some other parameters if necessary.

We use standard notion of existential unforgeability against adaptive chosen message attacks [29] (EUF-CMA in short) formally defined as follows.

Definition 2 (Existential Unforgeability against Adaptive Chosen Message Attacks). A signature scheme is existentially unforgeable against adaptive chosen message attacks if, for any polynomial-time adversary $\mathcal{A}$, the following experiment returns 1 with negligible probability.

Experiment :

$(vk, sk) \leftarrow \mathsf{SIG.Key}(1^\lambda)$

$(m^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{sign}}}(vk)$

Return 1 if $m^\star \notin Q_m$ and $1 \leftarrow \mathsf{SIG.Vrf}(vk^\star, m^\star, \sigma^\star)$. Return 0, otherwise.

$\mathcal{O}_{\mathsf{sign}}$ is the signing oracle that takes message m and returns $\sigma \leftarrow \mathsf{SIG.Sign}(sk, m)$. Q_m is the messages submitted to $\mathcal{O}_{\mathsf{sign}}$. By requireing $(m^\star, \sigma^\star) \notin Q_{m,\sigma}$ where $Q_{m,\sigma}$ is pairs of a message and a signature observed by $\mathcal{O}_{\mathsf{sign}}$, we have the notion of Strong EUF-CMA (denoted by sEUF-CMA for short).

2.3 Assumptions

We start with introducing a simple assumption, called Double Pairing Assumption (DBP), that holds in asymmetric bilinear setting, i.e., $\Lambda \in \{\Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$.

Assumption 1 (Double Pairing Assumption (DBP)). Given $\Lambda \in \{\Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$ and $(g_z, g_r) \leftarrow \mathbb{G}_1^{*2}$, it is hard to find $(z, r) \in \mathbb{G}_2^* \times \mathbb{G}_2^*$ such that

$$1 = e(g_z, z)\, e(g_r, r). \tag{1}$$

It is obvious that DBP does not hold for $\Lambda = \Lambda_{\mathsf{sym}}$ since $(z, r) = (g_r^{-1}, g_z) \neq (1, 1)$ fulfills the relation. On the other hand, we can show that DBP holds for $\Lambda \in \{\Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$ where DDH is assumed hard in $\mathbb{G}_1$.

Theorem 1. *If $DDH_{\mathbb{G}_1}$ holds for Λ, then DBP holds for Λ.*

The proof is by a straightforward reduction and given in Appendix A.1.

We note that the DBP assumption could be viewed as a simpler version of the Simultaneous Triple Pairing Assumption (STP) [33]. The DBP assumption was introduced in an earlier version of this work, and independently in [34] (personal communication) by Groth, who also showed explicitly that DBP implies STP.

Next is an extension of DBP, called Simultaneous Double Pairing Assumption (SDP), which is a weaker assumption and can be justified in any setting $\Lambda \in \{\Lambda_{\mathsf{sym}}, \Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$ by a standard argument in the generic bilinear group model.

Assumption 2 (Simultaneous Double Pairing Assumption (SDP)). Given Λ and $(g_z, h_z, g_r, h_u) \leftarrow \mathbb{G}_1^{*4}$, it is hard to find $(z, r, u) \in \mathbb{G}_2^{*3}$ such that

$$1 = e(g_z, z)\, e(g_r, r) \qquad \text{and} \qquad 1 = e(h_z, z)\, e(h_u, u). \tag{2}$$

As shown in [22], SDP is implied by DLIN.

Next we introduce a novel assumption by extending SDP so that it should be hard to find another answer given several answers. Observe that, given an answer to an instance of SDP, one can easily yield more answers by exploiting the linearity of the relation to be satisfied. We eliminate such a linearity by multiplying random pairings to both sides of the equations in (2). Intuition is that, it should be hard to merge two random pairings $e(s, t)\, e(s', t')$ into one equivalent pairing $e(s'', t'')$. We call such a random part *flexible* as random pairings can be easily randomized or combined when their relation with respect to the same bases is known.

Assumption 3 (Simultaneous Flexible Pairing Assumption (SFP)). Let Λ be a common parameter and let g_z, h_z, g_r, and h_u be random generators of $\mathbb{G}_1$. Let $(a, \tilde{a}), (b, \tilde{b})$ be random pairs in $\mathbb{G}_1 \times \mathbb{G}_2$. For $j = 1, \ldots, q$, let $R_j = (z, r, s, t, u, v, w)$ that satisfies

$$e(a, \tilde{a}) = e(g_z, z)\, e(g_r, r)\, e(s, t) \quad \text{and} \quad e(b, \tilde{b}) = e(h_z, z)\, e(h_u, u)\, e(v, w). \tag{3}$$

Given $(\Lambda, g_z, h_z, g_r, h_u, a, \tilde{a}, b, \tilde{b})$ and uniformly chosen $R_1, \ldots, R_q$, it is hard to find $(z^\star, r^\star, s^\star, t^\star, u^\star, v^\star, w^\star)$ that fulfill relations in (3) under the restriction that $z^\star \neq 1$ and $z^\star \neq z \in R_j$ for every R_j.

Theorem 2. *For any generic algorithm $\mathcal{A}$, the probability that $\mathcal{A}$ breaks SFP with ℓ group operations and pairings is bound by $\mathcal{O}(q^2 + \ell^2)/p$.*

In Appendix A.3, a proof of Theorem 2 is given for the case of $\Lambda = \Lambda_{\mathsf{sym}}$. The argument can be translated to the asymmetric settings. The following holds with respect to SFP and SDP.

Theorem 3. *SFP $\Rightarrow$ SDP.*

A formal proof is in Appendix A.2. An intuition is that, given an answer (z, r, u) to SDP, setting $(s, t, v, w) = (a, \tilde{a}, b, \tilde{b})$ results in a correct answer, (z, r, u, s, t, v, w), to SFP.

2.4 Other Basics

A brief introduction of the Groth-Sahai proof system is in Appendix E. For further details, we refer to the original paper [35].

3 Randomization Techniques

We introduce techniques that randomize elements in a pairing or a pairing product without changing their value in $\mathbb{G}_T$. These useful techniques are used throughout the paper.

- **Inner Randomization** $(x', y') \leftarrow \mathsf{Rand}(x, y)$**:** A pairing $A = e(x, y) \neq 1$ is randomized as follows. Choose $\gamma \leftarrow \mathbb{Z}_p^*$ and let $(x', y') = (x^\gamma, y^{1/\gamma})$. It then holds that (x', y') distributes uniformly over $\mathbb{G}_1 \times \mathbb{G}_2$ under the condition of $A = e(x', y')$. If $A = 1$, then first flip a coin and pick $e(1, 1)$ with probability $1/(2p - 1)$. If it is not selected, flip a coin and pick either $e(1, x)$ or $e(x, 1)$ with probability $1/2$. Then select x uniformly from the corresponding group except for 1.

- **Sequential Randomization** $\{x'_i, y'_i\}_{i=1}^k \leftarrow \mathsf{RandSeq}(\{x_i, y_i\}_{i=1}^k)$**:** A pairing product $A = e(x_1, y_1)\, e(x_2, y_2) \ldots e(x_k, y_k)$ is randomized into $A = e(x'_1, y'_1)\, e(x'_2, y'_2) \ldots e(x'_k, y'_k)$ as follows: Let $(\gamma_1, \ldots, \gamma_{k-1}) \leftarrow \mathbb{Z}_p^{k-1}$. We begin with randomizing the first pairing by using the second pairing as follows. First verify that $y_1 \neq 1$ and $x_2 \neq 1$. If $y_1 = 1$, replace the first pairing $e(x_1, 1)$ with $e(1, y_1)$ with a new random $y_1 (\neq 1)$. The case of $x_2 = 1$ is handled in the same manner. Then multiply $1 = e(x_2^{-\gamma_1}, y_1)\, e(x_2, y_1^{\gamma_1})$ to both sides of the formula. We thus obtain

$$A = e(x_1 x_2^{-\gamma_1}, y_1)\, e(x_2, y_1^{\gamma_1} y_2)\, e(x_3, y_3) \ldots e(x_k, y_k). \tag{4}$$

  Next we randomize the second pairing by using the third one. As before, if $y_1^{\gamma_1} y_2 = 1$ or $x_3 = 1$, replace them to random values. Then multiply $1 = e(x_3^{-\gamma_2}, y_1^{\gamma_1} y_2)\, e(x_3, (y_1^{\gamma_1} y_2)^{\gamma_2})$. We thus have

$$A = e(x_1 x_2^{-\gamma_1}, y_1)\, e(x_2 x_3^{-\gamma_2}, y_1^{\gamma_1} y_2)\, e(x_3, (y_1^{\gamma_1} y_2)^{\gamma_2} y_3) \ldots e(x_k, y_k). \tag{5}$$

This continues up to the $(k-1)$-st pairing. When done, the value of the i-th pairing distributes uniformly in $\mathbb{G}_T$ due to the uniform choice of γ_i. The k-th pairing follows the distribution determined by A and preceding $k-1$ pairings. To complete the randomization, every pairing is processed by the inner randomization.

The sequential randomization can be used to extend a product of k pairings a product of arbitrary $\geq k$ pairings by appending $e(1,1)$ before randomization. By $\{x'_i, y'_i\}_{i=1}^{k'} \leftarrow \mathsf{Extend}(\{x_i, y_i\}_{i=1}^{k})$ for $k'(> k)$ we denote the sequential randomization with extension. Parameters k and k' should be clear from the input and the output.

- **One-side Randomization** $\{x'_i\}_{i=1}^{k} \leftarrow \mathsf{RandOneSide}(\{g_i, x_i\}_{i=1}^{k})$**:** Let g_i be an element in $\mathbb{G}_1^*$ of symmetric setting Λ_{sym}. A pairing product $A = e(g_1, x_1)\, e(g_2, x_2) \ldots e(g_k, x_k)$ is randomized into $A = e(g_1, x'_1)\, e(g_2, x'_2) \ldots e(g_k, x'_k)$ as follows. Let $(\gamma_1, \ldots, \gamma_{k-1}) \leftarrow \mathbb{Z}_p^{k-1}$. First multiply $1 = e(g_1, g_2^{\gamma_1})\, e(g_2, g_1^{-\gamma_1})$ to both sides of the formula. We thus obtain

$$A = e(g_1, x_1 g_2^{\gamma_1})\, e(g_2, x_2 g_1^{-\gamma_1})\, e(g_3, x_3) \ldots e(g_k, x_k). \tag{6}$$

Next multiply $1 = e(g_2, g_3^{\gamma_2})\, e(g_3, g_2^{-\gamma_2})$. We thus have

$$A = e(g_1, x_1 g_2^{\gamma_1})\, e(g_2, x_2 g_1^{-\gamma_1} g_3^{\gamma_2})\, e(g_3, x_3 g_2^{-\gamma_2}) \ldots e(g_k, x_k). \tag{7}$$

This continues until γ_{k-1} and we eventually have $A = e(g_1, x'_1) \ldots e(g_k, x'_k)$. Observe that every x'_i for $i = 1, \ldots, k-1$ distributes uniformly in $\mathbb{G}$ due to the uniform multiplicative factor $g_{i+1}^{\gamma_i}$. In the k-th pairing, x'_k follows the distribution determined by A and the preceding $k-1$ pairings. Thus $(x'_1, \ldots, x'_k)$ is uniform over $\mathbb{G}^k$ under constraint of being evaluated to A.

Note that the algorithms yield uniform elements and thus may include pairings that evaluate to $1_{\mathbb{G}_T}$. If it is not preferable, it can be avoided by repeating that particular step once again excluding the bad randomness.

4 The Main Scheme: Constant-Size Signatures

4.1 Technical Overview

Combining a trapdoor commitment scheme and a strong assumption is a well-known approach for designing signature schemes. To bring this idea into a real construction, we need a trapdoor commitment scheme and a useful (and acceptable) assumption which are compatible with each other, something that is not easily obtained under strong design constraints. In our case, we can build efficient multi-message trapdoor commitment schemes from SDP as shown in Appendix B. Furthermore, SDP is implied by SFP as shown in Theorem 3, so that should allow a smooth combination.

A remaining technical issue is how to deal with "exceptions" such as $z^\star \neq 1$ in SFP. The signature scheme should not inherit it since when proving a knowledge of a signature, the condition $z \neq 1$ is not trivial to prove and affects the efficiency. We address this issue by involving another set of elements $(a_0, \tilde{a}_0)$ and $(b_0, \tilde{b}_0)$ in the verification predicate. In the proof of unforgeability, these elements hold a secret random offset $\tilde{g}^\zeta$ that will be multiplied to z in a forged signature so that the answer to SFP, $z^\star = z\tilde{g}^\zeta$, happens to be 1 only by chance. (The real proof is slightly more involved.)

The randomization techniques from Section 3 also help the construction and the security proof in such a way that elements in pairings are uniform in the real signatures and the ones created in the security reduction.

4.2 Construction

Let $\vec{m} = (m_1, \ldots, m_k) \in \mathbb{G}_2^k$ be a message to be signed. Parameter k determines the length of a message and shorter messages are implicitly padded with $1_{\mathbb{G}_2}$-s. Let $\Lambda \in \{\Lambda_{\mathsf{sym}}, \Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$. We remind that $\Lambda := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \tilde{g})$ is an implicit input to the algorithms described below.

- Key Generation. $\mathsf{SIG.Key}(1^\lambda)$: Choose random generators $g_r, h_u \leftarrow \mathbb{G}_1^*$. For $i = 1, \ldots, k$, choose $\gamma_i, \delta_i \leftarrow \mathbb{Z}_p^{*2}$ and compute $g_i = g_r^{\gamma_i}$ and $h_i = h_u^{\delta_i}$. Choose $\gamma_z, \delta_z \leftarrow \mathbb{Z}_p^{*2}$ and compute $g_z = g_r^{\gamma_z}$ and $h_z = h_u^{\delta_z}$. Also choose $\alpha, \beta \leftarrow \mathbb{Z}_p^{*2}$ and compute $\{a_i, \tilde{a}_i\}_{i=0}^1 \leftarrow \mathsf{Extend}(g_r, \tilde{g}^\alpha)$ and $\{b_i, \tilde{b}_i\}_{i=0}^1 \leftarrow \mathsf{Extend}(h_u, \tilde{g}^\beta)$. Set $vk = (g_z, h_z, g_r, h_u, \{g_i, h_i\}_{i=1}^k, \{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=0}^1)$ and $sk = (vk, \alpha, \beta, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^k)$. Output (vk, sk).

- Signature Issuing. $\mathsf{SIG.Sign}(sk, \vec{m})$: Choose $\zeta, \rho, \tau, \varphi, \omega$ randomly from $\mathbb{Z}_p^*$ and set:

$$z = \tilde{g}^\zeta, \quad r = \tilde{g}^{\rho - \gamma_z \zeta} \textstyle\prod_{i=1}^k m_i^{-\gamma_i}, \quad s = g_r^\tau, \quad t = \tilde{g}^{(\alpha-\rho)/\tau},$$
$$u = \tilde{g}^{\varphi - \delta_z \zeta} \textstyle\prod_{i=1}^k m_i^{-\delta_i}, \quad v = h_u^\omega, \quad w = \tilde{g}^{(\beta-\varphi)/\omega}.$$

  Output $\sigma = (z, r, s, t, u, v, w)$ as a signature.

- Verification. $\mathsf{SIG.Vrf}(vk, \vec{m}, \sigma)$: Parse σ into (z, r, s, t, u, v, w). Output 1 if

$$A = e(g_z, z)\ e(g_r, r)\ e(s, t)\ \prod_{i=1}^k e(g_i, m_i), \text{ and} \tag{8}$$

$$B = e(h_z, z)\ e(h_u, u)\ e(v, w)\ \prod_{i=1}^k e(h_i, m_i) \tag{9}$$

  hold for $A = e(a_0, \tilde{a}_0)\ e(a_1, \tilde{a}_1)$ and $B = e(b_0, \tilde{b}_0)\ e(b_1, \tilde{b}_1)$. Output 0, otherwise.

4.3 Security

Theorem 4. *SIG in Section 4.2 is correct. It is EUF-CMA if SFP holds for Λ.*

Proof. CORRECTNESS. Observe that

$$e(g_z, z)\ e(g_r, r)\ e(s, t)\ \prod_{i=1}^k e(g_i, m_i) = e\left(g_r^{\gamma_z}, \tilde{g}^\zeta\right)\ e\left(g_r, \tilde{g}^{\rho - \gamma_z \zeta} \prod_{i=1}^k m_i^{-\gamma_i}\right)\ e\left(s, t\right)\ \prod_{i=1}^k e\left(g_r^{\gamma_i}, m_i\right)$$
$$= e\left(g_r, \tilde{g}^\rho\right)\ e\left(g_r^\tau, \tilde{g}^{(\alpha-\rho)/\tau}\right) \quad = \quad e\left(g_r, \tilde{g}^\alpha\right) \quad = \quad A$$

holds. Thus (8) is fulfilled. Relation (9) is verified in the same manner.

UNFORGEABILITY. Let $\mathcal{A}$ be an adversary that has a non-negligible advantage of forging a signature for the above scheme on a message $\vec{m}^\dagger$, $\vec{m}^\dagger \notin \{\vec{m}_j\}_{j=1}^q$, after adaptively querying the signing oracle on messages $\vec{m}_j$, for $j = 1, \ldots, q$, and receiving signatures σ_j. We construct a reduction algorithm which takes an input Λ, g_z, h_z, g_r, h_u, $(a, \tilde{a})$, $(b, \tilde{b})$, and uniformly chosen tuples R_j for $j = 1, \ldots, q$ as defined in Assumption 3, and simulates the view of $\mathcal{A}$ in the attack environment as follows:

- (Simulating SIG.Key) : Use (g_z, h_z, g_r, h_u) as given in the input. For $i = 1, \ldots, k$ set $g_i = g_z^{\chi_i} g_r^{\gamma_i}$ and $h_i = h_z^{\chi_i} h_u^{\delta_i}$, where χ_i, γ_i, $\delta_i \leftarrow \mathbb{Z}_p^*$. Choose ζ, ρ, φ randomly from $\mathbb{Z}_p^*$. Then compute $((a_0, \tilde{a}_0), (a_1, \tilde{a}_1)) \leftarrow \mathsf{RandSeq}((g_z^{\zeta} g_r^{\rho}, \tilde{g}), (a, \tilde{a}))$ and $((b_0, \tilde{b}_0), (b_1, \tilde{b}_1)) \leftarrow \mathsf{RandSeq}((h_z^{\zeta} h_u^{\varphi}, \tilde{g}), (b, \tilde{b}))$. The verification key is $vk = (g_z, h_z, g_r, h_u, \{g_i, h_i\}_{i=1}^k, \{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=0}^1)$.
- (Simulating SIG.Sign) : Given message $\vec{m}$, take a fresh tuple $R_j = (z_j, r_j, s_j, t_j, u_j, v_j, w_j)$ from the input instance. Then compute

$$z = z_j \, \tilde{g}^{\zeta} \prod_{i=1}^{k} m_i^{-\chi_i}, \quad r = r_j \, \tilde{g}^{\rho} \prod_{i=1}^{k} m_i^{-\gamma_i}, \quad s = s_j, \quad t = t_j,$$

$$u = u_j \, \tilde{g}^{\varphi} \prod_{i=1}^{k} m_i^{-\delta_i}, \quad v = v_j, \quad w = w_j.$$

  The signature is $\sigma = (z, r, s, t, u, v, w)$. It is easy to verify that the signature satisfies the verification equations.

When $\mathcal{A}$ outputs $(\vec{m}^{\dagger}, (z^{\dagger}, r^{\dagger}, s^{\dagger}, t^{\dagger}, u^{\dagger}, v^{\dagger}, w^{\dagger}))$, compute

$$z^{\star} = z^{\dagger} \, \tilde{g}^{-\zeta} \prod_{i=1}^{k} \left(m_i^{\dagger}\right)^{\chi_i}, \quad r^{\star} = r^{\dagger} \, \tilde{g}^{-\rho} \prod_{i=1}^{k} \left(m_i^{\dagger}\right)^{\gamma_i}, \quad u^{\star} = u^{\dagger} \, \tilde{g}^{-\varphi} \prod_{i=1}^{k} \left(m_i^{\dagger}\right)^{\delta_i},$$

and set $s^{\star} = s^{\dagger}$, $t^{\star} = t^{\dagger}$, $v^{\star} = v^{\dagger}$, and $w^{\star} = w^{\dagger}$. The output is $(z^{\star}, r^{\star}, s^{\star}, t^{\star}, u^{\star}, v^{\star}, w^{\star})$. This completes the description of the reduction algorithm.

The above signatures follow correct distribution. So $\mathcal{A}$ outputs a successful forgery with a non-negligible probability. Then, for the output of the reduction algorithm, it holds that

$$\begin{aligned} e(g_z, z^{\star}) \, e(g_r, r^{\star}) \, e(s^{\star}, t^{\star}) &= e\left(g_z, z^{\dagger} \, \tilde{g}^{-\zeta} \prod_{i=1}^{k} \left(m_i^{\dagger}\right)^{\chi_i}\right) \, e\left(g_r, r^{\dagger} \, \tilde{g}^{-\rho} \prod_{i=1}^{k} \left(m_i^{\dagger}\right)^{\gamma_i}\right) \, e\left(s^{\dagger}, t^{\dagger}\right) \\ &= e\left(g_z^{-\zeta} g_r^{-\rho}, \tilde{g}\right) \, e\left(g_z, z^{\dagger}\right) \, e\left(g_r, r^{\dagger}\right) \, e\left(s^{\dagger}, t^{\dagger}\right) \prod_{i=1}^{k} e\left(g_i, m_i^{\dagger}\right) \\ &= e\left(g_z^{\zeta} g_r^{\rho}, \tilde{g}\right)^{-1} \prod_{i=0}^{1} e(a_i, \tilde{a}_i) = e(a, \tilde{a}). \end{aligned}$$

One can also verify that $e(g_z, z^{\star}) \, e(h_u, u^{\star}) \, e(v^{\star}, w^{\star}) = e(b, \tilde{b})$ holds in the same way.

