J Cryptol (2019) 32:973–1025
https://doi.org/10.1007/s00145-018-9300-5

Journal of
CRYPTOLOGY

Efficient Fully Structure-Preserving Signatures and Shrinking Commitments

Masayuki Abe
Secure Platform Laboratories, NTT Corporation, Tokyo, Japan
abe.masayuki@lab.ntt.co.jp

Jens Groth
Department of Computer Science, University College London, London, UK
j.groth@ucl.ac.uk

Markulf Kohlweiss
School of Informatics, University of Edinburgh, Edinburgh, UK
mkohlwei@ed.ac.uk

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

Mehdi Tibouchi
Secure Platform Laboratories, NTT Corporation, Tokyo, Japan
tibouchi.mehdi@lab.ntt.co.jp

Communicated by Ran Canetti.

Received 31 August 2016 / Revised 5 April 2018
Online publication 8 August 2018

Abstract. In structure-preserving signatures, public keys, messages, and signatures are all collections of source group elements of some bilinear groups. In this paper, we introduce fully structure-preserving signature schemes, with the additional requirement that even secret keys are group elements. This strong property allows efficient non-interactive proofs of knowledge of the secret key, which is useful in designing cryptographic protocols under simulation-based security where online extraction of the secret key is needed. We present efficient constructions under simple standard assumptions and pursue even more efficient constructions with the extra property of randomizability based on the generic bilinear group model. An essential building block for our efficient standard model construction is a shrinking structure-preserving trapdoor commitment scheme, which is by itself an important primitive and of independent interest as it appears to contradict a known impossibility result that structure-preserving commitments cannot be shrinking. We argue that a relaxed binding property lets us circumvent the impossibility while still retaining the usefulness of the primitive in important applications as mentioned above.

Keywords. Structure-preserving signatures, Structure-preserving commitments, Secret key extraction, Randomizability.

1. Introduction

In pairing-based cryptography, cryptographic primitives are often designed to have algorithms in which messages and public materials consist only of source group elements and correctness can be proved using pairing product equations to allow smooth coupling with other primitives. This interest in the so-called structure-preserving primitives [3] led to the study of algebraic algorithms with many positive but also negative results, e.g., [1,2,4–7,12,22,27,41].

In structure-preserving signature schemes, all components but secret keys are group elements. This raises a natural question: "*Can secret keys consist entirely of source group elements as well*?" A major obstacle in designing structure-preserving signatures is that messages are group elements so standard signature schemes relying on exponentiation with the message or some function thereof do not work. In existing structure-preserving signature schemes, this is overcome by having secret keys that are used in exponents. Thus, it is quite unclear how to construct structure-preserving signatures if even secret keys are group elements.

Besides the above question being a fascinating fundamental question in its own right, it is connected to practical protocol design since group element secret keys combined with the Groth–Sahai proof system [38] allow straight line (i.e., no rewinding) extraction of the secret keys. Verifiably encrypted secret keys are for instance useful in delegatable anonymous credential systems [13,26,32] extended with all-or-nothing non-transferability [24]. More applications of secret key extraction are introduced in Sect. 7.

While there are solutions in the random oracle model, e.g., [23,31], secret key extraction without random oracles is currently prohibitively expensive. Meiklejohn [43] demonstrates how to extract a secret key in the exponent using the Groth–Sahai proofs. It requires bit-by-bit decomposition of secret x, and the proof consists of $20\ \log_2 x + 18$ group elements. For instance, applying it to a structure-preserving signature scheme [2] whose secret key consists of $4 + 2L$ scalar values for signing messages of L group elements, proving secret keys for signing 10 group elements at the 128-bit security level, requires more than 61,000 group elements.

Our Contribution We summarize our contribution in the following.

1. We introduce the notion of *fully structure-preserving signature (FSPS)* schemes. FSPS is signature schemes whose message, signature, and key spaces, including secret keys, consist entirely of source group elements of bilinear groups. This extends the paradigm of structure-preserving cryptography to cover private materials.
2. We present a concrete construction of FSPS based on static (i.e., not q-type) standard assumptions. Its secret key consists only of four group elements, and a witness indistinguishable proof of knowledge of the secret key consists of 18 group elements (see Table 3 in Sect. 4.3). These are huge savings compared to the current bit-by-bit proof of knowledge of a secret exponent mentioned earlier.
3. We present a *shrinking structure-preserving trapdoor commitment scheme* (SPTC) that produces constant-size commitments consisting of a single group element regardless of the message size. It is used as an essential building block in our first construction of FSPS. Besides being an important primitive in itself, it is a remarkable result in light of the well-known impossibility [8] that SPTC schemes yielding

shorter commitments than messages cannot be binding. We get around the impossibility by relaxing the binding property only to honestly created commitments as in [28]. The relaxed notion that we call *chosen-message target collision resistance* (CMTCR) is in between collision resistance and target collision resistance if we use the terminology for hash functions. We show that CMTCR is sufficient to construct secure signature schemes in combination with a weak signature scheme that is secure only against extended random message attacks [2] where private coins used to chose random messages are exposed to the adversary.

4. To push efficiency further, we give a direct FSPS construction, where security is proved in the generic bilinear group model [18,19,40,42,45,48]. The latter construction has an optimally short verification key consisting of only a single group element. Furthermore, we show that its signatures can be either randomizable or strongly unforgeable. Recall that some structure-preserving signature schemes in the literature are randomizable meaning that some elements in a signature can be changed without losing correctness or security. This property is useful in particular when combining structure-preserving signatures with Groth–Sahai proofs since some of the signature elements can be revealed in the clear after being randomized. In other circumstances, however, quite the opposite may be the case and it may be desirable to have strongly unforgeable signatures where it is not only infeasible to forge signatures on messages that have not been seen before but it is also infeasible to create a new different signatures on messages that have already been signed. We define the notion of a *combined* signature scheme where the signer can choose for each message whether to make the signature strongly unforgeable or randomizable, and our latter construction is a combined signature scheme.
5. All our signatures have size $\Omega(\sqrt{L})$ for messages consisting of L group elements. This is no coincidence; we show that, for a (one-time) SPS scheme over the so-called Type-III asymmetric bilinear groups, there is a non-trivial trade-off $\kappa + \sigma \geq \sqrt{L}$ among verification key size κ, signature size σ, and message size L in the number of group elements. This means our constructions have optimal signature size for constant-size verification keys. We leave it as an open question to show (in)feasibility of constant-size signatures for verification keys of size growing in L. Such an alternative is more advantageous in communications with constrained bandwidth.

Related Works At least one FSPS scheme already exists [2] but with constraints on both security and usability. Namely, it only meets the weak security guarantee (unforgeability against extended random message attacks), and the signing function takes messages of the form (G^m, F^m, U^m) that essentially requires knowledge of m [14,39]. Nevertheless, it is a reasonable starting point for constructions in the standard model. Wang et al. in [50] improve our framework and present a concrete construction of FSPS that yields shorter signatures in exchange of longer secret keys.

Regarding the constructions in the generic group model, there are several SPS schemes in the literature. Abe et al. [5] have shown that 3 element signatures cannot be proven secure under a non-interactive assumption using black box reductions, so strong assumptions are needed to get optimal efficiency. One of our schemes designed for more efficiency thus relies on the generic group model for its security proof. The signature

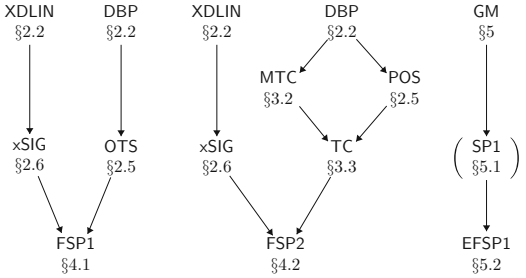

Fig. 1. Structure of our constructions.

scheme in [7] can be seen to be fully structure preserving with signatures consisting of 3 group elements. It is selectively randomizable where signatures are strong, but the signer can later choose to release a randomization token to make a signature randomizable. The notion of selective randomizability is different from the notion of combined signature schemes where the signer can choose to create randomizable or strong signatures at the time of signature generation. The advantage of selective randomizable signatures is that all signatures are verified with the same verification equation; the disadvantage is the need to issue randomization tokens when making a signature randomizable.

Regarding SPTC, the study by Abe et al.[8] is an important piece of context. It shows that no shrinking SPTC scheme can be binding, and indeed all existing SPTCs, e.g., [3], are expanding. The relaxed notion of binding, CMTCR, is a multi-session extension of honest-sender binding introduced in [28]. The use of trapdoor commitments and chameleon hashing has also been explored in the construction of online/off-line signatures [25,30].