What remains is to show that $z^{\star}$ is not in $\{1, z_1, \ldots, z_q\}$. It can be verified that parameters ζ and χ_i for $i = 1, \ldots, k$ are independent from the view of adversary $\mathcal{A}$. Namely, for any view of the adversary and for any choice of ζ and χ_i for $i = 1, \ldots, k$, there exist unique and consistent parameters ρ, φ, γ_i, δ_i, for $i = 1, \ldots, k$ and z_j, r_j, u_j for $j = 1, \ldots, q$. First we show that the probability $z^{\star} \in \{z_1, \ldots, z_q\}$ is negligible. For every z_j and signature $\sigma = (z, r, s, t, u, v, w)$ on a message $\vec{m}$ simulated by using z_j, it holds that

$$\frac{z^{\star}}{z_j} \quad = \quad \frac{z^{\dagger} \, \tilde{g}^{-\zeta} \prod_{i=1}^{k} \left(m_i^{\dagger}\right)^{\chi_i}}{z \, \tilde{g}^{-\zeta} \prod_{i=1}^{k} m_i^{\chi_i}} \quad = \quad \frac{z^{\dagger}}{z} \prod_{i=1}^{k} \left(\frac{m_i^{\dagger}}{m_i}\right)^{\chi_i}.$$

Since $\vec{m}^{\dagger} \neq \vec{m}$, there exists i such that $m_i^{\dagger} \neq m_i$. Since χ_i is information theoretically hidden from the view of the adversary, the probability that $z^{\star} = z_j$ is negligible due to the term $(m_i^{\dagger}/m_i)^{\chi_i}$ in

the above equation. To show that $z^\star = \left(z^\dagger\right) \tilde{g}^{-\zeta} \prod_{i=1}^{k} \left(m_i^\dagger\right)^{\chi_i} = 1$ with a negligible probability, notice that ζ is also independent from the view of the adversary and the claim holds due to the uniform choice of ζ. Therefore, the probability that $z^\star \notin \{1, z_1, \ldots, z_q\}$ is overwhelming. ∎

4.4 Notable Properties

This section introduces two useful properties of SIG, which we call *Partial Perfect Randomizability* and *Signature Binding Property*.

Partial Perfect Randomizability. Given a signature (z, r, s, t, u, v, w) one can randomize every element except for z by applying the sequential randomization technique with small tweak as follows. Define the function SigRand , $(r', s', t', u', v', w') \leftarrow \mathsf{SigRand}(r, s, t, u, v, w)$, as:

- Randomize (r, s, t) into (r', s', t') as follows.
  - First, if $t = 1$, set $s = 1$ and choose $t \leftarrow \mathbb{G}_2^*$.
  - Then, choose $\varrho \leftarrow \mathbb{Z}_p$ and compute

$$r' = r\, t^{\varrho}, \quad (s', t') \leftarrow \mathsf{Rand}(s g_r^{-\varrho}, t) \tag{10}$$

- Randomize (u, s, t) into (u', s', t') analogously.

Lemma 1. *The above (r', s', t', u', v', w') distributes uniformly over $(\mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_2)^2$ under constraint that $e(g_r, r)\, e(s, t) = e(g_r, r')\, e(s', t')$ and $e(h_u, u)\, e(v, w) = e(h_u, u')\, e(v', w')$.*

Proof. Uniformity of $r' \in \mathbb{G}_2$ follows from $t \neq 1$ and the uniformity of ϱ in (10). Under the described constraints, for any choice of r', their is a unique value $e(s', t') = e(g_r, r)\, e(s, t)\, e(g_r, r')^{-1}$. Then, uniformity of s' and t' holds from the property of Rand. The same is true for (u', v', w'). ∎

The claim implies that (s', t', v', w') is information theoretically independent of the remaining elements (z, r', u') in a signature, the message, and the verification key. (In general, the same is true for publishing any two elements from (r', s', t') and (u', v', w') respectively.) This property is useful in reducing the task of combined proofs. See Section 9.1 for typical use of this property.

Signature Binding Property. Roughly, it claims that no one can obtain two signatures which have the same s and v. In the following formal statement, the adversary is allowed to submit both $\vec{m}$ and $\vec{m}'$ to the signing oracle. Hence the property is not implied by EUF-CMA in general.

Lemma 2. *Under adaptive chosen message attacks, no adversary can output $(\vec{m}, \sigma)$ and $(\vec{m}', \sigma')$ such that $1 = \mathsf{SIG.Vrf}(vk, \vec{m}, \sigma) = \mathsf{SIG.Vrf}(vk, \vec{m}', \sigma')$, $\vec{m} \neq \vec{m}'$, and (s, v) are shared in σ and σ'.*

Recall that s and v are uniformly chosen in the signature generation algorithm. Hence they are independent of the remaining part of the signature and the message. Accordingly, in a way, publishing (s, v) together with the verification key works as a commitment of the signature and the message. This property is used in Section 5, and would find more applications.

Proof. Suppose that there is a successful adversary, $\mathcal{A}$ that outputs the signatures as in the lemma. We then construct an adversary $\mathcal{B}$ that breaks EUF-CMA of SIG.

Given vk and oracle access to $\mathcal{O}_{\mathsf{sign}}$, $\mathcal{B}$ invokes $\mathcal{A}$ with vk. Every signing query from $\mathcal{A}$ is directly passed to $\mathcal{O}_{\mathsf{sign}}$ and the signatures are returned directly to $\mathcal{A}$. Hence $\mathcal{B}$'s simulation is

perfect. Eventually, $\mathcal{A}$ terminates and outputs $\sigma = (z, r, s, t, u, v, w)$, $\vec{m} = (m_1, \ldots, m_k)$, $\sigma' = (z', r', s, t', u', v, w')$, and $\vec{m}' = (m'_1, \ldots, m'_k)$.

$\mathcal{B}$ then chooses $\varrho_1, \varrho_2 \leftarrow \mathbb{Z}_p^*$ and computes linear combinations

$$\begin{array}{lll} z^\star = (z'^{\varrho_1} z^{\varrho_2})^{1/(\varrho_1+\varrho_2)}, & r^\star = (r'^{\varrho_1} r^{\varrho_2})^{1/(\varrho_1+\varrho_2)}, & u^\star = (u'^{\varrho_1} u^{\varrho_2})^{1/(\varrho_1+\varrho_2)}, \\ t^\star = (t'^{\varrho_1} t^{\varrho_2})^{1/(\varrho_1+\varrho_2)}, & w^\star = (w'^{\varrho_1} w^{\varrho_2})^{1/(\varrho_1+\varrho_2)}, & m_i^\star = (m_i'^{\varrho_1} m_i^{\varrho_2})^{1/(\varrho_1+\varrho_2)}. \end{array}$$

Then outputs $\sigma^\star = (z^\star, r^\star, s, t^\star, u^\star, v, w^\star)$ and $\vec{m}^\star = (m_1^\star, \ldots, m_k^\star)$. This completes the specification of $\mathcal{B}$.

We verify the correctness of $\mathcal{B}$ as follows. Since these signatures are valid, they satisfy

$$A = e(g_z, z')\, e(g_r, r')\, e(s, t') \prod_{i=1}^{k} e(g_i, m'_i) = e(g_z, z)\, e(g_r, r)\, e(s, t) \prod_{i=1}^{k} e(g_i, m_i), \text{ and} \tag{11}$$

$$B = e(h_z, z')\, e(h_u, u')\, e(v, w') \prod_{i=1}^{k} e(h_i, m'_i) = e(h_z, z)\, e(h_u, u)\, e(v, w) \prod_{i=1}^{k} e(h_i, m_i). \tag{12}$$

From (11) and (12), the output of $\mathcal{B}$ satisfies

$$A = e(g_z, z^\star)\, e(g_r, r^\star)\, e(s, t^\star) \prod_{i=1}^{k} e(g_i, m_i^\star), \text{ and}$$

$$B = e(h_z, z^\star)\, e(h_u, u^\star)\, e(v, w^\star) \prod_{i=1}^{k} e(h_i, m_i^\star).$$

Accordingly, $\sigma^\star = (z^\star, r^\star, s, t^\star, u^\star, v, w^\star)$ is a correct signature for message vector $\vec{m}^\star = (m_1^\star, \ldots, m_k^\star)$. Since $\vec{m}' \neq \vec{m}$ there exists $i^\star$ such that $m'_{i^\star} \neq 1$ or $m_{i^\star} \neq 1$ (if no such index exists, $\vec{m}' = \vec{m} = \vec{1}$). Due to the randomness of ϱ_1 or ϱ_2, message $m_{i^\star}^\star = (m'_{i^\star}{}^{\varrho_1} m_{i^\star}{}^{\varrho_2})^{1/(\varrho_1+\varrho_2)}$ distributes uniformly over $\mathbb{G}_2$. Accordingly, $\vec{m}^\star$ is different from any message vector observed by $\mathcal{O}_{\mathsf{sign}}$ with overwhelming probability. Thus, $(\sigma^\star, \vec{m}^\star)$ is a valid forgery to SIG. ∎

4.5 Variations

- We can replace $a_i, \tilde{a}_i, b_i, \tilde{b}_i$ with $A = e(g_r, \tilde{g}^\alpha)$ and $B = e(h_u, \tilde{g}^\beta)$ in a verification-key, and use the A and B directly in the verification equations (8) and (9). The reason we include a representation of A (and B) in $\mathbb{G}_1$ and $\mathbb{G}_2$ is to address the needs to put the verification key into the base groups. The GS-proof system provides zero-knowledge property for statements that do not include elements from $\mathbb{G}_T$ except for $1_{\mathbb{G}_T}$. When WI is of only concern, one can include A and B in vk and use them directly in the verification. We use this modification in Section 9.1. The same is possible for other schemes in this paper.

- Let $\langle n \rangle$ denote a deterministic encoding of non-negative integer n ($< p$) to an element of $\mathbb{G}_2^*$. By limiting the maximum message length to be $k-1$ and putting $\langle |\vec{m}| \rangle$ at the tail of input message $\vec{m}$, shorter messages can be treated. Since the encoding is deterministic and black-box that is independent of the representation of the elements in $\vec{m}$, it does not impact the compatibility.

- As we observed in the very last stage of the security proof, $(a_0, \tilde{a}_0)$ and $(b_0, \tilde{b}_0)$ in a verification key is needed to handle the case where $z^\dagger = 1$ and $\vec{m}^\dagger = (1, \ldots, 1)$ happen at the same time. If $\vec{m}$ is encoded with length as $\vec{m}^\dagger = (1, \ldots, 1, \langle n \rangle)$, that exception case is not possible. Thus $(a_0, \tilde{a}_0)$ and $(b_0, \tilde{b}_0)$ can be removed from the scheme.
- In the asymmetric settings, one can swap $\mathbb{G}_1$ and $\mathbb{G}_2$ in the description of SIG to get the 'dual' scheme of SIG whose message space is $\mathbb{G}_1^k$.
- Dropping the flexible part $e(s,t)$ and $e(v,w)$ from the construction results in a strongly unforgeable one-time signature scheme based on the SDP assumption. It also saves $e(a_0, \tilde{a}_0)$ and $e(b_0, \tilde{b}_0)$. See Appendix C for details.

5 Signing Unbounded-Size Messages

5.1 Overview

Taking the advantage of having a constant-size signature scheme from Section 4, we introduce a simple "signature chain" approach to construct a signature scheme whose message size is not a-priori limited. That is, first sign m_1 to obtain σ_1, and next sign $\sigma_1 || m_2$ to obtain σ_2, then sign $\sigma_2 || m_3$ and so on. (Note that this rough description lacks some important details. In particular, signing only on m_1 at the beginning results in an insecure scheme. See also Appendix D where we discuss an alternative approach with efficiency comparison.)

A technical highlight in the construction is that, with our constant-size signature scheme, it is not necessary to include an entire signature into each step of chaining but including only two elements (s, v) constitutes a secure chain. This is possible due to the signature binding property of SIG as shown in Section 4.4.

5.2 Construction

Let SIG be the constant-size signature scheme from Section 4, whose message space is $\mathbb{G}_2^k$ for $k \geq 3$. We construct an unbounded-message signature scheme, $\mathsf{USIG1}$, as follows. Let $\Lambda = \Lambda_{\mathsf{sym}}$ be implicitly given to the functions described below. Recall that $\langle n \rangle$ is an encoding of n to an element of $\mathbb{G}_2^*$.

- $\mathsf{USIG1.Key}(1^\lambda)$: Generate random $(s_{-1}, v_{-1}) \in \mathbb{G}_1^2$. Invoke $(vk', sk) \leftarrow \mathsf{SIG.Key}(1^\lambda)$. Output $vk = (vk', s_{-1}, v_{-1})$ and sk.
- $\mathsf{USIG1.Sign}(sk, \vec{m})$: Parse $\vec{m}$ into $(m_1, \ldots, m_n)$. Let $\ell = \lceil \frac{n+1}{k-2} \rceil$. Let $m_0 = \langle n \rangle$ and $m_i = 1$ for $i = n+1, \ldots, \ell(k-2)$. For $i = 0, \ldots, \ell - 1$, compute $\sigma_i = (z_i, r_i, s_i, t_i, u_i, v_i, w_i) \leftarrow \mathsf{SIG.Sign}(sk, \vec{m}_i)$ where $\vec{m}_i = (s_{i-1}, v_{i-1}, m_{i(k-2)}, \ldots, m_{(i+1)(k-2)-1})$. Output $\sigma = (\sigma_0, \ldots, \sigma_{\ell-1})$.
- $\mathsf{USIG1.Vrf}(vk, \vec{m}, \sigma)$: Parse σ into $(\sigma_0, \ldots, \sigma_{\ell-1})$ and $\vec{m}$ into $(m_1, \ldots, m_n)$. Let $m_0 = \langle n \rangle$ and $m_i = 1$ for $i = n+1, \ldots, \ell(k-2)$. For $i = 0, \ldots, \ell - 1$, compute $b_i = \mathsf{SIG.Vrf}(vk', \vec{m}_i, \sigma_i)$ where $\vec{m}_i$ is formed in the same way as in $\mathsf{SIG.Sign}$. Output 1 if $b_i = 1$ for all $i = 0, \ldots, \ell - 1$. Output 0, otherwise.

The resulting signature is in the size of $7 \cdot \lceil \frac{n+1}{k-2} \rceil$.

Remarks. Filling $1_{\mathbb{G}}$ to the sloppy slots of the message space is for notational consistency. It does not increase either computation or storage. Setting $\Lambda = \Lambda_{\mathsf{sym}}$ is needed as (r_i, u_i) is in $\mathbb{G}_1^2$ while the message space is $\mathbb{G}_2^k$. It can be modified for the case of $\Lambda = \Lambda_{\mathsf{sxdh}}$ using the signature scheme

described in Section 7 (but not for the case of $\Lambda = \Lambda_{\mathsf{xdh}}$). If $\vec{m}$ is given as an on-line stream and the length is not known in advance, one can use the trapdoor commitment scheme TC2 from Appendix B so that m_0 is set to a random commitment and later opened to n when n is fixed. The opening information is included as a part of a signature. Since the opening information is a group element and the commitment verification predicate is a pairing product equation, the resulting verification predicate for USIG1 remains as a conjunction of pairing product equations.

Theorem 5. *If SIG is EUF-CMA, so is USIG1.*

Proof. Suppose that there is a successful adversary, say $\mathcal{A}$, that launches chosen message attacks and outputs a valid forgery, $((m_1^\dagger, \ldots, m_n^\dagger), (\sigma_0^\dagger, \ldots, \sigma_{\ell-1}^\dagger))$. Let $\vec{m}_i^\dagger$ be the message vector associated to $\sigma_i^\dagger$. We then have two cases.

Type-I. There is $\vec{m}_i^\dagger$ that has never been signed by the signing oracle.

Type-II. Every $\vec{m}_i^\dagger$ has been signed by the signing oracle (in separate queries).

Type-I forgery trivially breaks the unforgeability of SIG. For Type-II forgery, we show a reduction to the unforgeability of SIG as follows. Given verification key vk' of SIG and access to the signing oracle of SIG, we construct a simulator that uses adversary $\mathcal{A}$ and simulates USIG1 as follows. Let $\mathcal{O}_{\mathsf{sign}}$ be the signing oracle of SIG with respect to vk'.

- (Simulating USIG1.Key): Generate a random message vector $\vec{m}_{-1}$ of size k and send it to $\mathcal{O}_{\mathsf{sign}}$. Receive signature $(z_{-1}, r_{-1}, s_{-1}, t_{-1}, u_{-1}, v_{-1}, w_{-1})$ and output $vk = (vk', s_{-1}, v_{-1})$.
- (Simulating USIG1.Sign): On input $\vec{m}$, follow the legitimate signing algorithm by asking $\mathcal{O}_{\mathsf{sign}}$ to compute SIG.Sign. Then output the resulting signature.

Observe that s_{-1} and v_{-1} generated in the simulated USIG1.Key are uniform and independent of $\vec{m}_{-1}$. Simulation for USIG1.Sign is clearly perfect as it follows the legitimate procedure.

Suppose that adversary $\mathcal{A}$ outputs a valid forgery for USIG1. Then there exists a signing query (to the signing oracle of USIG1) in which $\vec{m}_{\ell-1}^\dagger$ is observed. Let $((m_1, \ldots, m_{n'}), (\sigma_0, \ldots, \sigma_{\ell'-1}))$ be the message and the signature with respect to the query and let $\vec{m}_i$ denote a message vector associated to σ_i. Let $i^\star$ be the index where $\vec{m}_{\ell-1}^\dagger = \vec{m}_{i^\star}$ happens. If $\ell - 1 = 0$, then $i^\star = 0$ is not the case because the message in the valid forgery must be fresh. In the case of $\ell - 1 \neq 0$ and $i^\star = 0$, it happens that $\vec{m}_{\ell-2}^\dagger \neq \vec{m}_{-1}$ with overwhelming probability since $\vec{m}_{-1}$ is chosen randomly and information theoretically independent from the view of the adversary. The same is true for the case of $\ell - 1 = 0$ and $i^\star > 0$. In the case of $\ell - 1 \neq 0$ and $i^\star > 0$, since the messages are prefix-free, there exists $j^\star$ such that $\vec{m}_{\ell-1-j^\star}^\dagger \neq \vec{m}_{i^\star-j^\star}$ happens for the first time when $j^\star$ is increased from 0 to $\min(\ell - 1, i^\star) + 1$. In any of the cases ($j^\star$ is set to 1 for the case of $i^\star = 0$ or $\ell - 1 = 0$), signature $\sigma_{\ell-1-j^\star}^\dagger$ shares s and v with $\sigma_{i^\star-j^\star}$ as they are included in $\vec{m}_{i^\star-j^\star+1} (= \vec{m}_{\ell-1-j^\star+1}^\dagger)$. This contradicts to the signature binding property of SIG as claimed in Lemma 2. ∎

6 Simulatable Signatures

6.1 Overview

A *simulatable signature scheme* is a signature scheme in the CRS model that allows to create valid signatures without the signing-key but with a trapdoor associated to the common reference string.

The notion is introduced in [2] but in an informal way dedicated for their purposes. We elaborate the notion and present a formal treatment with reasonable construction in this section.

A simulatable signature is a useful tool in combination with a witness indistinguishable (WI) proof system. Unlike zero-knowledge (ZK) proofs, WI proof system does not accompany a simulator. So when a signature is a part of the witness and the signer is corrupt and useless, simulatable signature can provide a correct witness to the entity having the trapdoor. This situation happens in reality, for instance, when we attempt to instantiate Fischlin's round-optimal blind signature scheme [24] (modified to use WI as suggested in [36, 2]).

It is known that a simulatable signature scheme can be unconditionally constructed from any regular signature scheme by modifying the verification predicate in such a way that a signature is accepted if it passes regular verification with respect to the signer's verification key *or* the verification key included in the CRS. This generic construction, however, inherently involves disjunction in the resulting verification predicate.

Our construction shares the idea of two-keys. But we use a trapdoor commitment scheme and a signature scheme combined. We assign a commitment-key to the CRS and use a signing-key for real signature generation. Then a reference signature on a default message is included into a verification key. When simulation is needed, we use the trapdoor for the commitment scheme and equivocate the reference signature to be valid with a given message. Since our main scheme in Section 4 already integrate a trapdoor commitment scheme in its construction, it would seem possible to move the commitment part of the verification key into the CRS. And we mostly follow this way. A formal proof however reveals that we need to have k flexible pairings to sign messages of size k, $k \geq 1$, without needing the trapdoor for the commitment part. This results in relying on k-SFP rather than SFP when dealing with messages of size $k \geq 2$.

6.2 Definitions

Definition 3 (Simulatable Signature Scheme). A simulatable signature scheme SSIG consists of algorithms $\mathsf{SSIG}.\{\mathsf{Crs}, \mathsf{Key}, \mathsf{Chk}, \mathsf{Sign}, \mathsf{Vrf}, \mathsf{Sim}\}$ where $\mathsf{SSIG}.\{\mathsf{Key}, \mathsf{Sign}, \mathsf{Vrf}\}$ constitute a regular signature scheme (except that they take the CRS), and the extra algorithms works as follows.

$\mathsf{SSIG.Crs}(1^\lambda)$: A CRS generation algorithm that, on input security parameter λ, outputs a common reference string Σ and a trapdoor τ.

$\mathsf{SSIG.Chk}(\Sigma, vk)$: A verification key checking algorithm that, on input a verification key, returns 1 or 0.

$\mathsf{SSIG.Sim}(\Sigma, vk, m, \tau)$: A signature simulation algorithm that computes a signature σ for message m by using trapdoor τ.

By $\mathcal{M}_{vk}$, we denote the message space associated to vk. By $\mathcal{K}$, we denote the set of (vk, sk) that can be generated by $\mathsf{SSIG.Key}(\Sigma)$. Also by $\mathcal{S}_{sk,m}$ we denote the set of signatures that can be generated by $\mathsf{SSIG.Sign}(\Sigma, sk, m)$.

Completeness is defined in a standard way; with respect to correctly generated CRS, verification keys, and signatures, the verification function outputs 1 with probability 1.

Signature simulatability is defined in such a way that whenever adversary selects an appropriate message and verification key, then, by using the trapdoor of the CRS, it is possible to generate a signature that could have been generated by the proper signing operation. Formal definition follows.

Definition 4 (Signature-Simulatability). A signature scheme in the CRS model is simulatable if, for every CRS Σ generated by $(\Sigma, \tau) \leftarrow \mathsf{SSIG.Crs}(1^\lambda)$, for any (m, vk), if $1 = \mathsf{SSIG.Chk}(\Sigma, vk) \wedge m \in \mathcal{M}_{vk}$, then there exists sk such that $(vk, sk) \in \mathcal{K}$, and $1 = \mathsf{SSIG.Vrf}(\Sigma, vk, m, \sigma)$ holds for any $\sigma \leftarrow \mathsf{SSIG.Sim}(\Sigma, vk, m, \tau)$.

A relaxation would allow a negligible error in $\mathsf{SSIG.Vrf}$ for a message and a verification key chosen by an adversary. Note that the signature simulatability does not require simulated signatures be indistinguishable from the real ones. It is considered as a role of witness indistinguishable proof system coupled with the signature scheme.

Unforgeability is defined with respect to adaptive chosen message attacks. In the CRS model, however, a CRS is used for generating many keys and therefore, we must be careful that the keys should not be badly affected each other. By reflecting this concern, we allow an adversary to access an oracle that outputs correctly generated verification keys with respect to the same CRS. Furthermore, in our potential applications, the adversary is given a witness indistinguishable proof of holding a correct signature with respect to a given message and verification key. Let $\pi \leftarrow \mathsf{NIWI.Prf}((\Sigma, vk_i, m), \sigma)$ denote the proof system for this purpose. Here (Σ, vk_i, m) is public, and σ is the witness, and π is the proof. We do not give much details to the proof system as only the property needed in this formulation is the witness indistinguishability. The CRS for this proof system is implicitly given to the adversary. In summary the attack model includes the following three oracles.

- (Key Generation Oracle $\mathcal{O}_{\mathsf{vk}}$): On receiving i-th request, compute $(vk_i, sk_i) \leftarrow \mathsf{SSIG.Key}(\Sigma)$, and return vk_i. Record vk_i to Q_K.
- (Signing Oracle $\mathcal{O}_{\mathsf{sign}}$): On input (vk_i, m), return $\perp$ if vk_i is not recorded. Otherwise, compute $\sigma \leftarrow \mathsf{SSIG.Sign}(\Sigma, sk_i, m)$ and return σ. Record m to $Q_m^{vk_i}$.
- (Proof Oracle $\mathcal{O}_{\mathsf{wi}}$): On input (vk_i, m), return $\perp$ if $0 \leftarrow \mathsf{SSIG.Chk}(\Sigma, vk_i)$ or $m \notin \mathcal{M}_{vk_i}$. Otherwise, compute $\sigma \leftarrow \mathsf{SSIG.Sim}(\Sigma, vk_i, m, \tau)$, and $\pi \leftarrow \mathsf{NIWI.Prf}((\Sigma, vk_i, m), \sigma)$. Then return π.

Definition 5 (Unforgeability with WI-Simulation). A signature scheme in the CRS model is unforgeable against adaptive chosen message and random verification key attacks with witness-indistinguishable simulation if, for any polynomial-time adversary $\mathcal{A}$, the following experiment returns 1 with negligible probability.

Experiment :
$(\Sigma, \tau) \leftarrow \mathsf{SSIG.Crs}(1^\lambda)$
$(m^\star, \sigma^\star, vk^\star) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{sign}}, \mathcal{O}_{\mathsf{vk}}, \mathcal{O}_{\mathsf{wi}}}(\Sigma)$
Return 1 if $vk^\star \in Q_K$ and $m \notin Q_m^{vk^\star}$ and $1 \leftarrow \mathsf{SSIG.Vrf}(\Sigma, vk^\star, m^\star, \sigma^\star)$.
Return 0, otherwise.

6.3 Construction

Let $\Lambda = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \tilde{g}) \in \{\Lambda_{\mathsf{sym}}, \Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$ be implicitly given to the algorithms below.

- $\mathsf{SSIG.Crs}(1^\lambda)$: Choose random generators g_z, h_z, g_r, h_u from $\mathbb{G}_1^*$. For $i = 1, \ldots, k$, choose χ_i, γ_i and δ_i from $\mathbb{Z}_p^*$ and compute $g_i = g_z^{\chi_i} g_r^{\gamma_i}$ and $h_i = h_z^{\chi_i} h_u^{\delta_i}$. The CRS is set to $\Sigma = (g_z, h_z, g_r, h_u, \{g_i, h_i\}_{i=1}^k)$, and the trapdoor is $tk = (\chi_1, \gamma_1, \delta_1, \ldots, \chi_k, \gamma_k, \delta_k)$.

- $\mathsf{SSIG.Key}(\Sigma)$: Choose $\alpha, \beta \leftarrow \mathbb{Z}_p^*$ and compute $\{a_i, \tilde{a}_i\}_{i=0}^k \leftarrow \mathsf{Extend}(g_r, \tilde{g}^\alpha)$ and $\{b_i, \tilde{b}_i\}_{i=0}^k \leftarrow \mathsf{Extend}(h_u, \tilde{g}^\beta)$. Let $sk = (\alpha, \beta)$. For some default message $\vec{m}^* \in \mathbb{G}_2^k$, compute a reference signature $\sigma^* = \mathsf{SSIG.Sign}(\Sigma, sk, \vec{m}^*)$ as shown below. Let $vk = (\{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=0}^k, \sigma^*)$. Output (vk, sk).

- $\mathsf{SSIG.Sign}(\Sigma, sk, \vec{m})$: For $i = 1$ to k and randomly chosen $\zeta_i, \rho_i, \varphi_i \leftarrow \mathbb{Z}_p^*$, set

$$\begin{aligned}
&(\text{If } m_i \neq 1): \quad s'_i = g_z^{\zeta_i} g_r^{\rho_i} g_i^{-1}, \quad t'_i = m_i, \quad v'_i = h_z^{\zeta_i} h_u^{\varphi_i} h_i^{-1}, \quad w'_i = m_i, \\
&(\text{If } m_i = 1): \quad s'_i = g_z^{\zeta_i} g_r^{\rho_i}, \qquad t'_i = \tilde{g}, \qquad v'_i = h_z^{\zeta_i} h_u^{\varphi_i}, \qquad w'_i = \tilde{g}.
\end{aligned}$$

  and

$$z = \prod_{i=1}^k t_i'^{-\zeta_i}, \qquad r = \tilde{g}^\alpha \prod_{i=1}^k t_i'^{-\rho_i}, \qquad u = \tilde{g}^\beta \prod_{i=1}^k w_i'^{-\varphi_i}.$$

  Then, compute $\{s_i, t_i\}_{i=1}^k \leftarrow \mathsf{RandSeq}(\{s'_i, t'_i\}_{i=1}^k)$ and $\{v_i, w_i\}_{i=1}^k \leftarrow \mathsf{RandSeq}(\{v'_i, w'_i\}_{i=1}^k)$. Output $\sigma = (z, r, u, \{s_i, t_i, v_i, w_i\}_{i=1}^k)$ as a signature.

- $\mathsf{SSIG.Vrf}(\Sigma, vk, \vec{m}, \sigma)$: Parse σ as $(z, r, u, \{s_i, t_i, v_i, w_i\}_{i=1}^k)$. Output 1 if

$$A = e(g_z, z)\, e(g_r, r) \prod_{i=1}^k e(g_i, m_i)\, e(s_i, t_i), \quad \text{and} \tag{13}$$

$$B = e(h_z, z)\, e(h_u, u) \prod_{i=1}^k e(h_i, m_i)\, e(v_i, w_i) \tag{14}$$

  hold for $A = \prod_{i=0}^k e(a_i, \tilde{a}_i)$ and $B = \prod_{i=0}^k e(b_i, \tilde{b}_i)$. Output 0, otherwise.

- $\mathsf{SSIG.Chk}(\Sigma, vk)$: Parse vk into $(\{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=0}^k, \sigma^*)$ and return 0 if it fails. Check if every element but σ^* is in appropriate group $\mathbb{G}_1$ or $\mathbb{G}_2$, and verify that $1 = \mathsf{SSIG.Vrf}(\Sigma, vk, \vec{m}^*, \sigma^*)$. If any of the checks fail, output 0. Otherwise, output 1.

- $\mathsf{SSIG.Sim}(\Sigma, vk, \vec{m}, tk)$: Take σ^* from vk and parse it into $(z, r, u, \{s_i, t_i, v_i, w_i\}_{i=1}^k)$. By using $tk = (\chi_1, \gamma_1, \delta_1, \ldots, \chi_k, \gamma_k, \delta_k)$, compute (z', r', u') as

$$z' = z \cdot \prod_{i=1}^k (m_i/m_i^*)^{-\chi_i}, \quad r' = r \cdot \prod_{i=1}^k (m_i/m_i^*)^{-\gamma_i}, \text{ and } \quad u' = u \cdot \prod_{i=1}^k (m_i/m_i^*)^{-\delta_i}.$$

  Output $\sigma = (z', r', u', \{s_i, t_i, v_i, w_i\}_{i=1}^k)$ as a signature for $\vec{m}$.

6.4 Security

The security of SSIG relies on k-SFP, a generalization of SFP that has k flexible pairings in each relation as formally defined below. In the case of $k = 1$, k-SFP becomes SFP.

Assumption 4 (Simultaneous k-Flexible Pairing Assumption (k-SFP)). Let Λ be a common parameter and let g_z, h_z, g_r, and h_u be random generators of $\mathbb{G}_1$. Let $\{(a_i, \tilde{a}_i), (b_i, \tilde{b}_i)\}_{i=1}^k$ be random elements in $(\mathbb{G}_1 \times \mathbb{G}_2)^{2k}$. For $j = 1, \ldots, q$, let R_j be a tuple $(z, r, u, \{s_i, t_i, v_i, w_i\}_{i=1}^k) \in {\mathbb{G}_2}^3 \times (\mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_2)^k$ that satisfies

$$\prod_{i=1}^{k} e(a_i, \tilde{a}_i) = e(g_z, z)\ e(g_r, r)\ \prod_{i=1}^{k} e(s_i, t_i), \quad \text{and} \tag{15}$$

$$\prod_{i=1}^{k} e(b_i, \tilde{b}_i) = e(h_z, z)\ e(h_u, u)\ \prod_{i=1}^{k} e(v_i, w_i). \tag{16}$$

Given Λ, g_z, h_z, g_r, h_u, $\{(a_i, \tilde{a}_i), (b_i, \tilde{b}_i)\}_{i=1}^k$, and uniformly chosen $R_1, \ldots, R_q$, it is hard to find $(z^\star, r^\star, u^\star, \{s_i^\star, t_i^\star, v_i^\star, w_i^\star\}_{i=1}^k)$, that fulfill relations (15) and (16). A restriction is that $z^\star \neq 1$ and $z^\star \neq z \in R_j$ for every R_j.

Theorem 6. *For any generic algorithm $\mathcal{A}$, the probability that $\mathcal{A}$ breaks k-SFP with ℓ group operations and pairings is bound by $\mathcal{O}(k^2 \cdot q^2 + \ell^2)/p$.*

Proof of Theorem 6 that justifies the assumption in the generic bilinear group model is in Appendix A.3. As well as Theorem 3, k-SFP implies SDP for any $k \geq 1$. Somewhat contradictory to the fact that k-SFP is a generalization of SFP, we do not see useful reduction between them for $k \geq 2$.

Theorem 7. *Signature scheme SSIG is correct and signature-simulatable. It is EUF-CMA with WI-simulation in the multi-user setting if k-SFP holds for Λ.*

Proof. Correctness. Let I (and I^*) denote the set of indexes where $m_i \neq 1$ (and $m_i = 1$, respectively) in SIG.Sign. Regarding the first relation in the verification predicates, we have:

$$\begin{aligned}
&e(g_z, z)\ e(g_r, r)\ \prod_{i=1}^{k} e(g_i, m_i)\ e(s_i, t_i) \\
&= e\left(g_z, \prod_{i=1}^{k} {t'_i}^{-\zeta_i}\right)\ e\left(g_r, \tilde{g}^\alpha \prod_{i=1}^{k} {t'_i}^{-\rho_i}\right)\ \prod_{i \in I} e\left(g_i, m_i\right)\ e\left(g_z^{\zeta_i} g_r^{\rho_i} g_i^{-1}, t'_i\right)\ \prod_{i \in I^*} e\left(g_z^{\zeta_i} g_r^{\rho_i}, t'_i\right) \\
&= e\left(g_z, \prod_{i=1}^{k} {t'_i}^{-\zeta_i}\right)\ e\left(g_r, \tilde{g}^\alpha \prod_{i=1}^{k} {t'_i}^{-\rho_i}\right)\ \prod_{i=1}^{k} e\left(g_z^{\zeta_i} g_r^{\rho_i}, t'_i\right) \\
&= e(g_r, \tilde{g}^\alpha) = A
\end{aligned}$$

The other relation can be verified in the same manner as $z = \prod_{i=1}^k {t'_i}^{-\zeta_i} = \prod_{i=1}^k {w'_i}^{-\zeta_i}$.

Signature-Simulatability. For every $vk = (\{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=1}^k, \sigma^*)$ such that $1 = \mathsf{SSIG.Chk}(\Sigma, vk)$, every elements in $\{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=1}^k$ is in the correct group $\mathbb{G}_1$ and $\mathbb{G}_2$. Clearly there are (α, β) so that $\prod_{i=1}^k e(a_i, \tilde{a}_i) = e(g_r, \tilde{g}^\alpha)$ and $\prod_{i=1}^k e(b_i, \tilde{b}_i) = e(h_u, \tilde{g}^\beta)$ hold. Therefore such $\{(a_i, \tilde{a}_i, b_i, \tilde{b}_i)\}_{i=1}^k$ and (α, β) are a correct key pair. The rest is to show that SSIG.Sim correctly works to turn valid signature $\sigma^* = (z, r, u, \{s_i, t_i, v_i, w_i\}_{i=1}^k)$ for $\vec{m}^*$ into a signature $\sigma = (z', r', u', \{s_i, t_i, v_i, w_i\}_{i=1}^k)$ for

message $\vec{m}$. It holds that

$$\begin{aligned}
&e(g_z, z')\, e(g_r, r') \prod_{i=1}^{k} e(g_i, m_i)\, e(s_i, t_i) \\
&= e(g_z, z \cdot \prod_{i=1}^{k} (m_i/m_i^*)^{-\chi_i})\, e(g_r, r \cdot \prod_{i=1}^{k} (m_i/m_i^*)^{-\gamma_i}) \prod_{i=1}^{k} e(g_i, m_i)\, e(s_i, t_i) \\
&= e(g_z, z)\, e(g_r, r) \prod_{i=1}^{k} e(g_i, m_i^*)\, e(s_i, t_i) = A.
\end{aligned}$$

The other relation $e(h_z, z')\, e(h_u, u') \prod_{i=1}^{k} e(h_i, m_i)\, e(v_i, w_i) = B$ can be verified in the same way. Thus the output from $\mathsf{SSIG.Sim}$ is a valid signature for $\vec{m}$.

EUF-CMA WITH WI-SIMULATION. Given an instance of k-SFP, we simulate the view of $\mathcal{A}$ in the attack environment as follows.

- (CRS generation) : Do the same as original $\mathsf{SSIG.Crs}$ by using given generators (g_r, h_u, g_z, h_z) in the input instance. The commitment-key is assigned as a CRS, $\Sigma = (g_z, h_z, g_r, h_u, \{g_i, h_i\}_{i=1}^k)$, and the trapdoor is $tk = (\chi_1, \gamma_1, \delta_1, \ldots, \chi_k, \gamma_k, \delta_k)$.

- (Key Generation Oracle $\mathcal{O}_{\mathsf{vk}}$) : Take $\{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=1}^k$ from the input instance. Choose $\zeta, \rho, \varphi \leftarrow \mathbb{Z}_p^*$ and $\tilde{g} \leftarrow \mathbb{G}_2^*$. Then compute $\{a'_i, \tilde{a}'_i\}_{i=0}^k \leftarrow \mathsf{RandSeq}((g_z^\zeta g_r^\rho, \tilde{g}), (a_1, \tilde{a}_1), \ldots, (a_k, \tilde{a}_k))$ and $\{b'_i, \tilde{b}'_i\}_{i=0}^k \leftarrow \mathsf{RandSeq}((h_z^\zeta h_u^\varphi, \tilde{g}), (b_1, \tilde{b}_1), \ldots, (b_k, \tilde{b}_k))$. Then simulate a reference signature σ_0 as described below. The verification key is $vk = (\{a'_i, \tilde{a}'_i, b'_i, \tilde{b}'_i\}_{i=0}^k, \sigma_0)$. Record vk to Q_K.

- (Signing Oracle $\mathcal{O}_{\mathsf{sign}}$) : Given message $\vec{m}$ and vk, return $\perp$ if vk is not in Q_K. Take a new tuple $R_j = (z_j, r_j, u_j, \{s_{ij}, t_{ij}, v_{ij}, w_{ij}\}_{i=1}^k)$ from the given instance. Then compute

$$z'_j = z_j\, \tilde{g}^\zeta \prod_{i=1}^{k} m_i^{-\chi_i}, \quad r'_j = r_j\, \tilde{g}^\rho \prod_{i=1}^{k} m_i^{-\gamma_i}, \quad u'_j = u_j\, \tilde{g}^\varphi \prod_{i=1}^{k} m_i^{-\delta_i}. \tag{17}$$

  by using (ζ, ρ, φ) used for generating vk in $\mathcal{O}_{\mathsf{vk}}$. The signature is $\sigma_j = (z'_j, r'_j, u'_j, \{s_{ij}, t_{ij}, v_{ij}, w_{ij}\}_{i=1}^k)$.

- (Simulation Oracle $\mathcal{O}_{\mathsf{wi}}$) : Given $\vec{m}$ and vk, return $\perp$ if $0 \leftarrow \mathsf{SSIG.Chk}(\Sigma, vk_i)$ or $m \notin \mathcal{M}_{vk_i}$. If vk is in Q_K, compute $\sigma \leftarrow \mathcal{O}_{\mathsf{sign}}(\vec{m}, vk)$. Otherwise, compute $\sigma \leftarrow \mathsf{SSIG.Sim}(\Sigma, vk, \vec{m}, tk)$. Then compute $\pi \leftarrow \mathsf{NIWI.Prf}((\Sigma, vk, m), \sigma)$ and return π.

When $\mathcal{A}$ outputs $(\vec{m}^\dagger, z^\dagger, r^\dagger, u^\dagger, \{s_i^\dagger, v_i^\dagger, t_i^\dagger, w_i^\dagger\}_{i=1}^k)$, compute

$$z^\star = \left(z^\dagger\right) \tilde{g}^{-\zeta} \prod_{i=1}^{k} \left(m_i^\dagger\right)^{\chi_i}, \quad r^\star = \left(r^\dagger\right) \tilde{g}^{-\rho} \prod_{i=1}^{k} \left(m_i^\dagger\right)^{\gamma_i}, \quad u^\star = \left(u^\dagger\right) \tilde{g}^{-\varphi} \prod_{i=1}^{k} \left(m_i^\dagger\right)^{\delta_i}, \tag{18}$$

and set $s_i^\star = s_i^\dagger$, $t_i^\star = t_i^\dagger$, $v_i^\star = v_i^\dagger$, and $w_i^\star = w_i^\dagger$ for $i = 1, \ldots, k$. The reduction algorithm outputs a tuple $(z^\star, r^\star, u^\star, \{s_i^\star, t_i^\star, v_i^\star, w_i^\star\}_{i=1}^k)$ and terminates.

It can be verified by inspection that the CRS, the verification-key and the signatures perfectly follow the legitimate distribution. When $\mathcal{A}$ is successful, for the outputs of the reduction algorithm,

it holds that

$$e(g_z, z^\star)\ e(g_r, r^\star)\ \prod_{i=1}^{k} e(s_i^\star, t_i^\star) = e\left(g_z, \left(z^\dagger\right)\ \tilde{g}^{-\zeta} \prod_{i=1}^{k} \left(m_i^\dagger\right)^{\chi_i}\right)\ e\left(g_r, \left(r^\dagger\right)\ \tilde{g}^{-\rho} \prod_{i=1}^{k} \left(m_i^\dagger\right)^{\gamma_i}\right)\ \prod_{i=1}^{k} e\left(s_i^\dagger, t_i^\dagger\right)$$

$$= e\left(g_z^{-\zeta} g_r^{-\rho}, \tilde{g}\right)\ e\left(g_z, z^\dagger\right)\ e\left(g_r, r^\dagger\right)\ \prod_{i=1}^{k} e\left(g_i, m_i^\dagger\right)\ e\left(s_i^\dagger, t_i^\dagger\right)$$

$$= e(a_0, \tilde{a}_0)^{-1}\ \prod_{i=0}^{k} e(a_i, \tilde{a}_i) = \prod_{i=1}^{k} e(a_i, \tilde{a}_i).$$

One can also verify that $e(g_z, z^\star)\ e(h_u, u^\star)\ \prod_{i=1}^{k} e(v_i^\star, w_i^\star) = \prod_{i=1}^{k} e(b_i, \tilde{b}_i)$ holds in the same way.

What remains is to show that $z^\star$ is not in $\{1, z_1, \ldots, z_q\}$. Basically the argument is the same as the one in the proof of Theorem 4 in Section 4.3. For the same argument to hold, we have to show that trapdoor tk used for simulating $\mathcal{O}_{\mathsf{wi}}$ is information theoretically hidden even after π is seen by the adversary. For vk generated by $\mathcal{O}_{\mathsf{vk}}$, simulation is done just by calling the signing oracle. So the same argument applies. On the other hand, for vk that is not generated by $\mathcal{O}_{\mathsf{vk}}$, SSIG.Chk guarantees that there exists a corresponding signing key sk. Since NIWI.Prf is witness indistinguishable, there exists a randomness that is consistent to a valid signature that cold have been generated by the signing key. Clearly, that signature can be generated without using tk. Thus tk remains independent from the view of the adversary. ∎

7 Signing Mixed-Group Messages in the SXDH Setting

7.1 Overview

By using the idea of signature chaining, we construct a signature scheme whose message space consists of a mixture of $\mathbb{G}_1$ and $\mathbb{G}_2$. We use two signature schemes: SIG1 signing messages from the space $\mathbb{G}_1^{k_1}$ and SIG2 signing messages from the space $\mathbb{G}_2^{k_2}$. A part of a signature from SIG2 is included into the message given to SIG1 as a joint. Surprisingly, the joint can be as minimal as only one group element s in this case.