Paper Organization In Sect. 2, we introduce notations and definitions used throughout the paper and review POS and xRMA-secure FSPS that will be used as building blocks. We then construct a shrinking SPTC scheme in Sect. 3 where we first present a new commitment scheme we call *a message-transposing commitment scheme* in Sect. 3.2, and use it to construct an CMTCR-secure SPTC in Sect. 3.3. We present FSPS schemes in Sect. 4. Starting from a simple construction in Sect. 4.1 that identifies problems, we present our main construction in Sect. 4.2. We then discuss their performance in Sect. 4.3 and a lower bound for signature and public key sizes in Sect. 6. In Sect. 5, we investigate more efficiency and functionality for FSPS in the generic group model. Starting from an efficient combined SPS in Sect. 5.1, we construct a combined FSPS in Sect. 5.2. In Fig. 1, we illustrate the structure of our construction. The upmost nodes are assumptions, and the bottom nodes are the FSPS schemes that we construct. Schemes in Sect. 5 are given security proofs in the generic bilinear group model (GM). We refer to [10] for variations of our FSPS constructions obtained by replacing some building blocks in our construction.

2. Preliminaries

2.1. *Notations*

We write λ for a security parameter given as input to all parties running a scheme. The intention is that we can strengthen the security of a scheme by increasing the security parameter. We say a function $f : \mathbb{N} \to [0, 1]$ is negligible when $f(\lambda) = \lambda^{-\omega(1)}$, and we say f is overwhelming when $1 - f(\lambda)$ is negligible.

We write $y \leftarrow A(x)$, when algorithm A takes x as input and outputs y. When it is clear from the context, we write $\boldsymbol{y} \leftarrow A(\boldsymbol{x})$ to denote sequential and independent executions of $y_i \leftarrow A(x_i)$ for $x_i \in \boldsymbol{x}$. When algorithm A is probabilistic, we let $A(x)$ denote the output distribution (or the set of outputs) of A with respect to input x. By $\Pr[\, A : X \,]$ we denote a probability that event X happens after process A is executed. When we count the number of source group elements, we use the notation (n_1, n_2) to represent n_1 and n_2 elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively.

We use asterisk $*$ as a wildcard that matches anything. For instance, $(a, *) \in X$ denotes that a set X includes a pair whose first item is a.

We will work extensively with cyclic groups, which we usually write with multiplicative notation. For a cyclic group $\mathbb{G}$, we define $\mathbb{G}^* = \mathbb{G} \backslash \{1_{\mathbb{G}}\}$. Or in the case of integers modulo p, which we write with additive notation, we define $\mathbb{Z}_p^* = \mathbb{Z}_p \backslash \{0\}$. We assume it is possible to sample elements uniformly at random and will for instance write $r \leftarrow \mathbb{Z}_p$ or $r \leftarrow \mathbb{Z}_p^*$ when sampling from these sets.

2.2. *Bilinear Groups*

Let $\mathcal{G}$ be a generator of bilinear groups that takes security parameter 1^λ as input and outputs $\Lambda := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G})$, where p is a λ-bit prime, $\mathbb{G}$, $\tilde{\mathbb{G}}$, $\mathbb{G}_T$ are groups of prime order p with efficiently computable group operations, membership tests, and bilinear map $e : \mathbb{G} \times \tilde{\mathbb{G}} \to \mathbb{G}_T$. The pairing operation e satisfies that $\forall A \in \mathbb{G}$, $\forall \tilde{B} \in \tilde{\mathbb{G}}$, $\forall x, y \in \mathbb{Z} : e(A^x, \tilde{B}^y) = e(A, \tilde{B})^{xy}$.

Elements G and $\tilde{G}$ are default random generators of $\mathbb{G}$, $\tilde{\mathbb{G}}$, and $e(G, \tilde{G})$ generates $\mathbb{G}_T$. We use multiplicative notation for group operations in $\mathbb{G}$, $\tilde{\mathbb{G}}$, and $\mathbb{G}_T$. An element in $\mathbb{G}$ is represented by a capital letter, e.g., $A \in \mathbb{G}$, and one in $\tilde{\mathbb{G}}$ is represented with tilde, e.g., $\tilde{B} \in \tilde{\mathbb{G}}$. And we often assign corresponding small case letters to represent the logarithm with respect to the default generator, e.g., $a = \log_G A$ and $b = \log_{\tilde{G}} \tilde{B}$. A vector of elements is possibly denoted by a bold letter. For a vector of scalar values $\boldsymbol{x} := (x_1, \ldots, x_n) \in \mathbb{Z}_p^n$ and a group element A, we write $A^{\boldsymbol{x}}$ for the vector $(A^{x_1}, \ldots, A^{x_n})$. Similarly, for a vector of group elements $\boldsymbol{X} := (X_1, \ldots, X_n)$ and a scalar value a, we write $\boldsymbol{X}^a$ for $(X_1^a, \ldots, X_n^a)$. Similar conventions apply to matrices of scalars and group elements.