7.2 Construction

Let SIG2 be the constant-size signature scheme from Section 4, whose message space is $\mathbb{G}_2^{k_2}$. Let SIG1 be a 'dual' scheme obtained by exchanging $\mathbb{G}_1$ and $\mathbb{G}_2$ in the same scheme. Let the message space of SIG1 is $\mathbb{G}_1^{k_1+1}$. (Note that we use the same letters for variables in a signature. Accordingly, z, r, u, t, and w are in the same group as the input message while s and v are in the other group.) By using these signature scheme, we construct signature scheme XSIG whose message space is $\mathbb{G}_1^{k_1} \times \mathbb{G}_2^{k_2}$ as follows. Let $(m, \tilde{m})$ be a message in $\mathbb{G}_1^{k_1} \times \mathbb{G}_2^{k_2}$. For vector $m \in \mathbb{G}_1^{k_1}$ and single element $s \in \mathbb{G}_1$, let $m||s$ denote a vector in $\mathbb{G}_1^{k_1+1}$ obtained by appending s to the end of vector m. Let $\Lambda = \Lambda_{\mathsf{sxdh}}$ be given to the functions described below.

- $\mathsf{XSIG.Key}(1^\lambda)$: Run $(vk_1, sk_1) \leftarrow \mathsf{SIG1.Key}(1^\lambda)$ and $(vk_2, sk_2) \leftarrow \mathsf{SIG2.Key}(1^\lambda)$. Output $(vk, sk) = ((vk_1, vk_2), (sk_1, sk_2))$.
- $\mathsf{XSIG.Sign}(sk, (\vec{m}, \vec{\tilde{m}}))$: Run $\sigma_2 = (z, r, s, t, u, v, w) \leftarrow \mathsf{SIG2.Sign}(sk_2, \vec{\tilde{m}})$ and $\sigma_1 = (z', r', s', t', u', v', w') \leftarrow \mathsf{SIG1.Sign}(sk_1, \vec{m}||s)$. Output $\sigma = (\sigma_1, \sigma_2)$.

- $\mathsf{XSIG.Vrf}(vk, (\vec{m}, \vec{\tilde{m}}), (\sigma_1, \sigma_2))$: Take $s \in \mathbb{G}_1$ from σ_2. Run $b_2 = \mathsf{SIG2.Vrf}(vk_2, \vec{\tilde{m}}, \sigma_2)$ and $b_1 = \mathsf{SIG1.Vrf}(vk_1, \vec{m}||s, \sigma_1)$. Output 1 if $b_2 = b_1 = 1$. Output 0, otherwise.

7.3 Security

Theorem 8. *If SIG1 and SIG2 are EUF-CMA, so is XSIG.*

Proof. Suppose that there is a successful adversary that launches chosen message attacks and outputs a valid forgery, $((\vec{m}^\dagger, \vec{\tilde{m}}^\dagger), (\sigma_1^\dagger, \sigma_2^\dagger))$. Consider $s^\dagger$ included in $\sigma_2^\dagger$. Observe that $\sigma_1^\dagger$ is a signature for $m^\dagger||s^\dagger$. We then have 3 cases.

Type-I $\vec{m}^\dagger||s^\dagger$ has never been signed by the signing oracle. This case contradicts to the unforgeability of SIG1.

Type-II $\vec{\tilde{m}}^\dagger$ has never been signed by the signing oracle. This case contradicts to the unforgeability of SIG2.

Type-III Both $\vec{m}^\dagger||s^\dagger$ and $\vec{\tilde{m}}^\dagger$ have been signed by the signing oracle in separate queries. This case contradicts to the DBP assumption.

Since the first two forgery cases are trivial, we focus on Type-III. We construct a reduction algorithm that simulates the environment for adversary $\mathcal{A}$ launching an adaptive chosen message attack on XSIG. The simulator only simulates SIG2 and honestly acts with respect to SIG1. We thus describe the simulation only with respect to SIG2. Given an instance of the DBP assumption, (Λ, g_z, g_r), the simulator works as follows:

- (Key Generation): Choose random h_z and h_u from $\mathbb{G}_1^*$. Then, for $i = 1, \ldots, k_2$, set $g_i = g_z^{\chi_i} g_r^{\gamma_i}$ and $h_i = h_z^{\chi_i} h_u^{\delta_i}$ for random χ_i, γ_i, and δ_i in $\mathbb{Z}_p^*$. Choose α, β from $\mathbb{Z}_p^*$ and $\tilde{g}$ from $\mathbb{G}_2^*$. Then compute $\{a_i, \tilde{a}_i\}_{i=0}^1 \leftarrow \mathsf{Extend}(g_r, \tilde{g}^\alpha)$ and $\{b_i, \tilde{b}_i\}_{i=0}^1 \leftarrow \mathsf{Extend}(h_u, \tilde{g}^\beta)$. Output $vk_1 = (g_z, h_z, g_r, h_u, \{g_i, h_i\}_{i=1}^{k_2}, \{a_i, \tilde{a}_i, b_i, \tilde{b}_i\}_{i=0}^1)$.
- (Signature Issuing): Given message $\tilde{m} \in \mathbb{G}_2^{k_2}$, choose ζ, ρ, τ, φ, $\omega \leftarrow \mathbb{Z}_p^*$ and set

$$z = \tilde{g}^\zeta \prod_{i=1}^{k_2} \tilde{m}_i^{-\chi_i}, \quad r = \tilde{g}^{\zeta\rho/\tau+\alpha} \prod_{i=1}^{k_2} \tilde{m}_i^{-\gamma_i}, \quad s = g_z^\tau g_r^\rho, \quad t = \tilde{g}^{-\zeta/\tau}$$

$$u = \tilde{g}^{\zeta\varphi/\omega+\beta} \prod_{i=1}^{k_2} \tilde{m}_i^{-\delta_i}, \quad v = h_z^\omega h_u^\varphi, \quad w = \tilde{g}^{-\zeta/\omega}.$$

  Output $\sigma_2 = (z, r, s, t, u, v, w)$.

To see the correctness of the simulated signatures, observe that

$$\begin{aligned}
& e(g_z, z)\ e(g_r, r)\ e(s, t) \prod_{i=1}^{k_2} e(g_i, m_i) \\
&= e\left(g_z, \tilde{g}^\zeta \prod_{i=1}^{k_2} m_i^{-\chi_i}\right)\ e\left(g_r, \tilde{g}^{\zeta\rho/\tau+\alpha} \prod_{i=1}^{k_2} m_i^{-\gamma_i}\right)\ e\,(s, t)\ \prod_{i=1}^{k_2} e\,(g_z^{\chi_i} g_r^{\gamma_i}, m_i) \\
&= e\left(g_z, \tilde{g}^\zeta\right)\ e\left(g_r, \tilde{g}^{\zeta\rho/\tau+\alpha}\right)\ e\left(g_z^\tau g_r^\rho, \tilde{g}^{-\zeta/\tau}\right) \\
&= e(g_r, \tilde{g}^\alpha) = \prod_{i=0}^{1} e(a_i, \tilde{a}_i)
\end{aligned}$$

holds. The other verification predicate holds in the same way. It is also not hard to inspect that the distribution of signatures is statistically close to the original one due to the random coins in the simulation.

Let $\sigma^\dagger = (\sigma_1^\dagger, \sigma_2^\dagger)$, where $\sigma_2^\dagger = (z^\dagger, r^\dagger, s^\dagger, t^\dagger, u^\dagger, v^\dagger, w^\dagger)$, be the forged signature for a message $(\vec{m}^\dagger, \vec{\tilde{m}}^\dagger)$. By the forgery type constrains, there exists a signing query with message $(\vec{m}^\dagger, \vec{\tilde{m}})$ such that $\vec{\tilde{m}} \neq \vec{\tilde{m}}^\dagger$ and its signature $\sigma_2 = (z, r, s, t, u, v, w)$ satisfies $s = s^\dagger$. Accordingly, we have

$$e(g_z, z^\dagger)\ e(g_r, r^\dagger)\ e(s^\dagger, t^\dagger)\ \prod_{i=1}^{k_2} e(g_i, \tilde{m}_i^\dagger) = e(g_z, z)\ e(g_r, r)\ e(s^\dagger, t)\ \prod_{i=1}^{k_2} e(g_i, \tilde{m}_i). \tag{19}$$

Recall that $s^\dagger = s = g_z^\tau g_r^\rho$. By dividing the left-hand of the above equation by its right-hand, we have

$$\begin{aligned}
1 &= e\left(g_z, \frac{z^\dagger}{z}\right)\ e\left(g_r, \frac{r^\dagger}{r}\right)\ e\left(s^\dagger, \frac{t^\dagger}{t}\right)\ \prod_{i=1}^{k_2} e\left(g_i, \frac{\tilde{m}_i^\dagger}{\tilde{m}_i}\right) \\
&= e\left(g_z, \frac{z^\dagger}{z} \prod_{i=1}^{k_2} (\frac{\tilde{m}_i^\dagger}{\tilde{m}_i})^{\chi_i}\right)\ e\left(g_r, \frac{r^\dagger}{r} \prod_{i=1}^{k_2} (\frac{\tilde{m}_i^\dagger}{\tilde{m}_i})^{\gamma_i}\right)\ e\left(g_z^\tau g_r^\rho, \frac{t^\dagger}{t}\right) \\
&= e(g_z, z^\star)\ e(g_r, r^\star),
\end{aligned}$$

where $z^\star = \frac{z^\dagger}{z}(\frac{t^\dagger}{t})^\tau \prod_i (\frac{\tilde{m}_i^\dagger}{\tilde{m}_i})^{\chi_i}$ and $r^\star = \frac{r^\dagger}{r}(\frac{t^\dagger}{t})^\rho \prod_i (\frac{\tilde{m}_i^\dagger}{\tilde{m}_i})^{\gamma_i}$.

Since $\vec{\tilde{m}}^\dagger \neq \vec{\tilde{m}}$, there exists $i^\star$ such that $\tilde{m}_{i^\star}^\dagger / \tilde{m}_{i^\star} \neq 1$. Observe that $\chi_{i^\star}$ is independent of the view of the adversary. Hence the probability that $z^\star = 1$ is negligible. The reduction algorithm outputs $(z^\star, r^\star)$ as a valid answer to the given instance of DBP. ∎

8 Strongly Unforgeable Signatures

8.1 A Generic Construction

We first show a generic construction of sEUF-CMA signature scheme with constant-size signatures. Let SIG be a signature scheme with EUF-CMA property, and OTS be a one-time signature scheme that is strongly unforgeable against one-time selective-message attacks. (In the one-time selective-message attacks, the adversary has to select a message before seeing the verification-key and creates a forged signature on that message. This is a weaker notion than regular one-time chosen message attacks.) The construction requires that the message space of SIG covers the public-key space of OTS.

- $\mathsf{FSIG1.Key}(1^\lambda)$: Run $(vk, sk) \leftarrow \mathsf{SIG.Key}(1^\lambda)$. Output (vk, sk).
- $\mathsf{FSIG1.Sign}(sk, \vec{m})$: $(vk_o, sk_o) \leftarrow \mathsf{OTS.Key}(1^\lambda)$, $\sigma_1 \leftarrow \mathsf{SIG.Sign}(sk, vk_o || \vec{m})$, $\sigma_2 \leftarrow \mathsf{OTS.Sign}(sk_o, \sigma_1)$. Output $\sigma = (vk_o, \sigma_1, \sigma_2)$.
- $\mathsf{FSIG1.Vrf}(vk, \vec{m}, \sigma)$: Parse σ into $(vk_o, \sigma_1, \sigma_2)$. Compute $b_1 \leftarrow \mathsf{SIG.Vrf}(vk, vk_o || \vec{m}, \sigma_1)$ and $b_2 \leftarrow \mathsf{OTS.Vrf}(vk_o, \sigma_1, \sigma_2)$. Output 1 if $b_2 = b_1 = 1$. Output 0, otherwise.

Theorem 9. *Signature scheme FSIG1 is strongly EUF-CMA if SIG is EUF-CMA and OTS is strongly unforgeable against one-time selective-message attacks.*

Proof. Let $\mathcal{O}_{\mathsf{sign}}$ be the signing oracle of FSIG1. Suppose that an adversary outputs a valid forgery $(vk_o^\dagger, \sigma_1^\dagger, \sigma_2^\dagger, \vec{m}^\dagger)$. Let $Q_i = (vk_o, \sigma_1, \sigma_2, \vec{m})$ for $i = 1, \ldots, q$ be the record of interaction between the adversary and $\mathcal{O}_{\mathsf{sign}}$. We consider two cases;

Case 1: $(vk_o^\dagger, \vec{m}^\dagger) \neq (vk_o, \vec{m})$ for any Q_i. Since $1 = \mathsf{SIG.Vrf}(vk, vk_o^\dagger || \vec{m}^\dagger, \sigma_1^\dagger)$ holds, $(\vec{m}^\star, \sigma^\star) = (vk_o^\dagger || \vec{m}^\dagger, \sigma_1^\dagger)$ is a successful forgery with respect to SIG. This contradicts to the EUF-CMA property of SIG.

Case 2: $(vk_o^\dagger, \vec{m}^\dagger) = (vk_o, \vec{m})$ and $(\sigma_1^\dagger, \sigma_2^\dagger) \neq (\sigma_1, \sigma_2)$ for some $Q_{i^\star}$. Since $1 = \mathsf{OTS.Vrf}(vk_o^\dagger, \sigma_1^\dagger, \sigma_2^\dagger) = \mathsf{OTS.Vrf}(vk_o, \sigma_1, \sigma_2)$ for $vk_o^\dagger = vk_o$, the fact that $(\sigma_1^\dagger, \sigma_2^\dagger) \neq (\sigma_1, \sigma_2)$ contradicts to the strong unforgeability of OTS with respect to vk_o.

Since $(vk_o^\dagger, \sigma_1^\dagger, \sigma_2^\dagger, \vec{m}^\dagger) \neq (vk_o, \sigma_1, \sigma_2, \vec{m})$ for all Q_i must hold for a successful forgery, the above cases cover all valid forgeries.

By instantiating SIG and OTS by the ones in Section 4 and Appendix C.1 with setting $\Lambda = \Lambda_{\mathsf{sym}}$, the resulting FSIG1 outputs a signature of 32 group elements ($|vk_o| = 22$, $|\sigma_1| = 7$, $|\sigma_2| = 3$) which is a constant in the size of $\vec{m}$.

8.2 More Efficient Non-Generic Construction

The construction is the same as that of FSIG1 with SIG from Section 4 and OTS from Appendix C.1. Only the difference is that OTS takes bases $(g_z, h_z, g_r, h_u, g_1, h_1, \ldots, g_7, h_7)$ from those of SIG. Then vk_o consits of 4 group elements, $(a, \tilde{a}, b, \tilde{b})$, which gives total singature of size 14. Let FSIG2 denote this signature scheme.

Theorem 10. *Signature scheme FSIG2 is sEUF-CMA if SFP holds for $\Lambda = \Lambda_{sym}$.*

Proof. First observe that we cannot show a black-box reduction to the security of SIG and OTS by using their signing oracles since they share the bases. We instead construct reduction to their underlying assumptions. This is possible because, in both security proofs for SIG and OTS, bases $(g_1, h_1, \ldots, g_7, h_7)$ are set in the same manner with respect to (g_z, h_z, g_r, h_u). Thus, while we simulate the signing oracle for SIG, we can also simulate signatures of OTS.

The outline of the proof is the same as that for Theorem 9. In case 1, we construct a reduction to SFP by simulating SIG as shown in the proof of Theorem 4.3. We also simulate OTS as shown in the proof of Theorem 16. Note that the simulation of OTS is possible since the way bases g_i and h_i are set in simulating SIG is exactly the same as that in simulating OTS. Thus we can successfully simulate FSIG2 by using these simulated SIG and OTS. It is important to see that exponents hidden in g_i and h_i remain independent of the view of the adversary even through the simulation of OTS as shown in the proof of Theorem 16. Thus a successful forgery results in a contradiction to SFP as shown in the proof of Theorem 4.3.

In case2, we show a reduction to SDP by simulating OTS as shown in the proof of Theorem 16. Since simulation of SIG needs an instance of SFP, we generate a random instance of SFP from that of SDP as follows. Given an SDP instance (g_z, h_z, g_r, h_u), set $(a, \tilde{a}) \leftarrow \mathsf{Rand}(g_r, \tilde{g}^\alpha)$ and $(b, \tilde{b}) \leftarrow \mathsf{Rand}(h_u, \tilde{g}^\beta)$. Then for $j = 1, \ldots, q$, compute reference $R_j = (z, r, u, s, t, v, w)$ by choosing $\zeta \leftarrow \mathbb{Z}_p$ and setting $z = \tilde{g}^\zeta$, $r' = \tilde{g}^\alpha$, $s' = g_z$, $t' = \tilde{g}^\zeta$, $u' = \tilde{g}^\beta$, $v' = h_z$, $w' = \tilde{g}^\zeta$ and applying $(r, s, t, u, v, w) \leftarrow \mathsf{SigRand}(r', s', t', u', v', w')$. The rest of the simulation for SIG is the same as that in the proof of Theorem 4.3. As well as the previous case, the simulation of SIG retains independence of exponents hidden in g_i and h_i. Thus a successful forgery contradicts to SDP as shown in the proof of Theorem 16.

Finally, applying Theorem 3 to reduce SDP to SFP completes the proof. ∎

9 Applications

In some cryptographic protocols, the existing state of the art constructions achieve the desired security properties with good efficiency, whereas for others certain compromises are made (achieving slightly weaker notion of security or being somewhat inefficient). Below we present constructions for round-optimal blind signatures following the framework of [24], the efficient instantiation of which has been an open problem since Crypto'06; and efficient fully secure group signatures supporting concurrent join procedure, with previous constructions being not in the standard model, secure under weaker model, not supporting concurrent join procedure, or being inefficient. Our signature schemes not only embody known modular protocol designs, but also achieve an excellent efficiency. These are good examples that enlightens the usefulness of modular protocol design and significance of developing efficient building blocks.

9.1 Round-Optimal Blind Signatures

We present an efficient instantiation of Fischlin's round-optimal blind signature scheme [24]. In fact, we use the modification of [36, 2] for which the generic construction uses a non-interactive witness indistinguishable (NIWI) proof system and a simulatable signature scheme. This gives the first efficient round-optimal non-committing blind signature scheme adaptively secure in the universally composability framework [20].

The structure of the framework is the following. A user commits to a message m with opening d and send the commitment c to the signer. The signer signs commitment c and return the signature σ to the user. Then the user computes a NIWI proof π with witness (c, d, σ) for the fact that he knows a commitment c of message m, he knows the correct opening d, and he has a valid signature on c with respect to a verification key vk of the signer. The security follows from the generic framework in [2].

To instantiate this generic scheme, we use the GS proof system, the simulatable signature scheme SSIG from Section 6 for $k = 1$ (i.e. for signing only a single group element), and the commitment scheme $\mathsf{TC2}$ in Appendix B.2. In fact, any commitment scheme suffices for our purpose as long as commitment key, commitments, and openings to be group elements and the verification is by pairing product equations. The choice of $\mathsf{TC2}$ is due to the efficiency; it has the smallest commitment size. The commitment scheme $\mathsf{TC2}$ could be viewed as a "pairing-based variant" of Pedersen commitment [44], and, indeed, is almost as efficient.

Let $\Lambda \in \{\Lambda_{\mathsf{sym}}, \Lambda_{\mathsf{sxdh}}\}$ be the common parameter. Let $(\Sigma_{\mathsf{com}}, tk) \leftarrow \mathsf{TC2.Key}(1^\lambda)$, $(\Sigma_{\mathsf{sig}}, tk') \leftarrow \mathsf{SSIG.Crs}(1^\lambda)$. Let Σ_{niwi} be the common reference string for the GS proof system in the simulation mode. Concretely, those are $\Sigma_{\mathsf{com}} = f \in \mathbb{G}_2$, $\Sigma_{\mathsf{sig}} = (g_z, h_z, g_r, h_u, g_1, h_1) \in \mathbb{G}_1^6$, and Σ_{niwi} set up in the way the simulated CRS is created according to Section E. The CRS for the blind signature scheme is $\Sigma = (\Lambda, \Sigma_{\mathsf{com}}, \Sigma_{\mathsf{sig}}, \Sigma_{\mathsf{niwi}})$. A signer runs $(vk, sk) \leftarrow \mathsf{SSIG.Key}(\Sigma_{\mathsf{sig}})$ where $vk = (A, B, \sigma^*)$ and publish vk as his verification key. The blind signature issuing protocols is as follows:

- On input $m \in \mathbb{Z}_p$, a user computes $(c, d) \leftarrow \mathsf{TC2.Com}(\Sigma_{\mathsf{com}}, m)$ where $(c, d) = (\tilde{g}^m f^\delta, g^\delta) \in \mathbb{G}_2 \times \mathbb{G}_1$. Then the user sends c to the signer.

- The signer computes $(z, r, s, t, u, v, w) \leftarrow \mathsf{SSIG.Sign}(sk, c)$ and sends σ to the user.