An equation of the form $\prod_i \prod_j e(A_i, B_j)^{a_{ij}} = 1$ for constants $a_{ij} \in \mathbb{Z}_p$, and constants or variables $A_i \in \mathbb{G}$, $B_j \in \tilde{\mathbb{G}}$ is called a pairing product equation (PPE). By $\mathbb{G}^*$, we denote $\mathbb{G} \backslash 1_{\mathbb{G}}$, and similar for $\tilde{\mathbb{G}}^*$ and $\mathbb{Z}_p^*$.

2.2.1. *Generic Bilinear Group Model*

Let $(p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G})$ be a description of groups with a bilinear map. We refer to deciding group membership, computing group operations in $\mathbb{G}$, $\tilde{\mathbb{G}}$ or $\mathbb{G}_T$, comparing group elements, and evaluating the bilinear mapping $e : \mathbb{G} \times \tilde{\mathbb{G}} \to \mathbb{G}_T$ as *generic* group operations. We will sometimes restrict algorithms to only use generic operations. To enforce the use of generic group operations, Shoup [48] introduced the generic group model, where group elements are represented as random strings and group operations handled through an oracle that knows the corresponding real group elements. The generic group model has been adapted to the bilinear groups setting [18,19].

We will now describe the generic group model for the asymmetric setting where there is no isomorphism between the two source groups $\mathbb{G}$ and $\tilde{\mathbb{G}}$ that is efficiently computable in either direction. Each element in a group is represented by a unique string obtained by applying a random injective encoding to the group element. Algorithms may get some encoded elements as input and are given access to a group operation oracle that performs generic group operations. For random encodings π_1, π_2, and π_T assigned to $\mathbb{G}$, $\tilde{\mathbb{G}}$, and $\mathbb{G}_T$, respectively, the oracle is given encoded elements together with a generic group operation to apply to the inputs. For instance, it may be given $\pi_1(A)$ and $\pi_1(B)$ with an instruction to perform a group operation, and return $\pi_1(A \cdot B)$. Or it may be given $\pi_1(A)$ and $\pi_2(\tilde{B})$ with an instruction to perform a pairing operation, in which case it returns $\pi_T(e(A, \tilde{B}))$. With such an oracle, a (potentially adversarial) algorithm $\mathcal{A}$ can only compute a new encoding of an element in a source group $\pi_x(X)$ by using the generic group operations on already seen elements, if it has seen elements $\pi_x(X_i)$ it can pick scalars a_i and use the generic group operation oracle to get $\pi_x(X_i^{a_i})$. The algorithm could also pick a random element in the range of the encoding function; however, this would just give it a random group element, so it might as well pick a random scalar $r \leftarrow \mathbb{Z}_p$ and compute $\pi_x(X^r)$ for a previously seen $\pi_x(X)$. A common use of the generic group model is to build trust in an intractability assumption by proving that it cannot be broken by generic operations. In such proofs, we use that the encoding is random and therefore seeing for instance $\pi_x(X)$ reveal no information about the group element X itself except what can be deduced by using generic group operations and testing equality of group elements.

We call algorithms that follow the above model generic. Note that, when a generic algorithm outputs a group element in $\mathbb{G}$ (or $\tilde{\mathbb{G}}$), it only depends on elements in $\mathbb{G}$ (or $\tilde{\mathbb{G}}$, respectively) given to the algorithm as input.

2.2.2. *Assumptions*

Throughout the paper, we work over asymmetric bilinear groups (so-called Type-III setting [34]) where no efficient isomorphisms exist between $\mathbb{G}$ and $\tilde{\mathbb{G}}$. Some building blocks in our construction rely on the double pairing assumption [3].

Assumption 1. *(Double Pairing Assumption in $\tilde{\mathbb{G}}$:* DBP*) The double pairing assumption holds in $\tilde{\mathbb{G}}$ relative to $\mathcal{G}$ if, for all probabilistic polynomial time algorithms $\mathcal{A}$*

$$\Pr\left[\begin{array}{l} \Lambda \leftarrow \mathcal{G}(1^\lambda); \\ \tilde{G}_z \leftarrow \mathbb{G}_2^*; \\ (Z, R) \leftarrow \mathcal{A}(\Lambda, \tilde{G}_z) \end{array} : \begin{array}{l} (Z, R) \in \mathbb{G}_1^* \times \mathbb{G}_1^*, \wedge \\ 1 = e(Z, \tilde{G}_z)\, e(R, \tilde{G}) \end{array}\right] \tag{1}$$

is negligible in the security parameter λ.