Scheme	#(rounds)	Communication	Signature Size	Security Model	Assumption
Oka06[42]	4	$3^{[N^2]} + 4^{[1]} + 10^{[p]}$	$4^{[1]} + 1^{[p]}$	SA	2SDH,DCR
KZ08[40]	6	$9^{[N^2]} + 7^{[1]} + 7^{[p]}$	$4^{[1]} + 1^{[p]}$	UC	2SDH,DCR
Fuc10[25]	2	$22^{[1]}$	$30^{[1]}$	SA	DAHSDH,HDL,DLIN
(ours)	2	$8^{[1]}$	$28^{[1]}$	UC	SFP, DLIN

Table 1: Summary of properties of concurrently secure efficient blind signatures. Columns for "Communication" and "Signature Size" count the number of elements, indicating the groups they belong to ($[N^2]$, $[1]$, and $[p]$, respectively, for $\mathbb{Z}_{N^2}$, $\mathbb{G}_1$, and $\mathbb{Z}_p$). UC: Universally Composable Security with Adaptive Corruption [21, 2]. SA: Stand-Alone Security. 2SDH: 2-Variable Strong Diffie-Hellman Assumption [42]. DCR: Decision Composite Residuosity [43]. DAHSDH,HDL: See [25]

- The user computes $(r', s', t', u', v', w') \leftarrow \mathsf{SigRand}(r, s, t, u, v, w)$ as in Section 4.4, and gives a GS-proof π with a witness (c, d, z, r', u') for pairing product equations

$$e(g, c)\, e(d, f^{-1}) = e(g, \tilde{g}^m), \tag{20}$$

$$e(g_z, z)\, e(g_r, r')\, e(g_1, c) = A \cdot e(s', t')^{-1}, \tag{21}$$

$$e(h_z, z)\, e(h_u, u')\, e(h_1, c) = B \cdot e(v', w')^{-1}. \tag{22}$$

  Then output a signature $\sigma = (s', t', v', w', \pi)$ for m.

Given (σ, m), a verifier accepts $\sigma = (s', t', v', w', \pi)$ if π is a correct GS-proof with respect to relations (20), (21), and (22).

In the construction, the use of $\mathsf{SigRand}$ is for better efficiency and does not affect to the framework due to the nature of perfect randomness. The resulting blind signature consists of 4 group elements, 5 GS commitments to group elements, and proof elements for 3 pairing product equations. Note that when $\Lambda = \Lambda_{\mathsf{sym}}$, we could swap the elements in the second pairing of the first equation and get all three equations to be one-sided pairing products. Thus, the size of final blind signature is 38 group elements for $\Lambda = \Lambda_{\mathsf{sxdh}}$ using GS-proof system with SXDH setting, and 28 group elements for $\Lambda = \Lambda_{\mathsf{sym}}$ using GS-proof system with DLIN setting. The communication complexity is quite low. Only 8 group elements are exchanged in total, and achieves optimal 2 moves. This could be a good efficiency standard for "crafted" constructions to compare.

By replacing SSIG with SIG from Section 4, one could also instantiate the very original Fischlin's scheme that is secure against static adversaries. This, however, requires NIZK proofs and hence becomes less efficient; NIZK requires that we replace A and B with their pairing product representations as originally described for SIG in Section 4. We also remark that the construction can be extended to a *partially-blind* scheme [1] as SSIG (and SIG) can sign multiple group elements at once.

Table 1 summaries efficiency of some known blind signature schemes. There are other schemes that achieve concurrent security without random oracles, e.g., [17, 39, 36, 40]. [42] is a representative from those without GS-proofs. Sizes for [42] vary in parameter setting and include some approximation. Numbers for [40] translates numbers in $\mathbb{Z}_{N^3}$ and $\mathbb{Z}_N$ into that of $\mathbb{Z}_{N^2}$ with appropriate factors . (Precisely, $9^{[N^2]}$ is a translation of $1^{[N^3]} + 6^{[N^2]} + 3^{[N]}$.) Our instantiation is very strong in communication while the schemes in [42, 40] with classical blind-then-unblind structure have an advantage in the signature size.

9.2 Group Signatures with Concurrent Join

This section highlights a useful property of our signature schemes that the message space is compatible with the verification key space. In particular, we present the most efficient instantiation of a group signature scheme that allows efficient concurrent join protocol [38].

In the symmetric setting $\Lambda = \Lambda_{\mathsf{sym}}$, the message space of USIG1 from Section 5 includes the verification key space. This allows Alice to sign Bob's key and Bob can sign Charlie's key and so on. Such a chaining can be hidden by applying NIZK. A signature scheme that allows to sign its own verification key is introduced as automorphic signatures in [25]. It is proven to have some interesting high-level applications such as proxy signatures.

Conceptually, a group signature scheme is a special case of such anonymous delegation system with only one hop of delegation. As sketched in [19] and embodied in [38], the above single-round certification protocol between Alice and Bob brings some favorable properties in the construction of efficient group signature schemes. In the following, we revisit the general idea of [19, 38, 32] with CPA-anonymity [9] by using terminology of proof of knowledge. The construction extends to CCA-anonymity by following the generic construction in [32]. Let SIG0 and SIG1 be sigantrue schemes, and NIWI be a witness indistinguishable proof of knowledge system. A group signature, GSIG, consists of 5 algorithms $\{\mathsf{Setup}, \mathsf{Join}, \mathsf{Sign}, \mathsf{Vrf}, \mathsf{Open}\}$ such that:

- GSIG.Setup is a setup algorithm that takes security parameter 1^λ and runs $(vk_{\mathrm{c}}, sk_{\mathrm{c}}) \leftarrow \mathsf{SIG0.Key}(1^\lambda)$ and also sets up a CRS Σ_{niwi} and a trapdoor sk_{o} for NIWI. Group verification-key is $vk_{\mathrm{g}} = (vk_{\mathrm{c}}, \Sigma_{\mathsf{niwi}})$. The certification-key sk_{c} is given privately to the issuer and the opening-key sk_{o} is given privately to the opener.

- GSIG.Join is an interactive protocol between a group member and the issuer. The group member generates his own key-pair $(vk_{\mathrm{u}}, sk_{\mathrm{u}}) \leftarrow \mathsf{SIG1.Key}(1^\lambda)$ and send vk_{u} to the issuer. The issuer signs on vk_{u} by $\sigma_{\mathrm{c}} \leftarrow \mathsf{SIG0.Sign}(sk_{\mathrm{c}}, vk_{\mathrm{u}})$ and send the certificate σ_{c} to the member.

- GSIG.Sign is a signing algorithm run by a group member to sign message m. It consists of signing on message m by $\sigma_{\mathrm{u}} \leftarrow \mathsf{SIG1.Sign}(sk_{\mathrm{u}}, m)$ and generating a non-interactive witness indistinguishable proof of knowledge $\pi \leftarrow \mathsf{NIWI.Prf}(\Sigma_{\mathsf{niwi}}, \mathrm{pub}, \mathrm{wit})$ that proves relation $1 = \mathsf{SIG0.Vrf}(vk_{\mathrm{c}}, vk_{\mathrm{u}}, \sigma_{\mathrm{c}})$ and $1 = \mathsf{SIG1.Vrf}(vk_{\mathrm{u}}, m, \sigma_{\mathrm{u}})$ with respect to witness $\mathrm{wit} = (vk_{\mathrm{u}}, \sigma_{\mathrm{c}}, \sigma_{\mathrm{u}})$ and public information $\mathrm{pub} = (vk_{\mathrm{c}}, m)$. Final output is π, which is a group signature.

- GSIG.Vrf takes $(vk_{\mathrm{g}}, m, \pi)$ as input and verifies correctness of π by verifying π as a NIWI proof with respect to $\mathrm{pub} = (vk_{\mathrm{c}}, m)$ and CRS Σ_{niwi}.

- GSIG.Open is an opening algorithm run by the opener who has opening-key sk_{o}. Given π and sk_{o} as input, the algorithm runs the knowledge extractor of the NIWI proof system and extracts witness $(vk_{\mathrm{u}}, \sigma_{\mathrm{c}}, \sigma_{\mathrm{u}})$. The exposed verification key vk_{u} identifies the group member who actually created π. This algorithm will be associated with another algorithm that publicly verifies the correctness of the opening.

Theorem 11. *Group signature scheme GSIG is CPA-anonymous, traceable, and non-frameable.*

We refer to [9] and [6] for formal definitions of the security notions stated in the theorem. Intuitively, CPA-anonymity is that the adversary cannot distinguish group signatures from two members. As CPA security, the adversary is not given oracle access to the opener. Traceability guarantees that once a group signature is opened, it identifies a group memeber who once completed GSIG.Join. Non-frameability means that no one but a group member can issue a valid group signature that points to the member if opened.

Scheme	Concurrent Join	Non-Frameability	Signature Size	Assumptions
BW07[14]	yes	no	$6^{[N]}$	SD, HSDH
Gro07[32]	no	yes	$28^{[1]}$	SDH, q-U, DLIN
GSIG([22]+BB[7])	yes	yes	$297^{[1]} + 1^{[p]}$	DLIN,SDH
GSIG(SIG+BB[7])	yes	yes	$43^{[1]} + 1^{[p]}$	SFP, SDH

Table 2: Summary of efficiency and properties of group signature schemes with CPA-anonymity. The signature size counts the number of elements and indicating the groups they belong to ([1], $[N]$, and $[p]$ respectively for $\mathbb{G}_1$, $\mathbb{Z}_N$, and $\mathbb{Z}_p$). SD: Subgroup Decision Assumption [11]. q-U: See [32].

Proof. CPA-anonymity follows directly from the (computational) WI property [35] of the proof system NIWI. For traceability, suppose that there is a valid signature π on message m. Due to the knowledge soundness of NIWI, the opener can extract $(vk_{\mathrm{u}}, \sigma_{\mathrm{c}}, \sigma_{\mathrm{u}})$ from π and $(vk_{\mathrm{u}}, \sigma_{\mathrm{c}})$ fulfills $1 = \mathsf{SIG0.Vrf}(vk_{\mathrm{c}}, vk_{\mathrm{u}}, \sigma_{\mathrm{c}})$. If vk_{u} does not point any group member registered through GSIG.Join, σ_{c} is a valid forgery for SIG0, which contradicts to the EUF-CMA property of SIG0. Thus vk_{u} allows tracing. For non-frameability, suppose that the opener extracts $(vk_{\mathrm{u}}, \sigma_{\mathrm{c}}, \sigma_{\mathrm{u}})$ from a group signature on message m. If $1 = \mathsf{SIG1.Vrf}(vk_{\mathrm{u}}, m, \sigma_{\mathrm{u}})$ holds but the owner of vk_{u} have never signed on m, it is a valid forgery against SIG1, contradicting the EUF-CMA property of SIG1. ∎

As mentioned in [38], the above framework has been known without efficient instantiation in the standard model. Using our main signature scheme SIG as SIG0 and GS-proof system as NIWI, we can instantiate the construction with efficiency. We assess the efficiency in the setting $\Lambda = \Lambda_{\mathsf{sym}}$ as follows. Let SIG1 be a signature scheme whose verification key vk_{u} and signature σ_{u} consist of α and β group elements, respectively. Let γ be the number of group elements needed to prove relation $1 = \mathsf{SIG1.Vrf}(vk_{\mathrm{u}}, m, \sigma_{\mathrm{u}})$ including GS-commitments for vk_{u} and σ_{u}. Regardless of size vk_{u} to be certified, our SIG0 outputs σ_{c} of size 7. Since 4 out of the 7 elements in σ_{c} can be perfectly randomized and given in the clear as we have done in Section 9.1, we need only 3 GS-commitments in proving relation $1 = \mathsf{SIG0.Vrf}(vk_{\mathrm{c}}, vk_{\mathrm{u}}, \sigma_{\mathrm{c}})$, which consists of two one-sided pairing product equations and costs 6 elements in a proof. (Commitments of vk_{u} is already included in γ.) In total we have (Group Sig Size) $= 19 + \gamma$. One can instantiate SIG0 with the signature scheme in [22], that has $9\alpha + 4$ elements in σ_{c} and $3\alpha + 3$ one-sided and 3α double-sided pairing product equations in SIG0.Vrf. In that case, the size of a group signature is (Group Sig Size) $= 63\alpha + 21 + \gamma$.

If we instantiate SIG1 with full EUF-CMA Boneh-Boyen signature scheme from [7], vk_{u} consists of $\alpha = 4$ group elements (including the bases). A signature consists of one group element and one scalar value but the scalar value is totally random and independent of the verification-key. So we have $4+1$ GS-commitments in proving $1 = \mathsf{SIG1.Vrf}(vk_{\mathrm{u}}, m, \sigma_{\mathrm{u}})$. The verification predicate consists of a double-sided pairing product equation, which yields 9 group elements in a proof. In total, we have $\gamma = 24$ and a group signature consists of 43 group elements and 1 scalar value. With [22] for SIG0, the signature size will be 297 group elements and 1 scalar value.

Table 2 summarizes some efficient group signature schemes that provide CPA-anonymity in the standard model. [13, 14] provide efficiency with more reasonable assumptions. However, they are not non-frameable, i.e., the issuer can frame any group member. [38] allows concurrent join but the security is argued in the random oracle model [5]. The scheme in [32] yields a group signature of 28 group elements in the case of CPA anonymity. Their GSIG.Join protocol includes 6 rounds of interaction and does not provide concurrent security. Also, the traceability needs to put a strong dedicated assumption on top of the security of the building blocks. The increased signature size in

our construction is the price for achieving concurrent join property and allowing very simple and modular security argument without dedicated assumptions.

Some final remarks follow: CCA-anonymity is obtained by following the approach in [32], which uses a strong one-time signature scheme and a selective-tag CCA secure tag-based public-key encryption scheme. As stated in [32], this strengthening costs extra 15 group elements in a signature. Accordingly, we have a CCA-anonymous group signature scheme with concurrent join whose signature consits of 58 group elements and one scalar value. One of the advantages of using our SIG for SIG0 is that it allows to insert a warranty in the clear to σ_c so that the signing policy given to a group member is explicit. Due to the constant-size property of SIG, this useful extension can be done without impacting to the size of the group signature (except for the warranty itself) at all.

10 Conclusion

This paper presented a practical signature scheme all components of which are group elements in bilinear settings. Signing arbitrary k group elements at the same time results in a signature of size only 7 group elements. This solves an open problem since [31] (explicitly stated in [22]). Its technically interesting properties are enlightened by presenting variations with advanced properties. Combined with Groth-Sahai proof system, our signature schemes give handy and reasonably practical solutions to many cryptographic tasks. As an example, we give an answer to the open problem of instantiating the round-optimal blind signature framework from [24, 2].

The most challenging open problem is to base security on weaker and well studied assumptions while retaining the efficiency and compatibility. Also, it would be interesting to explore generic domain extension methods since this is rather a new issue that has not been needed in ordinary signature schemes that can compress arbitrarily long messages with a hash function. Finally, it is left as an open problem to construct a compatible commitment scheme with constant-size commitments in the base groups.

References

[1] M. Abe and E. Fujisaki. How to date blind signatures. In K. Kim and T. Matsumoto, editors, *Advances in Cryptology – ASIACRYPT '96*, volume 1163 of *LNCS*, pages 244–251. Springer-Verlag, 1996. (Cited on page 23.)

[2] M. Abe and M. Ohkubo. A framework for universally composable non-committing blind signatures. In M. Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 435–450. Springer-Verlag, 2009. (Cited on page 1, 2, 13, 22, 23, 26.)

[3] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. Non-interactive anonymous credentials. In R. Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008*, volume 4948 of *LNCS*. Springer-Verlag, 2008. Also available on IACR ePrint Archive, 2007/384. (Cited on page 1.)

[4] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements and a construction based on general assumptions. In E. Biham, editor, *Advances in Cryptology - EUROCRPYT 2003*, volume 2656 of *LNCS*, pages 614–629. Springer-Verlag, 2003. (Cited on page 1.)

[5] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communication Security*, pages 62–73. Association for Computing Machinery, 1993. (Cited on page 25.)

[6] M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In A. Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–154. Springer-Verlag, 2005. Full version available at IACR e-print 2004/077. (Cited on page 1, 24.)

[7] D. Boneh and X. Boyen. Short signatures without random oracles. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology — Eurocrypt '04*, volume 3027 of *LNCS*, pages 56–73. Springer-Verlag, 2004. (Cited on page 25.)

[8] D. Boneh and X. Boyen. Short signatures without random oracles and the sdh assumption in bilinear groups. *Journal of Cryptology*, 21(2):149–177, 2008. (Cited on page 1, 2.)

[9] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *Advances in Cryptology — CRYPTO '04*, volume 3152 of *LNCS*, pages 41–55. Springer-Verlag, 2004. (Cited on page 1, 3, 24.)

[10] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432. Springer-Verlag, 2003. (Cited on page 1.)

[11] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-DNF formulas on ciphertexts. In J. Kilian, editor, *Theory of Cryptography Conference– TCC'2005*, volume 3378 of *LNCS*, pages 325–341. Springer-Verlag, 2005. (Cited on page 25.)

[12] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In C. Boyd, editor, *Advances in Cryptology – Asiacrypt 2001*, volume 2248 of *LNCS*, pages 514–532. Springer-Verlag, 2001. (Cited on page 36.)

[13] X. Boyen and B. Waters. Compact group signatures without random oracles. In *Advances in Cryptology — Eurocrypt '06*, volume 4004 of *LNCS*, pages 427–444. Springer-Verlag, 2006. Full version available from IACR ePrint Archive 2005/381. (Cited on page 25.)

[14] X. Boyen and B. Waters. Full-domain subgroup hiding and constant-size group signatures. In *Public Key Cryptography—PKC 2007*, volume 4450 of *LNCS*, pages 1–15. Springer-Verlag, 2007. Available at `http://www.cs.stanford.edu/~xb/pkc07/`. (Cited on page 25.)

[15] J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In *Advances in Cryptology - EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 351–368. Springer-Verlag, 2009. (Cited on page 1, 43.)

[16] J. Camenisch, M. Kohlweiss, and C. Soriente. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In *Public Key Cryptography, PKC 2009*, volume 5443 of *LNCS*, pages 481–500. Springer-Verlag, 2009. (Cited on page 1.)

[17] J. Camenisch, M. Koprowski, and B. Warinschi. Efficient blind signatures without random oracles. In C. Blundo and S. Cimato, editors, *Security in Communication Networks, 4th International Conference, SCN 2004, Amalfi, Italy, September 8-10, 2004, Revised Selected Papers*, volume 3352 of *LNCS*, pages 134–148. Springer-Verlag, 2005. (Cited on page 23.)

[18] J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Advances in Cryptology — CRYPTO '04*, volume 3152 of *LNCS*, pages 56–72. Springer-Verlag, 2004. (Cited on page 1.)

[19] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In B. S. Kaliski Jr., editor, *Advances in Cryptology — CRYPTO '97*, volume 1294 of *LNCS*, pages 410–424. Springer-Verlag, 1997. (Cited on page 24.)

[20] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the* 42*nd IEEE Annual Symposium on Foundations of Computer Science*, pages 136–145, 2001. (Cited on page 22.)

[21] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. Technical Report 2000/067, IACR e-print Archive, 2005. 2nd version updated on 13 Dec 2005. (Cited on page 23.)

[22] J. Cathalo, B. Libert, and M. Yung. Group encryption: Non-interactive realization in the standard model. In M. Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 179–196. Springer-Verlag, 2009. (Cited on page 1, 4, 25, 26, 34, 41, 42.)

[23] J. H. Cheon. Security analysis of hte strong diffie-hellman problem. In *Advances in Cryptology — Eurocrypt '06*, volume 4004 of *LNCS*, pages 1–11. Springer-Verlag, 2006. (Cited on page 2.)

[24] M. Fischlin. Round-optimal composable blind signatures in the common reference model. In C. Dwork, editor, *Advances in Cryptology — CRYPTO '06*, volume 4117 of *LNCS*, pages 60–77. Springer-Verlag, 2006. (Cited on page 1, 2, 13, 22, 26.)

[25] G. Fuchsbauer. Automorphic signatures in bilinear groups. IACR ePrint Archive 2009/320, updated Feb.10, 2010, 2009-10. (Cited on page 1, 2, 23, 24, 40.)

[26] G. Fuchsbauer, D. Pointcheval, and D. Vergnaud. Transferable anonymous constant-size fair e-cash. IACR ePrint Archive 2009/146. Also to appear in CANS 2009., 2009. (Cited on page 1.)

[27] S. Galbraith, K. Paterson, and N. Smart. Pairings for cryptographers. Technical Report 2006/165, IACR ePrint archive, 2006. (Cited on page 3.)

[28] S. D. Galbraith and V. Rotger. Easy decision-diffie-hellman groups. *LMS Journal of Computation and Mathematics*, 7:2004, 2004. (Cited on page 3.)

[29] S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988. (Cited on page 3.)

[30] M. Green and S. Hohenberger. Universally composable adaptive oblivious transfer. In J. Pieprzyk, editor, *Advances in Cryptology - ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 179–197. Springer-Verlag, 2008. Preliminary version: IACR ePrint Archive 2008/163. (Cited on page 1.)

[31] J. Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *Advances in Cryptology - ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer-Verlag, 2006. (Cited on page 1, 26.)

[32] J. Groth. Fully anonymous group signatures without random oracles. In *Advances in Cryptology – Asiacrypt'07*, volume 4833 of *LNCS*, pages 164–180. Springer-Verlag, 2007. (Cited on page 24, 25, 26.)

[33] J. Groth. Homomorphic trapdoor commitments to group elements. Cryptology ePrint Archive, Report 2009/007, January 2009. (Cited on page 4, 34, 35.)

[34] J. Groth. Homomorphic trapdoor commitments to group elements. Unpublished Manuscript, 2010. (Cited on page 4.)

[35] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Advances in Cryptology — Eurocrypt '08*, volume 4965 of *LNCS*, pages 415–432. Springer-Verlag, 2008. Full version available: IACR ePrint Archive 2007/155. (Cited on page 1, 5, 25, 42, 43, 44.)

[36] C. Hazay, J. Katz, C. Koo, and Y. Lindell. Concurrently-secure blind signatures without random oracles or setup assumptions. In *Theory of Cryptography Conference, TCC 2007*, volume 4392 of *LNCS*, pages 323–341. Springer-Verlag, 2007. (Cited on page 13, 22, 23.)

[37] A. Kiayias and M. Yung. Group signatures with efficient concurrent join. In *Advances in Cryptology – Eurocrypt 2005*, volume 3494 of *LNCS*, pages 198–214. Springer-Verlag, 2005. (Cited on page 1.)

[38] A. Kiayias and M. Yung. Group signatures with efficient concurrent join. In R. Cramer, editor, *Advances in Cryptology — EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 198–214. Springer-Verlag, 2005. (Cited on page 24, 25.)

[39] A. Kiayias and H. Zhou. Concurrent blind signatures without random oracles. In *SCN 2006*, volume 4116 of *LNCS*, pages 49–62. Springer-Verlag, 2006. (Cited on page 23.)

[40] A. Kiayias and H. Zhou. Equivocal blind signatures and adaptive uc-security. In R. Canetti, editor, *Theory of Cryptography Conference, TCC 2008*, volume 4948 of *LNCS*, pages 340–355. Springer-Verlag, 2008. (Cited on page 2, 23.)

[41] N. Koblitz and A. Menezes. Another look at generic group. IACR ePrint Archive 2006/230, 2006. (Cited on page 2.)

[42] T. Okamoto. Efficient blind and partially blind signatures without random oracles. In S. Halevi and T. Rabin, editors, *Theory of Cryptography Conference, TCC 2006*, volume 3876 of *LNCS*, pages 80–99. Springer-Verlag, 2006. Full version avaialble on ePrint archive. (Cited on page 23.)

[43] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *LNCS*, pages 223–238. Springer-Verlag, 1999. (Cited on page 23.)

[44] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, volume 576 of *LNCS*, pages 129–140. Springer-Verlag, 1992. (Cited on page 22.)

[45] M. Rückert and D. Schröder. Security of verifiably encrypted signatures and a construction without random oracles. IACR ePrint Archive 2009/027, 2009. (Cited on page 1.)

[46] A. Rupp, G. Leander, E. Bangerter, A.-R. Sadeghi, and A. W. Dent. Sufficient conditions for intractability over black-box groups: Generic lower bounds for generalizaed dl and dh problems. IACR ePrint Archive 2007/360, 2007. (Cited on page 31.)

[47] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4), 1980. (Cited on page 31.)

[48] M. Scott. Authenticated id-based key exchange and remote log-in with simple token and pin number. Cryptology ePrint Archive, Report 2002/164, 2002. `http://eprint.iacr.org/`. (Cited on page 3.)

[49] V. Shoup. Lower bounds for discrete logarithms and related problems. In W. Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *LNCS*, pages 256–266. Springer-Verlag, 1997. (Cited on page 2, 31.)

[50] E. R. Verheul. Evidence that xtr is more secure than supersingular elliptic curve cryptosystems. *J. Cryptology*, 17(4):277–296, 2004. (Cited on page 3.)

Appendices

A Proofs Related to Assumptions

A.1 Proof of Theorem 1 ($\mathbf{DDH}_{\mathbb{G}_1} \Rightarrow \mathbf{DBP}$)

Proof. Assume that the DBP assumption does not hold and there is an adversary $\mathcal{A}$ that produces a pair $(z, r) \neq (1, 1)$ satisfying the equation $e(g_z, z)\ e(g_r, r) = 1$ for randomly chosen g_z, g_r with non-negligible probability. We construct $\mathcal{B}$ which breaks the DDH assumption in $\mathbb{G}_1$.

The $\mathrm{DDH}_{\mathbb{G}_1}$ assumption says that given a tuple $e(g, g_a, g_b, g_c) \in \mathbb{G}_1^4$, where $g_a = g^a$, $g_b = g^b$, and $g_c = g^c$, for $a, b, c \in \mathbb{Z}_p^*$ it is hard to distinguish distinguish between $c = ab$ and $c \neq ab$ with non-negligible probability. For a challenge tuple (g, g_a, g_b, g_c), $\mathcal{B}$ chooses a random $\psi \in \mathbb{Z}_p^*$ and gives to $\mathcal{A}$ an input (g^ψ, g_a^ψ) along with the appropriate public parameters. If $\mathcal{A}$ outputs $(z, r) \neq (1, 1)$ satisfying $e(g^\psi, z)\ e(g_a^\psi, r) = 1$, it is true that $z = r^{-a}$. Then, $e(g_b, z)\ e(g_c, r) = e(g^b, r^{-a})\ e(g^c, r) = e(g, r)^{c-ab}$; that equation is equal to 1 if and only $ab = c \mod p$.

Therefore, $\mathcal{B}$ has the same success probability of breaking the $\mathrm{DDH}_{\mathbb{G}_1}$ assumption as $\mathcal{A}$ of breaking the DBP assumption. ∎

A.2 Proof of Theorem 3 (SFP $\Rightarrow$ SDP)

Proof. Suppose that there exists an algorithm, $\mathcal{A}$, that successfully finds (z, r, u) that fulfills (2). We construct an algorithm that breaks SFP as follows. Given an SFP instance $(\Lambda, g_z, h_z, g_r, h_u, a, \tilde{a}, b, \tilde{b}, R_1, \ldots, R_q)$, input (g_z, h_z, g_r, h_u) to $\mathcal{A}$. If $\mathcal{A}$ outputs $(z^\star, r^\star, u^\star)$, set $(s^\star, t^\star, v^\star, w^\star) = (a, \tilde{a}, b, \tilde{b})$ and output $R^\star = (z^\star, r^\star, s^\star, t^\star, u^\star, v^\star, w^\star)$.

Now, multiplying $1 = e(g_z, z^\star), e(g_r, r^\star)$ to both sides of $(a, \tilde{a}) = e(s^\star, t^\star)$ results in the first equation in (3). Similarly, multiplying $1 = e(h_z, z^\star), e(h_u, u^\star)$ to both sides of $(b, \tilde{b}) = e(v^\star, w^\star)$ results in the second equation in (3). Thus $R^\star$ fulfills relations in (3). Since $(z^\star, r^\star, u^\star)$ is a valid answer to SDP, $z^\star \neq 1$ holds. Since every z_j in R_j is uniformly chosen and independent of $(g_z, h_z, g_r, h_u, a, \tilde{a}, b, \tilde{b})$, it is independent of the view of the adversary. Thus $z^\star = z_j$ happens only with negligible probability for every $j \in \{1, \ldots, q\}$. Thus $R^\star$ is a correct and valid answer to the SFP instance. ∎

A.3 Proof of Theorem 2 and Theorem 6 (Justification of SFP and k-SFP)

Proof.

Outline. In the generic model, every group element is represented by a unique index. The group operation to two group elements corresponds to addition of two indices. In the simulation, the index for an independent group element is represented by a unique variable. The index for a group element that is related to independent group elements is represented by a function of the variables determined by the relation.

A security argument in the generic group model consists of three steps. First we argue that no linear combinations of indices of initially given group elements could yield a new set of indices that fulfills the target predicate implied by the assumption. This step is done by inspecting the form of possible representation of indices. Although the argument looks intricate and lengthy, the underlying idea follows the standard approach.

The second step is to estimate the success probability of the adversary when uniform assignment is done to the variables. The adversary is considered as successful either when the simulation happens to be inconsistent to the concrete assignment or when the output of the adversary happens to fulfill the target predicate. For this, we estimate the probability that two indices represented by functions of variables are not identical but fall into the same value when concrete values are assigned to the variables. A common idea for this step is to apply Schwartz's lemma [49, 47] to the formula representing the difference between two indices. When the formula is a polynomial, it promptly gives an upper bound to the probability due to the degree of the polynomial. In the case of SDH, however, the formula will be in the form of $\frac{1}{x-c_1} + \cdots + \frac{1}{x-c_q}$ that results in having x^q in the polynomial expression of the formula. Thus it gives an upper bound with factor of q. By accumulating the error probability for all combinations of indices that could be generated through ℓ group operations, total error probability is bound by $\mathcal{O}(q\ell^2)$, which is apart from the optimal $\mathcal{O}(\ell^2)$ bound in DL and CDH [49] by the factor of q. In the case of SFP, the formula is in the form of $u_1 + \frac{1}{u_1} + \cdots + u_q + \frac{1}{u_q}$, which is a Laurent polynomial. Directly applying Schwartz's lemma as introduced in [49] or its variation for Laurent polynomial in [46] results in loose bound with factor of q since the formula is in degree q in its (regular) polynomial form. We instead follow the *proof idea* of the original Schwartz's lemma as introduced in [47] and show a constant bound with factor of 2. In the actual argument, we have to consider products of two indices to take care of pairing operations, which makes the analysis more intricate. But we can show a constant bound in the same way. As a result, the factor of q is eliminated and we have optimal $\mathcal{O}(\ell^2)$ bound.

Details. SFP is a special case of k-SFP at $k = 1$. For simplicity, we start with the case of SFP and then show how to generalize the argument to the case of k-SFP. We consider the generic group model for symmetric group setting where $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$. The symmetric setting is not just for simplicity but also for generality as our argument in the symmetric setting trivially holds in the asymmetric setting as well. (The error probability can be slightly improved in the asymmetric setting since each group has less elements.)

In the following, we focus on the relation between the indexes of the group elements by translating the assumption appropriately. To make the argument easily linkable to its original representation, we use the same letter to denote the index of a group element with respect to an implicit fixed base. For instance, for $a \in \mathbb{G}$, we denote $\log a \in \mathbb{Z}_p$ by a itself. By using this notation, SFP is translated as follows. Let I be a tuple that consists of $(g_z, h_z, g_r, h_u, a, \tilde{a}, b, \tilde{b})$ and $R_j = (z_j, r_j, s_j, t_j, u_j, v_j, w_j)$ for $j = 1, \ldots, q$, which intotal consits of $8 + 7q$ variables. Recall R_j

fulfills relations

$$A \stackrel{\text{def}}{=} a \cdot \tilde{a} = g_z \cdot z_j + g_r \cdot r_j + s_j \cdot t_j, \quad \text{and} \tag{23}$$

$$B \stackrel{\text{def}}{=} b \cdot \tilde{b} = h_z \cdot z_j + h_u \cdot u_j + v_j \cdot w_j. \tag{24}$$

Let $O = (z^\star, r^\star, s^\star, t^\star, u^\star, v^\star, w^\star)$ be a 7-tuple of linear combinations of the variables in I. We show that there is no O that identically fulfills relations

$$A = g_z \cdot z^\star + g_r \cdot r^\star + s^\star \cdot t^\star, \quad \text{and} \tag{25}$$

$$B = h_z \cdot z^\star + h_u \cdot u^\star + v^\star \cdot w^\star \tag{26}$$

under the constraint that $z^\star \notin \{0, z_1, \ldots, z_q\}$.

For a polynomial $x \in \mathbb{Z}_p[I]$ of degree 1, let $\psi(x)$ be the linear form (line matrix of $1 \times |I|$) associated to x. Then, for $x, y \in \mathbb{Z}_p[I]$, product $x \cdot y$ is represented by a $|I| \times |I|$ symmetric matrix, say $\Psi(x \cdot y)$,

$$\Psi(x \cdot y) = \frac{1}{2}\left(\psi(x) \circ \psi(y)^T + \psi(y) \circ \psi(x)^T\right). \tag{27}$$

It is important to see that $\mathrm{Rank}(\Psi(x \cdot y)) = 2$. Each row and column is associated to a variable, e.g., a in I and called a-row and a-column. By $\Psi(X+Y)$ we denote a matrix obtained by $\Psi(X)+\Psi(Y)$. In the following, we re-write $a, \tilde{a}, b, \tilde{b}$ by s_0, t_0, v_0, w_0, respectively, and let $z_0 = r_0 = u_0 = 0$ for seamless argument. Define A_j and B_j as $A_j = \Psi(g_z \cdot z_j + g_r \cdot r_j + s_j \cdot t_j)$ and $B_j = \Psi(h_z \cdot z_j + h_u \cdot r_j + v_j \cdot w_j)$ for $j = 0, \ldots, q$.

Relation (25) holds if and only if $\Psi(g_z \cdot z^\star + g_r \cdot r^\star + s^\star \cdot t^\star)$ is identical to a quadratic form representation of A. Since $\mathrm{Rank}(\Psi(g_z \cdot z^\star + g_r \cdot r^\star + s^\star \cdot t^\star)) \leq 6$, it suffices to consider a quadratic form representation of A whose rank is less than 6. Recall that A is defined by A_0 and equivalently by A_j for $j = 1, \ldots, q$. For some symmetric matrix X of size $|I| \times |I|$, Consider $R = (((X \bmod A_0) \bmod A_1) \ldots)$. If $R \neq \emptyset$, there exists an assignment to the variables that evaluates the quadratic form associated by R to a non-zero value. Accordingly, for a matrix X to be evaluated to A for any valid assignments to the variables, X must be identical to $\sum c_j A_j$ where $\sum c_j = 1$ for some constant $c_j \in \mathbb{Z}_p$. Observe that $\mathrm{Rank}(A_j) = 6$ for $j = 1, \ldots, q$. Also observe that for any $j = 0, \ldots, q$, and for any $j' \neq j$, it holds that $\mathrm{Rank}(c_j A_j + c_{j'} A_{j'}) \geq 8$ when $c_i \neq 0$ and $c_{i'} \neq 0$. Thus only individual A_j for $j = 0, \ldots, q$ are the quadratic form representations of A with rank equal to or less than 6. It therefore suffices to consider A_j for the left hand of (25). By the same argument, it suffices to consider B_j for the left hand of (26).

Observe that $\mathrm{Rank}(A_j) = 6$ for $j = 1, \ldots, q$. Also observe that for any $j = 0, \ldots, q$, and for any $j' \neq j$, s_j-row and t_j-row in A_j are linearly independent from $A_{j'}$ and these rows are zeros in $A_{j'}$. Accordingly, $\mathrm{Rank}(c_j A_j + c_{j'} A_{j'}) \geq 8$ if $c_i \neq 0$ and $c_{i'} \neq 0$. Thus only individual A_j for $j = 0, \ldots, q$ are quadratic form representation of A with rank equal to or less than 6. It therefore suffices to consider A_j for the left hand of (25). The same argument applies to (26).

Suppose that (25) holds, namely:

$$\Psi(g_z \cdot z_j) + \Psi(g_r \cdot r_j) + \Psi(s_j \cdot t_j) = \Psi(g_z \cdot z^\star) + \Psi(g_r \cdot r^\star) + \Psi(s^\star \cdot t^\star) \tag{28}$$

holds. Observe that all cells in matrix $\Psi(g_z \cdot z^\star)$ are zeros except for those in the g_z-row and g_z-column. Thus $\Psi(s_j \cdot t_j)$ is not covered by $\Psi(g_z \cdot z^\star)$. Similarly, $\Psi(s_j \cdot t_j)$ is not covered by $\Psi(g_r \cdot r^\star)$, either. Thus $\Psi(s^\star \cdot t^\star)$ covers $\Psi(s_j \cdot t_j)$. Observe that $\mathrm{Rank}(\Psi(s^\star \cdot t^\star)) = 2$ and the s_j-row and t_j-row in $\Psi(s^\star \cdot t^\star)$ are linearly independent each other. Since $z^\star \neq z_j$, we have either $z^\star = c \cdot z_j$ for some constant $c \neq 1$ or $z^\star$ is a linear combination of variables including at least one variable other

than z_j. In either case, the g_z-row of $\Psi(g_z \cdot z^\star)$ is linearly dependent on the s_j-row and t_j-row in $\Psi(s^\star \cdot t^\star)$. This results in having non-zero terms of s_j or t_j or both in $z^\star$. Without loss of generality, assume that $z^\star$ includes a term in s_j. Now consider the other relation

$$\Psi(h_z \cdot z_{j'}) + \Psi(h_u \cdot r_{j'}) + \Psi(v_{j'} \cdot w_{j'}) = \Psi(h_z \cdot z^\star) + \Psi(h_u \cdot r^\star) + \Psi(v^\star \cdot w^\star) \tag{29}$$

where j' can be different from j in (28). Since $z^\star$ includes s_j, cell (h_z, s_j) is non-zero in $\Psi(h_z \cdot z^\star)$. Since the cell is zero in the matrices in the left hand of (29), it must be offset by $\Psi(h_u \cdot r^\star) + \Psi(v^\star \cdot w^\star)$. In $\Psi(h_u \cdot r^\star)$, the cell is zero since only h_u-row and h_u-column can have non-zero cells. Therefore cell (h_z, s_j) is non-zero in $\Psi(v^\star \cdot w^\star)$. Due to the same reason as before, $\Psi(v^\star \cdot w^\star)$ must cover $\Psi(v_{j'} \cdot w_{j'})$ in (29). Since $\mathrm{Rank}(\Psi(v_{j'} \cdot w_{j'}))$ is two, the h_z-row must be linearly dependent on $v_{j'}$-row and $w_{j'}$-row. Thus either $(v_{j'}, s_j)$ or $(w_{j'}, s_j)$ must be non-zero. However, none of the matrices in (29) has non-zero value in $(v_{j'}, s_j)$ and $(w_{j'}, s_j)$. Thus (29) cannot hold. This completes the case of $k = 1$.

To generalize the above argument to the case of $k \geq 2$, simply replace $\Psi(s_j \cdot t_j)$ with $\Psi(\sum_{i=1}^{k} s_{ji} \cdot t_{ji})$ and argue in the same way based on the observation that $\mathrm{Rank}(\Psi(\sum_{i=1}^{k} s_{ji} \cdot t_{ji})) = 2k$.

We now proceed to evaluate the error probability of the generic group oracle simulation for the case of general k. Namely, we consider the probability, say P_1, that two distinct elements in $\mathbb{G}$ evaluates to the same value by assignment. An index for an element in $\mathbb{G}$ is a linear combination of variables in I. For index f and f' of two distinct elements in $\mathbb{G}_T$, probability P_1 is $\Pr[f - f' = 0]$. Here the probability is taken over uniform assignments to the independent variables in I. Among the variables in I we consider r_j and u_j be dependent and represent them by

$$r_j = \left(\sum_{i=1}^{k} (a_i \cdot \tilde{a}_i - s_{ji} \cdot t_{ji}) - g_z \cdot z_j \right) / g_r, \text{ and} \tag{30}$$

$$u_j = \left(\sum_{i=1}^{k} \left(b_i \cdot \tilde{b}_i - v_{ji} \cdot w_{ji} \right) - h_z \cdot z_j \right) / h_u. \tag{31}$$

Let F be a polynomial obtained by replacing r_j and u_j in f with the right hands of the above equations and multiplying $g_r \cdot h_u$. Define F' for f' in the same way. Then we have $P_1 = \Pr[F - F' = 0]$. Since $\deg(F - F') = 3$, we have $P_1 \leq 3/p$ from Schwartz's lemma. Having initial $|I| = 2 + 4k + (3 + 4k)q$ elements and at most ℓ_1 group operations, we have the upper bound

$$\binom{2 + 4k + (3 + 4k)q + \ell_1}{2} \cdot 3/p \leq O((k \cdot q + \ell_1)^2)/p \tag{32}$$

for the simulation error of elements in $\mathbb{G}$.

The error probability, say P_T, in simulating $\mathbb{G}_T$ is estimated in the similar way. An index of an element in $\mathbb{G}_T$ is in a quadratic form of variables in I. Let F_T and F'_T be polynomials obtained from the indexes of two distinct elements in $\mathbb{G}_T$ in the same way as above. Then $P_T = \Pr[F_T - F'_T = 0]$. Since $\deg(F_T - F'_T) = 4$, we have $P_T \leq 4/p$ from Schwartz's lemma. Having at most ℓ_T pairing operations and group operations in $\mathbb{G}_T$, at most ℓ_T elements appear in $\mathbb{G}_T$ during the simulation. Thus the error probability throughout the simulation is bound by $\binom{\ell_T}{2} \cdot P_T = \mathcal{O}(\ell_T^2)/p$.

In total, the simulation error is upper bound by $P_1 + P_T = \mathcal{O}(\ell_T^2 + (k \cdot q + \ell_1)^2)/p$. By setting $\ell = \ell_1 \approx \ell_T$, it is simplified to $\mathcal{O}(k^2 \cdot q^2 + \ell^2)/p$ as stated in Theorem 6. Setting $k = 1$ gives $\mathcal{O}(q^2 + \ell^2)/p$ as in Theorem 2. ▌

Scheme	Λ	$\mathcal{K}$	$\mathcal{M}$	$\mathcal{C}$	$\mathcal{D}$	#(pairings)	#(PPE)	assumption
TC1	any	$2k+2^{[1]}$	$k^{[2]}$	$2^{[T]}$	$2^{[2]}$	$2k+2$	2	SDP $\ll$ DLIN
TC4	$\Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}$	$k+1^{[1]}$	$k^{[2]}$	$1^{[T]}$	$1^{[2]}$	$k+1$	1	DBP $\ll$ $\mathrm{DDH}_{\mathbb{G}_1}$
Gro09[33]	any	$2k+4^{[1]}$	$k^{[2]}$	$2^{[T]}$	$2^{[2]}$	$2k+4$	2	STP $\ll$ DLIN
TC3	Λ_{sym}	$2k+2^{[1]}$	$k^{[1]}$	$2k+2^{[1]}$	$2^{[1]}$	2k+ 2	2	SDP $\ll$ DLIN
CLY09[22]	Λ_{sym}	$5^{[1]}$	$1^{[1]}$	$3^{[1]}$	$3^{[1]}$	9	3	DLIN
TC2	any	$2^{[2]}$	$1^{[p]}$	$1^{[2]}$	$1^{[1]}$	2	1	XDHI ($\ll$ DLIN)

Table 3: Summary of homomorphic trapdoor commitments. Columns from $\mathcal{K}$ to $\mathcal{D}$ count the number of elements and indicating the groups they belong to ([1], [2], [T], and [p] respectively for $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, and $\mathbb{Z}_p$). #(pairings) and #(PPE) count the number of pairings and pairing product equations in the verification predicate. On top are the multi-message schemes committing to k group elements at once; in the middle are the schemes *not* using any group element in $\mathbb{G}_T$; and at the bottom is the efficient scheme when the message is in $\mathbb{Z}_p$ and the other components are in $\mathbb{G}_1$ and $\mathbb{G}_2$. $X \ll Y$: Assumption X is implied by Y (if $\Lambda = \Lambda_{\mathsf{sym}}$).

B Homomorphic Trapdoor Commitment Schemes

This section presents several homomorphic trapdoor commitment schemes in bilinear settings. Not all of them are used in our construction. Nevertheless, we introduce them because they have different properties and may be useful in applications needing specific properties. We note that all the schemes in this section can also work as a chameleon hash. Namely, it is possible to equivocate any commitment generated by TC.Com rather than the ones simulated by TC.Sim. Indeed, we integrate TC1 as a chameleon hash in the construction of SSIG in Section 6.