The DBP assumption in $\mathbb{G}$ is defined by swapping $\mathbb{G}$ and $\tilde{\mathbb{G}}$ in the above definition. Note that the DBP assumption (in $\mathbb{G}$ and $\tilde{\mathbb{G}}$) is implied by the Decision Diffie–Hellman assumption [3] (in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively) which is often assumed in Type-III setting.

We also use a building block that requires more assumptions such as DDH_2, XDLIN_1, and $\mathsf{co\text{-}CDH}_2$ defined as follows.

Assumption 2. *(Decisional Diffie–Hellman Assumption in* $\tilde{\mathbb{G}}$*:* DDH_2*) Any probabilistic polynomial time algorithm* $\mathcal{A}$ *decides whether* $b = 1$ *or* 0 *with negligible advantage* $\mathrm{Adv}^{\mathsf{ddh2}}_{\mathcal{G},\mathcal{A}}(\lambda)$ *in* λ *given* $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, $\tilde{G} \leftarrow \tilde{\mathbb{G}}$, *and* $(\tilde{G}^x, \tilde{G}^y, Z_b)$ *where* $Z_1 = \tilde{G}^{xy}$ *and* $Z_0 = \tilde{G}^z$ *for random* $x, y, z \leftarrow \mathbb{Z}_p$ *and random bit* b,

Assumption 3. *(External Decision Linear Assumption in* $\mathbb{G}$*:* XDLIN_1*) Any probabilistic polynomial time algorithm* $\mathcal{A}$ *decides whether* $b = 1$ *or* 0 *with negligible advantage* $\mathrm{Adv}^{\mathsf{xdlin1}}_{\mathcal{G},\mathcal{A}}(\lambda)$ *given* $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, *and* $(G^a, G^b, G^c, G^{ax}, G^{by}, \tilde{G}^a, \tilde{G}^b, \tilde{G}^c, \tilde{G}^{ax}, \tilde{G}^{by}, Z_b)$ *where* $Z_1 = G^{c(x+y)}$, *and* $Z_0 = G^z$ *for random* $a, b, c \leftarrow \mathbb{Z}_p^*$, $x, y, z \leftarrow \mathbb{Z}_p$ *and random bit* b.

The XDLIN_1 assumption is equivalent to the DLIN_1 assumption in the generic bilinear group model where one can simulate the extra elements, $\tilde{G}^a, \tilde{G}^b, \tilde{G}^c, \tilde{G}^{ax}, \tilde{G}^{by}$, in XDLIN_1 from $G^a, G^b, G^c, G^{ax}, G^{by}$ in DLIN_1.

Assumption 4. *(Computational co-Diffie–Hellman Assumption in* $\tilde{\mathbb{G}}$*:* $\mathsf{co\text{-}CDH}_2$*) Any probabilistic polynomial time algorithm* $\mathcal{A}$ *outputs* $\tilde{G}^{xy}$ *with negligible probability* $\mathrm{Adv}^{\mathsf{co\text{-}cdh}}_{\mathcal{G},\mathcal{A}}(\lambda)$ *given* $\Lambda \leftarrow \mathcal{G}(1^\lambda)$, $G \leftarrow \mathbb{G}^*$, $\tilde{G} \leftarrow \tilde{\mathbb{G}}^*$, G^x, G^y, $\tilde{G}^x$, *and* $\tilde{G}^y$ *for* $x, y \leftarrow \mathbb{Z}_p$.

Similar to Assumption 3, $\mathsf{co\text{-}CDH}_2$ is equivalent to computational Diffie–Hellman assumption in $\mathbb{G}$ or $\tilde{\mathbb{G}}$ in the generic bilinear group model, but generally they are unrelated in Type-III setting. We refer [49] for more discussion on variations of computational Diffie–Hellman assumptions over bilinear groups.

2.3. *Joint Setup*

Building blocks in this paper are defined with individual setup functions. As we work over bilinear groups, an output from a setup function should include a description of bilinear groups Λ. Some random generators specific to each building block may be included as well.

The idea behind structure-preserving cryptography is that all schemes will work over the same bilinear group and hence be easy to compose. We will therefore for composed schemes assume they use a joint setup consisting of a bilinear group and a number of

random group elements corresponding to the maximum number needed by any of the building blocks. More precisely, suppose that two building blocks, say A and B, are used together and have setups $gk_{\mathsf{A}} \leftarrow \mathsf{A.Setup}(1^\lambda)$ and $gk_{\mathsf{B}} \leftarrow \mathsf{B.Setup}(1^\lambda)$. We say they have a common setup function if there exists a polynomial time algorithm Setup such that, given $gk \leftarrow \mathsf{Setup}(1^\lambda)$ it is possible in polynomial time from gk to recover individual setups gk_{A} and gk_{B}, each one having correct probability distribution. It is also required that gk can be simulated given either gk_{A} or gk_{B}. In the rest of the paper, we assume a common setup gk for all the individual schemes. In general each individual setup samples a bilinear group and a number of uniformly random group elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$. We can then pick as a common setup a bilinear group sampled with the same distribution and a number of uniformly random elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$ matching the maximum number any individual schemes uses in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively.

2.4. *Digital Signatures*

In this section, we recall definitions of digital signatures and their security notions. On top of the standard notions, we define structure-preserving and fully structure-preserving signatures.

Definition 1. (*Digital Signature Scheme*) A digital signature scheme SIG is a tuple of polynomial time algorithms $(\mathsf{Setup}, \mathsf{Key}, \mathsf{Sign}, \mathsf{Vrf})$ where $gk \leftarrow \mathsf{Setup}(1^\lambda)$ is a probabilistic setup algorithm that, given a security parameter λ, generates common parameter gk, which defines a message space $\mathcal{M}$ for which membership is efficiently decidable, $(vk, sk) \leftarrow \mathsf{Key}(gk)$ is a probabilistic key generation algorithm that takes common parameter gk and generates a verification key vk and a signing key sk, $\sigma \leftarrow \mathsf{Sign}(sk, m)$ is a probabilistic signature generation algorithm that computes a signature σ for input message $m \in \mathcal{M}$ by using signing key sk, and $1/0 \leftarrow \mathsf{Vrf}(vk, m, \sigma)$ is a verification algorithm that outputs 1 for acceptance or 0 for rejection according to the input.

For any legitimately generated gk, vk, sk and any $m \in \mathcal{M}$, it must hold that $1 = \mathsf{Vrf}(vk, m, \mathsf{Sign}(sk, m))$. A key pair (vk, sk) is valid with respect to gk if it is in the range of $\mathsf{Key}(gk)$.

Definition 2. (*Unforgeability against Adaptive Chosen-Message Attacks*) A signature scheme, $\mathsf{SIG} = \{\mathsf{Setup}, \mathsf{Key}, \mathsf{Sign}, \mathsf{Vrf}\}$, is unforgeable against adaptive chosen-message attacks ($\mathsf{UF\text{-}CMA}$) if for any polynomial time adversary $\mathcal{A}$ the following advantage function is negligible.

$$\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l|ll} & & m^\dagger \notin Q \wedge \\ gk \leftarrow \mathsf{Setup}(1^\lambda), & : & 1 = \mathsf{Vrf}(vk, m^\dagger, \sigma^\dagger) \\ (vk, sk) \leftarrow \mathsf{Key}(gk), & & \\ (\sigma^\dagger, m^\dagger) \leftarrow \mathcal{A}^{\mathcal{O}_{sk}}(vk) & & \end{array}\right], \quad (2)$$

where $\mathcal{O}_{sk}$ is an oracle that, given m, executes $\sigma \leftarrow \mathsf{Sign}(sk, m)$, records m in Q, and returns σ.

It is strongly unforgeable if $\mathcal{O}_{sk}$ records (m, σ) in Q and the winning condition $m^\dagger \notin Q$ is replaced with $(m^\dagger, \sigma^\dagger) \notin Q$.

By UF-NACMA we denote a relaxed notion of security where adversary $\mathcal{A}$ has to commit to the messages to query before seeing vk. ($\mathcal{A}$ is given gk that defines the message space.) Such an attack model is called a weak chosen-message attack in [18], and a generic chosen-message attack in [35]. Formally:

Definition 3. (*Unforgeability against Non-Adaptive Chosen-Message Attacks*) A signature scheme, $\mathsf{SIG} = \{\mathsf{Setup}, \mathsf{Key}, \mathsf{Sign}, \mathsf{Vrf}\}$, is unforgeable against non-adaptive chosen-message attacks (UF-NACMA) if for any polynomial time adversary $\mathcal{A}$ the following advantage function is negligible.