By $(\mathcal{K}, \mathcal{M}, \mathcal{C}, \mathcal{D})$ we denote spaces for commitment-keys, messages, commitments, and decommitments. The commitment-key refers to elements not included in Λ. Table 3 shows a summary of the schemes in their space parameters and performance in verifying the correct opening. For comparison, we list schemes from [33] and [22] which are the only homomorphic trapdoor commitment schemes we aware in the literature whose messages are group elements and the verification is done by checking pairing product equations.

TC3 and [22] are GS-compatible schemes whose components are all in the base groups. In particular, TC3 is the first multi-commitment scheme that commits to k elements at a time. Its commitment has $2k+2$ group elements while it will be $3k$ if we repeatedly use [22] for k times. It is an interesting open problem to construct a constant-size commitment scheme while being compatible with GS-proofs.

For multi-message commitment schemes, TC1, TC3, TC4, let $\vec{m} = \{m_1, \ldots, m_k\} \in {\mathbb{G}_2}^k$ be a message. For single-message commitment scheme, TC2, let m be an element of $\mathbb{Z}_p$. In the following description, we assume that Λ is given to all algorithms implicitly.

B.1 Scheme TC1

$\mathsf{TC1.Key}(1^\lambda)$: Choose random generators g_r, h_u from $\mathbb{G}_1^*$. For $i = 1, \ldots, k$, choose γ_i and δ_i from $\mathbb{Z}_p^*$ and compute $g_i = g_r^{\gamma_i}$ and $h_i = h_u^{\delta_i}$. Output commitment-key $ck = (g_r, h_u, g_1, h_1, \ldots, g_k, h_k)$ and trapdoor $tk = (\gamma_1, \delta_1 \ldots, \gamma_k, \delta_k)$.

$\mathsf{TC1.Com}(ck, \vec{m})$: Choose r and u randomly from $\mathbb{G}_2$, and compute

$$C_1 = e(g_r, r) \prod_{i=1}^{k} e(g_i, m_i) \quad \text{and} \quad C_2 = e(h_u, u) \prod_{i=1}^{k} e(h_i, m_i). \tag{33}$$

Output commitment $c = (C_1, C_2)$ and decommit-key $dk = (r, u)$.

$\mathsf{TC1.Vrf}(ck, c, \vec{m}, dk)$: Parse c into (C_1, C_2) and dk into (r, u). Output 1 if (33) holds. Output 0, otherwise.

$\mathsf{TC1.Sim}(ck)$: Choose r and u randomly from $\mathbb{G}_2$ and compute $C_1 = e(g_r, r)$ and $C_2 = e(h_u, u)$. Output commitment $c = (C_1, C_2)$ and equivocation-key $ek = (r, u)$.

$\mathsf{TC1.Equiv}(ck, \vec{m}, ek, tk)$: Parse ek into (r, u) and tk into $(\gamma_1, \delta_1 \ldots, \gamma_k, \delta_k)$. Then compute $r' = r \cdot \prod_{i=1}^{k} m_i^{-\gamma_i}$, and $u' = u \cdot \prod_{i=1}^{k} m_i^{-\delta_i}$. Then output decommit-key $dk = (r', u')$.

The above scheme shares many similarities with that of Groth [33], but the security is based on a different computational assumption, i.e. the SDP assumption. It should be noted that both assumptions are implied by DLIN assumption.

Theorem 12. *Trapdoor commitment scheme TC1 is perfectly hiding and computationally binding under the SDP assumption.*

Proof. For perfect hiding, observe that, for any $(C_1, C_2) \in \mathbb{G}_T^2$, any $\vec{m} \in \mathbb{G}_2^k$, there exits a unique $(r, u) \in \mathbb{G}_2^2$ that fulfills relation (33).

For computational binding, suppose that there exists an adversary that successfully opens a commitment to two distinct messages. We show that one can break SDP by using such an adversary. Given an instance of SDP, $(\Lambda, g_r, h_u, g_z, h_z)$, do as follows.

- Set $g_i = g_z^{\chi_i} g_r^{\gamma_i}$ and $h_i = h_z^{\chi_i} h_u^{\delta_i}$ for $i = 1, \ldots, k$. Run the adversary with $ck = (g_r, h_u, \{g_i, h_i\}_{i=1}^k)$.
- Given two openings $(\vec{m}, r, u)$ and $(\vec{m}', r', u')$ from the adversary, compute

$$z^\star = \prod_{i=1}^{k} (\frac{m_i}{m_i'})^{\chi_i}, \quad r^\star = \frac{r}{r'} \prod_{i=1}^{k} (\frac{m_i}{m_i'})^{\gamma_i}, \quad u^\star = \frac{u}{u'} \prod_{i=1}^{k} (\frac{m_i}{m_i'})^{\delta_i}. \tag{34}$$

- Output $(z^\star, r^\star, u^\star)$.

Since the openings fulfills (33), we have

$$\begin{aligned} 1 &= e\left(g_r, \frac{r}{r'}\right) \prod e\left(g_i, \frac{m_i}{m_i'}\right) = e\left(g_z, \prod_{i=1}^{k} \left(\frac{m_i}{m_i'}\right)^{\chi_i}\right) e\left(g_r, \frac{r}{r'} \prod_{i=1}^{k} \left(\frac{m_i}{m_i'}\right)^{\gamma_i}\right) \\ &= e(g_z, z^\star)\, e(g_r, r^\star), \text{ and} \\ 1 &= e\left(h_u, \frac{u}{u'}\right) \prod e\left(h_i, \frac{m_i}{m_i'}\right) = e\left(h_z, \prod_{i=1}^{k} \left(\frac{m_i}{m_i'}\right)^{\chi_i}\right) e\left(h_u, \frac{u}{u'} \prod_{i=1}^{k} \left(\frac{m_i}{m_i'}\right)^{\delta_i}\right) \\ &= e(h_z, z^\star)\, e(h_u, u^\star). \end{aligned}$$

But $\vec{m} \neq \vec{m}'$, so there exists i such that $m_i/m_i' \neq 1$. Also, χ_i is independent from the view of the adversary. That is, for every choice of χ_i, there exist corresponding γ_i and δ_i that gives the same g_i and h_i. Therefore, $z^\star = \prod_i (m_i/m_i')^{\chi_i} \neq 1$ with overwhelming probability. Hence $(z^\star, r^\star, u^\star)$ is a valid answer to the instance of SDP. ∎

B.2 Scheme TC2

Let $g \in \mathbb{G}_1$ and $\tilde{g} \in \mathbb{G}_2$ be random bases. Common parameter Λ is given to all algorithms described below.

- $\mathsf{TC2.Key}(1^\lambda)$: Select $\gamma \in \mathbb{Z}_p$ and set $\tilde{f} = \tilde{g}^\gamma$. Output commitment key $ck = (\Lambda, \tilde{f})$ and trapdoor $tk = \gamma$.
- $\mathsf{TC2.Com}(ck, m)$: Choose random $\delta \in \mathbb{Z}_p$ and compute commitment $c = \tilde{g}^m \tilde{f}^\delta \in \mathbb{G}_2$ and decommit-key $d = g^\delta \in \mathbb{G}_1$. Output c and d.
- $\mathsf{TC2.Vrf}(ck, c, m, d)$: Output 1 if $e(g, c/\tilde{g}^m) = e(d, \tilde{f})$. Output 0, otherwise.
- $\mathsf{TC2.Sim}(ck)$: Choose random $\delta \in \mathbb{Z}_p$ and output a commitment $c = \tilde{f}^\delta$ and an equivocation-key $ek = \delta$.
- $\mathsf{TC2.EqOpen}(ck, m, ek, tk)$: Let $\delta = ek$ and $\gamma = tk$. Output $d = g^{\delta - m/\gamma}$.

The correctness follows since

$$e(g, c/\tilde{g}^m) = e(g, \tilde{f}^\delta) = e(g^\delta, \tilde{f}) = e(d, \tilde{f}).$$

The trapdoor property holds because

$$e(d, \tilde{f}) = e(g^{\delta - m/\gamma}, \tilde{f}) = e(g, \tilde{g}^{-m} \tilde{f}^\delta) = e(g, c/\tilde{g}^m).$$

To prove computational binding property, we assume that the following variant of Diffie-Hellman inversion problem (XDHI) is hard with respect to Λ.

Assumption 5 (XDHI). Given Λ and $(g, \tilde{g}, \tilde{g}^a) \in \mathbb{G}_1^* \times \mathbb{G}_2^* \times \mathbb{G}_2^*$ for random $a \in \mathbb{Z}_p^*$, it is hard to compute $g^{1/a} \in \mathbb{G}_1$.

Depending on setting Λ, the XDHI assumption is implied by basic assumptions, Computational Diffie-Hellman Assumption (CDH), Computational Co-Diffie-Hellman Assumption (co-CDH) [12], and Decisional Diffie-Hellman Assumption in $\mathbb{G}_2$ ($\mathrm{DDH}_{\mathbb{G}_2}$), as follows. Note that, CDH is implied by DLIN in Λ_{sym} and $\mathrm{DDH}_{\mathbb{G}_2}$ is implied by SXDH in Λ_{sxdh}.

Lemma 3. *CDH $\Rightarrow$ XDHI for Λ_{sym}. co-CDH $\Rightarrow$ XDHI for Λ_{xdh}. $DDH_{\mathbb{G}_2} \Rightarrow$ XDHI for Λ_{sxdh}.*

Proof. Let $\mathcal{A}$ be an XDHI adversary. In Λ_{sym}, given an CDH instance (g, g^α, g^β), input (g^α, g^β, g) to $\mathcal{A}$. It outputs $g^{\alpha\beta}$, which is the answer to the CDH instance. Next, in Λ_{xdh}, given an co-CDH instance $(g, g^\alpha, \tilde{g}, \tilde{g}^\beta) \in \mathbb{G}_2^{*3}$, input $(g^\alpha, \tilde{g}^\beta, \tilde{g})$ to $\mathcal{A}$. It outputs $g^{\alpha\beta}$, which is the answer to the co-CDH instance. In Λ_{sxdh}, observe that, on input $(g, \tilde{g}^x, \tilde{g})$, adversary $\mathcal{A}$ outputs g^x. Thus $\mathcal{A}$ provides a mapping from $\mathbb{G}_2$ to $\mathbb{G}_1$. Now, given an instance $(\tilde{g}, \tilde{g}^\alpha, \tilde{g}^\beta, \tilde{g}^\gamma)$ of $\mathrm{DDH}_{\mathbb{G}_2}$, input $(g, \tilde{g}^\alpha, \tilde{g})$ to $\mathcal{A}$ and receive g^α. Then $\gamma = \alpha\beta$ can be tested by checking if $e(g^\alpha, \tilde{g}^\beta) = e(g, \tilde{g}^\gamma)$ holds or not. ∎

Theorem 13. *Trapdoor commitment scheme TC2 is perfectly hiding. It is binding if the XDHI assumption holds for Λ.*

Proof. The perfect hiding property holds from the fact that, for any $c \in \mathbb{G}_2$, for every $m \in \mathbb{Z}_p$ there exists a single consistent $\delta \in \mathbb{Z}_p$.

The binding property is proven by showing a reduction to XDHI. Given an instance of XDHI, $(g, \tilde{g}, \tilde{g}^a)$, set $\tilde{f} = \tilde{g}^a$. Suppose that an adversary outputs a commitment c correctly opened to (m, d) and (m', d') for $m \neq m'$. Then $e(g, c/\tilde{g}^m) = e(d, \tilde{f})$ and $e(g, c/\tilde{g}^{m'}) = e(d', \tilde{f})$ hold. By dividing both sides of the equations, we have $e(g, \tilde{g}^{m-m'}) = e(d'/d, \tilde{f}) = e(d'/d, \tilde{g}^a)$. Thus $(d'/d)^{1/m-m'} = g^{1/a}$, which is a correct answer to the XDHI instance. ∎

B.3 Scheme TC3

All components for this scheme is in $\mathbb{G}_1$ and $\mathbb{G}_2$. The underlying idea is to use TC1 and, instead of publishing a commitment in $\mathbb{G}_T$, we publish the decommit-key and the message in a randomized way by applying the one-side randomization RandOneSide from Section 3.

TC3.Key(1^λ): Choose random generators g_r, h_u from $\mathbb{G}_2^*$. For $i = 1, \ldots, k$, choose γ_i and δ_i from $\mathbb{Z}_p^*$ and compute $g_i = g_r^{\gamma_i}$ and $h_i = h_u^{\delta_i}$. Output commitment-key $ck = (\Lambda, g_r, h_u, \ldots, g_k, h_k)$ and trapdoor $tk = (\gamma_1, \delta_1 \ldots, \gamma_k, \delta_k)$.

TC3.Com($ck, \vec{m}$): Choose r and u randomly from $\mathbb{G}_2$, and compute

$$\{c_{ai}\}_{i=0}^k \leftarrow \mathsf{RandOneSide}((g_r, t), (g_1, m_1), \ldots, (g_k, m_k)), \text{ and} \tag{35}$$

$$\{c_{bi}\}_{i=0}^k \leftarrow \mathsf{RandOneSide}((h_u, w), (h_1, m_1), \ldots, (h_k, m_k)). \tag{36}$$

Output commitment $c = (\{c_{ai}\}_{i=0}^k, \{c_{bi}\}_{i=0}^k)$ and decommit-key $dk = (r, u)$.

TC3.Vrf($ck, c, \vec{m}, dk$): Parse c into $(\{c_{ai}\}_{i=0}^k, \{c_{bi}\}_{i=0}^k) \in \mathbb{G}_2^{2k+2}$ and dk into $(z, r, u) \in \mathbb{G}_2^3$. Output 1 if they satisfy the following predicates. Output 0, otherwise.

$$1 = e(g_r, r/c_{a0}) \prod_{i=1}^k e(g_i, m_i/c_{ai}) \quad \text{and} \quad 1 = e(h_u, u/c_{b0}) \prod_{i=1}^k e(h_i, m_i/c_{bi}) \tag{37}$$

TC3.Sim(ck): Do the same as TC3.Com with $\vec{m} = (1, \ldots, 1)$.

TC3.Equiv($ck, \vec{m}, ek, tk$): Parse ek into (r, u) and tk into $(\gamma_1, \delta_1 \ldots, \gamma_k, \delta_k)$. Then compute $r' = r \cdot \prod_{i=1}^k m_i^{-\gamma_i}$, and $u' = u \cdot \prod_{i=1}^k m_i^{-\delta_i}$. Then output decommit-key $dk = (r', u')$.

Theorem 14. *Trapdoor commitment scheme TC3 is perfectly hiding and computationally binding under the SDP assumption.*

The hiding property is clear from the uniform output property of RandOneSide and that of TC1. The binding property is taken over from TC1 and can be proven in the same way as for TC1.

B.4 Scheme TC4

This is the most efficient scheme both in computation and storage. The scheme virtually 'half' the scheme of TC1. Let $\Lambda \in \{\Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$.

TC4.Key(1^λ): Choose random generators g_r from $\mathbb{G}_1^*$. For $i = 1, \ldots, k$, choose γ_i from $\mathbb{Z}_p^*$ and compute $g_i = g_r^{\gamma_i}$. Output commitment-key $ck = (\Lambda, g_r, g_1, \ldots, g_k)$ and trapdoor $tk = (\gamma_1, \ldots, \gamma_k)$.

$\mathsf{TC4.Com}(ck, \vec{m})$: Choose r randomly from $\mathbb{G}_2$, and compute

$$c = e(g_r, r) \prod_{i=1}^{k} e(g_i, m_i). \tag{38}$$

Output commitment c and decommit-key $dk = r$.

$\mathsf{TC4.Vrf}(ck, c, \vec{m}, dk)$: Output 1 if (38) holds. Output 0, otherwise.

$\mathsf{TC4.Sim}(ck)$: Choose r randomly from $\mathbb{G}_2$ and compute $c = e(g_r, r)$. Output commitment c and equivocation-key $ek = r$.

$\mathsf{TC4.Equiv}(ck, \vec{m}, ek, tk)$: Take r and $(\gamma_1, \ldots, \gamma_k)$ out from ek and tk, respectively. Then compute $r' = r \cdot \prod_{i=1}^{k} m_i^{-\gamma_i}$, and Then output decommit-key $dk = r'$.

Theorem 15. *TC4 is perfectly hiding and computationally binding if the DBP assumption holds for* Λ.

Proof. The hiding property holds because, for any commitment $c \in \mathbb{G}_T$ and any $\vec{m} \in \mathbb{G}_2^k$, there exists consistent $t \in \mathbb{G}_2$ that fulfills relation (38).

The binding property is shown similarly to Theorem 12:
Given an instance of DBP, (Λ, g_z, g_r), do as follows.

- Set $g_i = g_z^{\chi_i} g_r^{\gamma_i}$. Run the adversary with $ck = (g_r, \{g_i\}_{i=1}^k)$.
- Given two openings $(\vec{m}, r)$ and $(\vec{m}', r')$ from the adversary, compute

$$z^\star = \prod_{i=1}^{k} (m_i/m_i')^{\chi_i}, \qquad r^\star = (r/r') \prod_{i=1}^{k} (m_i/m_i')^{\gamma_i}. \tag{39}$$

- Output $(z^\star, r^\star)$.

Since the openings fulfills (33), we have

$$\begin{aligned} 1 &= e\left(g_r, \frac{r}{r'}\right) \prod e\left(g_i, \frac{m_i}{m_i'}\right) = e\left(g_z, \prod_{i=1}^{k} \left(\frac{m_i}{m_i'}\right)^{\chi_i}\right) e\left(g_r, \frac{r}{r'} \prod_{i=1}^{k} \left(\frac{m_i}{m_i'}\right)^{\gamma_i}\right) \\ &= e(g_z, z^\star)\, e(g_r, r^\star). \end{aligned}$$

But $\vec{m} \neq \vec{m}'$, so there exists i such that $m_i/m_i' \neq 1$. Also, χ_i is independent from the view of the adversary. That is, for every choice of χ_i, there exist corresponding γ_i that gives the same g_i. Therefore, $z^\star = \prod_i (m_i/m_i')^{\chi_i} \neq 1$ with overwhelming probability. Hence $(z^\star, r^\star)$ is a valid answer to the instance of SDP. ∎

One can have a variant of TC4 whose commitment is in $\mathbb{G}_1$ and $\mathbb{G}_2$ in a similar way we convert TC1 to TC3. Unlike the previous case, however, RandOneSide cannot be used as TC4 is in $\Lambda = \Lambda_{\mathsf{sxdh}}$. So we instead use RandSeq keeping g_r and h_u intact. This modification results in $2k + 1$ group elements in a commitment, which is 1 element less than that of TC4. However, depending on applications, this may not be a gain since the resulting verification predicate is not one-sided.

C One-Time Signature Schemes

C.1 A One-Time Signature Scheme in Any Setting

Let $\Lambda \in \{\Lambda_{\mathsf{sym}}, \Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$.

- $\mathsf{OTS1.Key}(1^\lambda)$: Choose random generators $g_z, h_z, g_r, h_u \leftarrow \mathbb{G}_1^*$. For $i = 1, \ldots, k$, choose χ_i, γ_i, $\delta_i \leftarrow \mathbb{Z}_p^*$ and compute $(g_i, h_i) = (g_z^{\chi_i} g_r^{\gamma_i}, h_z^{\chi_i} h_u^{\delta_i})$. Also choose $\zeta, \rho, \varphi \leftarrow \mathbb{Z}_p^*$ and set $(a, \tilde{a}) \leftarrow \mathsf{Rand}(g_z^\zeta g_r^\rho, \tilde{g})$ and $(b, \tilde{b}) \leftarrow \mathsf{Rand}(h_z^\zeta h_u^\varphi, \tilde{g})$. Set $vk = (g_z, h_z, g_r, h_u, \{g_i, h_i\}_{i=1}^k, a, \tilde{a}, b, \tilde{b})$ and $sk = (vk, \zeta, \rho, \varphi, \{\chi_i, \gamma_i, \delta_i\}_{i=1}^k)$. Output (vk, sk).

- $\mathsf{OTS1.Sign}(sk, \vec{m})$: Compute

$$z = \tilde{g}^\zeta \prod_{i=1}^k m_i^{-\chi_i}, \quad r = \tilde{g}^\rho \prod_{i=1}^k m_i^{-\gamma_i}, \quad u = \tilde{g}^\varphi \prod_{i=1}^k m_i^{-\delta_i}.$$

  Output $\sigma = (z, r, u)$ as a signature.

- $\mathsf{OTS1.Vrf}(vk, \vec{m}, \sigma)$: Parse σ into (z, r, u). Output 1 if the following equations hold. Output 0, otherwise.

$$e(a, \tilde{a}) = e(g_z, z)\, e(g_r, r) \prod_{i=1}^k e(g_i, m_i) \tag{40}$$

$$e(b, \tilde{b}) = e(h_z, z)\, e(h_u, u) \prod_{i=1}^k e(h_i, m_i) \tag{41}$$

Theorem 16. *One-time signature scheme OTS1 is strongly unforgeable against one-time chosen message attacks if SDP holds for Λ.*

Proof. Suppose that there is an adversary, $\mathcal{A}$, that finds a forged signature $\sigma^\dagger = (z^\dagger, r^\dagger, u^\dagger)$ for message $\vec{m}^\dagger$ after seeing a one-time signature (z, r, u) for message $\vec{m}$ of its choice. We construct a reduction algorithm to SDP as follows.

Given an instance (g_z, h_z, g_r, h_u) of SDP, do the same as $\mathsf{OTS1.Key}$ by using the input instance as the bases. When $\mathcal{A}$ submit message $\vec{m}$, run $\mathsf{OTS1.Sign}$ and return (z, r, u) to $\mathcal{A}$. Given output $(z^\dagger, r^\dagger, u^\dagger)$ and $\vec{m}^\dagger$ from $\mathcal{A}$, compute

$$z^\star = (z^\dagger / z) \prod_{i=1}^k (m_i^\dagger / m_i)^{\chi_i}, \quad r^\star = (r^\dagger / r) \prod_{i=1}^k (m_i^\dagger / m_i)^{\gamma_i}, \quad u^\star = (u^\dagger / u) \prod_{i=1}^k (m_i^\dagger / m_i)^{\delta_i}. \tag{42}$$

Then output $(z^\star, r^\star, u^\star)$. This completes the description of the reduction algorithm.

Suppose that adversary $\mathcal{A}$ is successful. By dividing both sides of (40) with respect to $(z^\star, r^\star, u^\star)$ and (z, r, u), we have

$$\begin{aligned} 1 &= e(g_z, z^\dagger / z)\, e(g_r, r^\dagger / r) \prod_{i=1}^k e(g_i, m_i^\dagger / m_i) \\ &= e(g_z, z^\dagger / z \prod_{i=1}^k (m_i^\dagger / m_i)^{\chi_i})\, e(g_r, r^\dagger / r \prod_{i=1}^k (m_i^\dagger / m_i)^{\gamma_i}) \\ &= e(g_z, z^\star)\, e(g_r, r^\star). \end{aligned}$$

Similarly, with respect to (41), we have

$$\begin{aligned}1 &= e(h_z, z^\dagger/z)\, e(h_u, u^\dagger/u) \prod_{i=1}^{k} e(h_i, m_i^\dagger/m_i) \\ &= e(h_z, z^\dagger/z \prod_{i=1}^{k} (m_i^\dagger/m_i)^{\chi_i})\, e(h_u, u^\dagger/u \prod_{i=1}^{k} (m_i^\dagger/m_i)^{\delta_i}) \\ &= e(h_z, z^\star)\, e(h_u, u^\star).\end{aligned}$$

Hence $(z^\star, r^\star, u^\star)$ is a correct answer to the SDP instance.

What remains to show is $z^\star \neq 1$. We first consider the case of $\vec{m} = \vec{m}^\dagger$. In this case, $(z^\dagger, r^\dagger, u^\dagger) \neq (z, r, u)$ must hold. Observe that $z^\dagger = z$ cannot be the case since it implies $r^\dagger = r$ and $u^\dagger = u$ to fulfill (40) and (41). Since $\vec{m} = \vec{m}^\dagger$, we have $z^\star = z^\dagger/z \neq 1$. Next we consider the case of $\vec{m} \neq \vec{m}^\dagger$. In this case, there exists $i^\star$ for which $m_{i^\star} \neq m_{i^\star}^\dagger$ holds. For such $i^\star$, parameter $\chi_{i^\star}$ is information theoretically independent from the view of the adversary. Namely, for any view of the adversary and for any $\chi_{i^\star} \in \mathbb{Z}_p^*$, there exist consistent $\tilde{g}^\zeta$, $\gamma_{i^\star}$, and $\delta_{i^\star}$. This can be verified by seeing that $(a, \tilde{a})$, $(b, \tilde{b})$, and g_i, h_i are perfectly hiding commitment of ζ and $\chi_{i^\star}$ and the one-time signature does not identify ζ and $\chi_{i^\star}$ at the same time. Therefore, having the factor of $(m_{i^\star}^\dagger/m_{i^\star})^{\chi_{i^\star}}$ with $m_{i^\star}^\dagger/m_{i^\star} \neq 1$, $z^\star = 1$ happens only with negligible probability over the choice of $\chi_{i^\star}$. ∎

C.2 More Efficient Scheme in the Asymmetric Setting

In the case of $\Lambda \in \{\Lambda_{\mathsf{xdh}}, \Lambda_{\mathsf{sxdh}}\}$ we can construct a more efficient scheme, say OTS2, that halves OTS1 just like TC4 does for TC1. The verification equation would be:

$$e(a, \tilde{a}) = e(g_z, z)\, e(g_r, r) \prod_{i=1}^{k} e(g_i, m_i) \tag{43}$$

Scheme OTS2 is strongly unforgeable against one-time chosen message attacks under the DBP assumption.

D Signing Unbounded-Size Messages – Alternative Construction

When the spaces for messages and verification keys are compatible, signing unbounded-size messages is generally possible (with some constraint about the message length) by the "double-decker" approach inspired by [25].[2] On receiving a message, first sign an ephemeral verification-key and the message length by using the base signing-key. Then divide the message into appropriate size as determined by the ephemeral key and sign each piece with a tag that identifies the position by using the ephemeral key repeatedly. A positive point of this approach is that it is as general as to accept a signature scheme with linear-size signatures as the building block. On the other hand, it chokes the message space for the individual signing and yields considerable overhead both in computation and storage. We give a formal analysis and efficiency assessment to this generic construction in the following.

[2]The generic extension in [25] is vulnerable against chosen message attacks: One can create a signature on $\vec{m} = (\langle 3\rangle, \langle 1\rangle, \langle 2\rangle)$ by swapping some elements in a given signature on $\vec{m}' = (\langle 2\rangle, \langle 3\rangle, \langle 1\rangle)$ where $\langle n\rangle$ is an encoding of integer n to an element in the message space.

CONSTRUCTION. Let SIG be a signature scheme whose verification key size is $\alpha_1 k+\alpha_2$ and signature size is $\beta_1 k + \beta_2$ for messages of size k where α_1, α_2, β_1, and β_2 are non-negative constants. It is required that k can be set to ≥ 2 for SIG. The construction is as follows.

- $\mathsf{USIG2.Key}(1^\lambda)$: Run $(vk, sk) \leftarrow \mathsf{SIG.Key}(1^\lambda)$ with parameter k.
- $\mathsf{USIG2.Sign}(sk, \vec{m})$: Let $k' = \lfloor \frac{k-\alpha_2-1}{\alpha_1} \rfloor$ and $n = \lceil \frac{|\vec{m}|}{k'-1} \rceil$. Run $(vk', sk') \leftarrow \mathsf{SIG.Key}(1^\lambda)$ with parameter k'. Set $\vec{m}_0 = vk' || \langle |\vec{m}| \rangle$ and compute $\sigma_0 \leftarrow \mathsf{SIG.Sign}(sk, \vec{m}_0)$. For $i = 1, \ldots, n$, set $\vec{m}_i = \langle i \rangle || (m_{(i-1)(k'-1)+1}, \ldots, m_{i(k'-1)})$ and compute $\sigma_i \leftarrow \mathsf{SIG.Sign}(sk', \vec{m}_i)$. Output $\sigma = (vk', \sigma_0, \ldots, \sigma_n)$.
- $\mathsf{USIG2.Vrf}(vk, \vec{m}, \sigma)$: Let $n = \lceil \frac{|\vec{m}|}{k'-1} \rceil$. Set $\vec{m}_0 = vk' || \langle |\vec{m}| \rangle$ and compute $b_0 = \mathsf{SIG.Vrf}(vk, \vec{m}_0, \sigma_0)$. For $i = 1, \ldots, n$, set $\vec{m}_i = \langle i \rangle || (m_{(i-1)(k'-1)+1}, \ldots, m_{i(k'-1)})$ and compute $b_i = \mathsf{SIG.Vrf}(vk', \vec{m}_i, \sigma_i)$. Output 1 if $b_i = 1$ for all $i = 0, \ldots, n$. Output 0, otherwise.

If $|\vec{m}|$ is not a multiple of $k' - 1$, an appropriate padding is applied in both $\mathsf{USIG2.Sign}$ and $\mathsf{USIG2.Vrf}$. The length of a signature is $|\sigma| = \beta_1 k + \beta_2 + n(\beta_1 k' + \beta_2)$.

SECURITY.

Theorem 17. *If SIG is EUF-CMA, so is USIG2.*

Proof. Given verification-key vk, adversary $\mathcal{A}$ launches an adaptive chosen message attack to $\mathsf{USIG2}$. It eventually outputs a successful forgery, $(\vec{m}^\dagger, \sigma^\dagger)$. Let $\sigma^\dagger = (vk'^\dagger, \sigma_0^\dagger, \ldots, \sigma_n^\dagger)$. Let $\vec{m}_i^\dagger$ for $i = 0, \ldots, n$ be message vector that associates to $\sigma_i^\dagger$. Let $\{\vec{m}_0\}$ be messages signed by vk. Let $\{\vec{m}_i\}_{vk'}$ be messages signed by vk'. We classify adversaries by two events.

- $(\vec{m}_0^\dagger \notin \{\vec{m}_0\})$: In this case, $(\vec{m}_0^\dagger, \sigma_0^\dagger)$ is a valid forgery for underlying SIG with vk. For adversaries that causes this case, we simulate $\mathsf{USIG2}$ in a straightforward manner by using the signing oracle of SIG with respect to vk.
- $(\vec{m}_0^\dagger \in \{\vec{m}_0\})$: Let $\vec{m}_0^\dagger = vk'^\dagger || \langle n \rangle$. Let $\vec{m}$ be a message given to the signing oracle of $\mathsf{USIG2}$ when $\vec{m}_0$ appears. Since $\vec{m}^\dagger$ must be fresh, $\vec{m}^\dagger \neq \vec{m}$. Thus there exists $i^\star \in \{1, \ldots, n\}$ where $\vec{m}_{i^\star}^\dagger \neq \vec{m}_{i^\star}$ holds. Observe that both $\vec{m}_{i^\star}^\dagger$ and $\vec{m}_{i^\star}$ has $\langle i^\star \rangle$ as the first element. Furthermore, since vk' is used only once with high probability, $\vec{m}_{i^\star}$ is the only one in $\{\vec{m}_i\}_{vk'}$ that has $\langle i^\star \rangle$ as the first element. Thus, $\vec{m}_{i^\star}^\dagger \notin \{\vec{m}_i\}_{vk'}$ holds. Accordingly, $(\vec{m}_{i^\star}^\dagger, \sigma_{i^\star}^\dagger)$ is a valid forgery with respect to vk'. For such adversary, the reduction algorithm first guesses from which query adversary picks vk' and generate (vk, sk) honestly. It then simulate a signature to that query by using sk to sign $\vec{m}_0^\dagger$ and asking the signing oracle of SIG for the remaining part of the signature. Other queries are answered correctly by using sk.

In either case, existence of successful adversary $\mathcal{A}$ to $\mathsf{USIG2}$ contradicts to EUF-CMA of SIG. ∎

EFFICIENCY. We estimate the efficiency of $\mathsf{USIG2}$ for the cases where SIG is instantiated with the scheme in [22] and the one in Section 4. To see concrete figures, we estimate the size of a self-certificate, i.e., a signature on the verification key itself. See Table 4 for summary. For comparison, we include the figures about the signature-chain scheme from Section 5.2 in the table.

Scheme	Underlying Signature	verification key size	signature size at $k = 18$	self-certificate size at $k = 18$
USIG2 (double-decker)	[22]	3k+11	$22\|\vec{m}\| + 166$	1596
USIG2 (double-decker)	SIG (Sec.4)	2k+12	$7\|\vec{m}\| + 7$	343
USIG1 (sig-chaining)	SIG (Sec.4)	2k+14	$7\lceil\frac{\|\vec{m}\|+1}{16}\rceil$	28

Table 4: Summary of efficiency assessment for schemes for unbounded-size messages. The size counts the number of group elements. Parameter k is the maximum message size for the underlying signature scheme and set to 18 in computing the size of a signature and self-certificate. The size of a self-certificate counts the number of group elements in a signature when $\vec{m}$ is its own verification key.

- With the scheme in [22], the parameter is $(\alpha_1, \alpha_2, \beta_1, \beta_2) = (3, 11, 9, 4)$. [3] Since $k' \geq 2$, we have $k \geq 18$. When $k' = 2$ and $k = 18$, the total signature size for message of size $|\vec{m}|$ is $9{\cdot}18{+}4{+}|\vec{m}|(9{\cdot}2{+}4) = 22|\vec{m}|{+}166$. A self-certificate thus consists of $22(3{\cdot}18{+}11){+}166 = 1596$ group elements.
- With the scheme in Section 4, the parameter is $(\alpha_1, \alpha_2, \beta_1, \beta_2) = (2, 12, 0, 7)$. Since $k' \geq 2$, we have $k \geq 17$. With the same setting $k' = 2$ and $k = 18$, a signature is in size $7|\vec{m}| + 7$. A self-certificate thus consists of $7(2 \cdot 18 + 12) + 7 = 343$ group elements.
- With the signature-chain construction instantiated with our scheme in Section 5.2, the parameter is $(\alpha_1, \alpha_2, \beta_1, \beta_2) = (2, 14, 0, 7)$. The size of signature for $k = 18$ is $7\lceil\frac{|\vec{m}|+1}{16}\rceil$. A self-certificate thus consists of $7\lceil\frac{2\cdot18+15}{16}\rceil = 28$ group elements.

E Groth-Sahai Proof System

The Groth-Sahai (GS) proof system [35] gives efficient non-interactive witness-indistinguishable (NIWI) proofs and non-interactive zero-knowledge (NIZK) proofs for languages that can be described as a set of satisfiable equations, each of which falls in one of the following categories: pairing product equations, multi-exponentiation equations, and general arithmetic gates; we describe those equation types later in this section. The proof system could be instantiated under different assumption, in particular Λ_{sxdh} setting (which is asymmetric pairings for which the the DDH holds both in $\mathbb{G}_1$ and $\mathbb{G}_2$) or under the DLIN assumption in the Λ_{sym} setting. Below, we define $K = 1$ when using the Λ_{sxdh} setting and $K = 2$ when using the latter.

The proof system has two type of CRS which are computationally indistinguishable: one called "real" which yields perfect soundness and allows extraction if in possession of the trapdoor key, and another "simulated" which yields perfect witness-indistinguishable (WI) proofs which could also be made zero-knowledge (ZK) for some of the equations. The common reference strings consists of $\vec{\mathsf{u}}_0, \vec{\mathsf{u}}_1, \ldots, \vec{\mathsf{u}}_K, \vec{\mathsf{u}} \in \mathbb{G}_1^{K+1}$, where $\vec{\mathsf{u}}_1, \ldots, \vec{\mathsf{u}}_K$ are linearly independent vectors as defined below and $\vec{\mathsf{u}}_0$ and $\vec{\mathsf{u}}$ are defined differently depending on the CRS. For a real CRS, $\vec{\mathsf{u}}_0$ is chosen to be a random linear combination of $\vec{\mathsf{u}}_1, \ldots, \vec{\mathsf{u}}_K$; and $\vec{\mathsf{u}}$ is chosen to be random and linearly independent of those vectors. This way, the commitments defined in the next paragraph are perfectly binding. Whereas for a simulated CRS, $\vec{\mathsf{u}}_0$ is chosen to be random and linearly independent of $\vec{\mathsf{u}}_1, \ldots, \vec{\mathsf{u}}_K$; $\vec{\mathsf{u}}$ is chosen as a random linear combination of those vectors; and the commitments are perfectly hiding. In both cases $\vec{\mathsf{u}}_1 = (\mathsf{u}_0, \mathsf{u}_1, 1, \ldots, 1)$, $\vec{\mathsf{u}}_2 = (\mathsf{u}_0, 1, \mathsf{u}_2, 1, \ldots, 1)$, $\ldots$, $\vec{\mathsf{u}}_K = (\mathsf{u}_0, 1, \ldots, 1, \mathsf{u}_K)$ for randomly chosen $\mathsf{u}_0, \mathsf{u}_1, \ldots, \mathsf{u}_K \in \mathbb{G}_1^*$. The CRS also includes analogously defined $\vec{\mathsf{v}}_0, \vec{\mathsf{v}}_1, \ldots, \vec{\mathsf{v}}_K, \vec{\mathsf{v}} \in \mathbb{G}_2^{K+1}$.

[3] The original scheme in [22] includes $e(g, g)^\alpha \in \mathbb{G}_T$ in a verification key. One can instead include $g^{\alpha/\gamma}$ and g^γ with random γ without impacting to the security.

When proving a statement, described as a set of equations, the GS proof system first commits to the witness components and then for each equation produces proof elements that the corresponding committed values satisfy the equation. The commitments are done as follows: to commit to $x \in G_1$, compute $\vec{\mathfrak{c}}_x \leftarrow GSCom(x; \vec{r}) = (x, 1, \ldots, 1) \prod_{j=0}^{K} \vec{\mathfrak{u}}_j^{r_j}$, where $\vec{r} \leftarrow \mathbb{Z}_p^{K+1}$; similarly commit to $\tilde{x} \in G_2$ as $GSCom(\tilde{x}; \vec{r}) = (\tilde{x}, 1, \ldots, 1) \prod_{j=0}^{K} \vec{\mathfrak{v}}_j^{r_j}$ for a randomly chosen $\vec{r}$. Committing to $\chi \in \mathbb{Z}_p$ is done by choosing random $\vec{r} \in \mathbb{Z}_p^K$ and computing $\vec{\mathfrak{c}}_\chi \leftarrow GSCom(\chi; \vec{r}) = \vec{\mathfrak{v}}^\chi \prod_{j=1}^{K} \vec{\mathfrak{v}}_j^{r_j}$ which is used for multi-exponentiation equations in $\mathbb{G}_1$; for same type of equations in $\mathbb{G}_2$ we commit as $\vec{\mathfrak{u}}^\chi \prod_{j=1}^{K} \vec{\mathfrak{u}}_j^{r_j}$. Each commitment consists of $(K+1)$ group elements.

From the type of equations the GS proof system supports, we are usually interested in pairing product equations, e.g. $\prod_{i=1}^{n} e(x_i, \tilde{x}_i) = T$, and multi-exponentiations, e.g. $\prod_{i=1}^{n} x_i^{\chi_i} = t$. Both types have slightly more general forms (see [35]) and also simpler (and more efficient) forms of which we are often interested. A one-sided equation has a witness with all components being committed either in $\mathbb{G}_1^{K+1}$ or $\mathbb{G}_2^{K+1}$, but not in both as the multi-exponentiation example above for which x-s are committed in the former space but χ-s are committed in the latter. The one-sided pairing production equations are of the form $\prod_{i=1}^{n} e(g_i, \tilde{x}_i) = T$, where $g_1, \ldots, g_n$ are some constant group elements; we use this particular type in Section 9.1. Similarly, the one-sided multi-exponentiation equations are of the form $\prod_{i=1}^{n} g_i^{\chi_i} = t$, where g-s are constants.

As mentioned before, a GS proof first commits to each witness component and then produces proof elements for each equation (in the set of equations being shown satisfiable). Below we describe how those elements are computed for the one-sided equations (the simplest of all) borrowing some of the notation from [15], and refer the reader to [35] for full description in general.

One-sided Multi-exponentiation Equations

We consider the one-sided multi-exponentiation in $\mathbb{G}_2$; the case of $\mathbb{G}_1$ is handled analogously.

For an equation of the following type:

$$g_0 = g_1^{\chi_1} g_2^{\chi_2} \ldots g_n^{\chi_n}$$

where $g_0, \ldots, g_n \in \mathbb{G}_2$ are constants (one could view an equation being described by those constants) and $\chi_1, \ldots, \chi_n \in \mathbb{Z}_q$ are variables (the witness for which the equation is satisfiable), the proof elements are $p_1, \ldots, p_K$:

$$p_j = \prod_{i=1}^{n} g_i^{r_{ij}},\ j = 1, \ldots, K,$$

where $\vec{r_i} \in \mathbb{Z}_p^K$ is the randomness used to commit to χ_i, i.e. $\vec{\mathfrak{c}}_{\chi_i} \leftarrow GSCom(\chi_i; \vec{r_i})$.

When verifying a proof, for each equation $g_0 = g_1^{\chi_1} g_2^{\chi_2} \ldots g_n^{\chi_n}$ the verifier checks that the proof elements corresponding to the equation and the commitments satisfy

$$\prod_{i=1}^{n} E(\vec{\mathfrak{c}}_{\chi_i}, g_i) = E(\vec{\mathfrak{u}}, g_0) \prod_{j=1}^{K} E(\vec{\mathfrak{u}}_j, p_j),$$

where $E : \mathbb{G}_1^{K+1} \times \mathbb{G}_2 \to \mathbb{G}_T^{K+1}$, sending $((\alpha_0, \ldots, \alpha_K), \beta)$ to $(e(\alpha_0, \beta), \ldots, e(\alpha_K, \beta))$, is a bilinear map.

The proofs are ZK. A proof for set of q equations being satisfiable with a witness of size n results in a proof of size $(K+1)n + Kq$ group elements. Note again that $K = 1$ or 2 depending on the setting we work in.

One-sided Pairing Product Equations

For an equation

$$\prod_{i=1}^{n} e(g_i, \tilde{x}_i) = T$$

where $g_1, \ldots, g_n \in \mathbb{G}_1$ and $T \in \mathbb{G}_T$ are constants and $\tilde{x}_1, \ldots, \tilde{x}_n \in \mathbb{G}_2$ are variables, the proof elements $p_0, \ldots, p_K$ are computed as follows:

$$p_j = \prod_{i=1}^{n} g_i^{r_{ij}},\ j = 0, \ldots, K,$$

where $\vec{r_i} \in \mathbb{Z}_p^{K+1}$ is the randomness used to commit to $\tilde{x}_i$, i.e. $\vec{\mathfrak{c}}_{\tilde{x}_i} \leftarrow GSCom(\tilde{x}_i; \vec{r_i})$. When verifying a proof, for each equation $\prod_{i=1}^{n} e(g_i, \tilde{x}_i) = T$ the verifier checks that the proof elements corresponding to the equation and the commitments satisfy

$$\prod_{i=1}^{n} E(g_i, \vec{\mathfrak{c}}_{\tilde{x}_i}) = (T, 1, 1, \ldots, 1) \prod_{j=0}^{K} E(p_j, \vec{\mathfrak{v}}_j),$$

where $E : \mathbb{G}_1 \times \mathbb{G}_2^{K+1} \to \mathbb{G}_T^{K+1}$, sending $(\alpha, (\beta_0, \ldots, \beta_K))$ to $(e(\alpha, \beta_0), \ldots, e(\alpha, \beta_K))$, is a bilinear map.

These proofs are only witness indistinguishable (WI), and for a set of q equations satisfiable with a witness of size n, the proof size is $(K + 1)(n + q)$.

When representation of T as a pairing product is known, the trivial such case is $T = 1_{\mathbb{G}_T}$, the proof could be transformed into ZK but results in little less efficient proofs [35].