$$\mathrm{Adv}^{\text{uf-nacma}}_{\mathsf{SIG},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda), \\ (\boldsymbol{m_i}, \omega) \leftarrow \mathcal{A}(gk) \\ (vk, sk) \leftarrow \mathsf{Key}(gk), \\ \boldsymbol{\sigma_i} \leftarrow \mathsf{Sign}(sk, \boldsymbol{m_i}) \\ (\sigma^\dagger, m^\dagger) \leftarrow \mathcal{A}(\omega, vk, \boldsymbol{\sigma_i}) \end{array} : \begin{array}{l} m^\dagger \notin \boldsymbol{m_i} \wedge \\ 1 = \mathsf{Vrf}(vk, m^\dagger, \sigma^\dagger) \end{array}\right], \quad (3)$$

where ω is an internal state, $\boldsymbol{m_i}$ is a polynomial number of messages, and $\boldsymbol{\sigma_i} \leftarrow \mathsf{Sign}(sk, \boldsymbol{m_i})$ is a process of signing each m_i with sk and outputting the resulting signature as σ_i.

A one-time signature scheme is a digital signature scheme with the limitation that a signing key is intended to be used only once. Unforgeability against one-time chosen-message attacks is defined as in Definition 2 by restricting the signing oracle to answer only a single query.

Definition 4. (*Structure-Preserving Signature Scheme*) A digital signature scheme is structure preserving relative to bilinear group generator $\mathcal{G}$ if the common parameters gk consist of group description Λ generated by $\mathcal{G}$, some constants, and some source group elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$ in Λ, and verification keys vk, messages m and signatures σ solely consist of group elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, and the verification algorithm Vrf consists only of evaluating membership in $\mathbb{G}$ and $\tilde{\mathbb{G}}$ and relations described by paring product equations.

When messages consist of elements from both source groups, $\mathbb{G}$ and $\tilde{\mathbb{G}}$, they are called bilateral. We say a message is unilateral if it consists exclusively of elements from one of the source groups.

The notion of structure-preserving cryptography requires *public* components to be group elements. We extend it so that *private* components consist of group elements as well.

Definition 5. (*Fully Structure-Preserving Signature Scheme*) A structure-preserving signature scheme is fully structure preserving if signing keys sk consist of group elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, and the validity of sk with respect to vk can be verified by evaluating membership in $\mathbb{G}$ and $\tilde{\mathbb{G}}$ and relations described by pairing product equations.

Note that, in reality, vk will include gk and sk will include vk to be consistent with the standard interface of the functions in a signature scheme. We will ignore those nested

objects when we argue that a scheme is structure preserving and measure the size of keys and signature without counting the elements in gk.

Once the additional conditions in Definition 5 are satisfied, one can construct proofs of knowledge of secret keys by using the Groth–Sahai proof system, which allows the extraction of a correct secret key corresponding to the verification key. It is, however, important to note that there could exist more than one valid secret key for a verification key and different secret keys may yield signatures with different distributions. One might need stronger extractability that enables the extraction of the secret key for a particular distribution of signatures. This is for instance the case for the group signature application mentioned in Sect. 1. Our concrete scheme allows one to efficiently prove the relation between a secret key and a signature. See Sect. 2.6.

The notion of standard unforgeability does not prevent the adversary from changing signatures as long as the associated message is intact. Constructive use of this property is known as randomizable signature schemes defined as follows.

Definition 6. (*Randomizable Signature Scheme*) A signature scheme is randomizable if there exists an efficient algorithm Rand that takes gk, vk, m, and σ as input and outputs a new signature σ'. We require for all $\lambda \in \mathbb{N}$, $gk \leftarrow \mathsf{Setup}(1^\lambda)$, $(vk, sk) \leftarrow \mathsf{Key}(gk)$, $m \in \mathcal{M}$, $\sigma \leftarrow \mathsf{Sign}(sk, m)$, and all randomized signatures $\sigma' \leftarrow \mathsf{Rand}(vk, m, \sigma)$, it must hold that $1 \leftarrow \mathsf{Vrf}(vk, m, \sigma')$.

Signatures are perfectly randomizable if a randomized signature looks exactly like a fresh signature on the same message.

Definition 7. (*Perfect Randomizability*) A signature scheme is perfectly randomizable if for all adversaries $\mathcal{A}$ outputting messages $m \in \mathcal{M}$ we have

$$\Pr\left[\begin{array}{c} gk \leftarrow \mathsf{Setup}(1^\lambda);\ (vk, sk) \leftarrow \mathsf{Key}(gk);\ b \leftarrow \{0,1\}; \\ m \leftarrow \mathcal{A}(vk, sk);\ \sigma_0 \leftarrow \mathsf{Sign}(sk, m);\ \sigma_1 \leftarrow \mathsf{Rand}(vk, m, \sigma_0) \end{array} : \mathcal{A}(\sigma_b) = b\right] = \frac{1}{2}.$$

Unless a signature scheme is deterministic, it cannot be both perfectly randomizable and also strongly unforgeable. However, we can combine both properties in a single signature scheme that allows a signer to issue two types of signatures: randomizable signatures and strongly unforgeable signatures. We call such a scheme a *combined signature scheme*.

Definition 8. (*Combined Signature Scheme*) A combined signature scheme is a set of polynomial time algorithms $(\mathsf{Setup}, \mathsf{Key}, \mathsf{Sign}_0, \mathsf{Vrf}_0, \mathsf{Rand}, \mathsf{Sign}_1, \mathsf{Vrf}_1)$ where $(\mathsf{Setup}, \mathsf{Key}, \mathsf{Sign}_0, \mathsf{Vrf}_0, \mathsf{Rand})$ is a randomizable signature scheme and $(\mathsf{Setup}, \mathsf{Key}, \mathsf{Sign}_1, \mathsf{Vrf}_1)$ is a strongly unforgeable signature scheme.

A naïve combined signature scheme would have a verification key containing two verification keys, one for randomizable signatures and one for strong signatures. However, this solution has the disadvantage of increasing key size. Instead, we will in this paper construct a combined signature scheme where the verification key is just a single group element that can be used to verify either type of signature. This dual use of the verifica-

tion key means that we must carefully consider the security implications of combining two signature schemes though, so we will now define a combined signature scheme.

To capture the attacks that can occur against a combined signature scheme, we assume the adversary may arbitrarily query a signer for randomizable or strong signatures. We want the signature scheme to be combined existentially unforgeable in the sense that even seeing randomizable signatures does not help in breaking strong existential unforgeability and on the other hand seeing strong signatures does not help in producing randomizable signatures.

Definition 9. (*Combined Existential Unforgeability Under Chosen-Message Attack*) The combined signature scheme is combined existentially unforgeable under adaptive chosen-message attack (C-EUF-CMA) if for all probabilistic polynomial time adversaries $\mathcal{A}$

$$\Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda);\ (vk, sk) \leftarrow \mathsf{Key}(gk) \\ (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}_0(sk,\cdot),\mathsf{Sign}_1(sk,\cdot)}(vk) \end{array} : \begin{array}{l} \mathsf{Vrf}_0(vk, m, \sigma) = 1 \wedge m \notin Q_0 \ \text{ or} \\ \mathsf{Vrf}_1(vk, m, \sigma) = 1 \wedge (m, \sigma) \notin Q_1 \end{array}\right]$$

is negligible, where $\mathcal{A}$ can make signing queries on arbitrary $m \in \mathcal{M}$ and the output message m must belong to $\mathcal{M}$, Q_0 is the set of messages that have been queried to Sign_0, and Q_1 is the set of message and signature pairs from queries to Sign_1.

2.5. *Partially One-Time Signatures*

When only a part of a signing key of one-time signatures must be updated for every signing, i.e., the remaining part of the key can be used an unbounded number of times, the scheme is called a two-tier signature scheme or a partially one-time signature scheme[2, 16].

Definition 10. (*Partially One-time Signature Scheme*) A partially one-time signature scheme is a set of algorithms $\mathsf{POS} = \{\mathsf{Setup}, \mathsf{Key}, \mathsf{Ovk}, \mathsf{Sign}, \mathsf{Vrf}\}$ such that

- $\mathsf{Setup}(1^\lambda) \rightarrow gk$: A setup function that, given a security parameter λ, generates common parameter gk, which defines message space $\mathcal{M}$.
- $\mathsf{Key}(\mathrm{gk}) \rightarrow (vk, sk)$: A long-term key generation function that takes gk and outputs a long-term key pair (vk, sk).
- $\mathsf{Ovk}(\mathrm{gk}) \rightarrow (ovk, osk)$: A one-time key generation function that takes gk and outputs a one-time key pair (ovk, osk).
- $\mathsf{Sign}(sk, osk, m) \rightarrow \sigma$: A signing function that takes sk, osk and a message m as inputs and issues a signature σ.
- $\mathsf{Vrf}(vk, ovk, m, \sigma) \rightarrow 1/0$: A verification function that outputs 1 or 0 for acceptance and rejection, respectively.

For any $gk \leftarrow \mathsf{Setup}(1^\lambda)$, $(vk, sk) \leftarrow \mathsf{Key}(gk)$, $m \in \mathcal{M}$, and $(ovk, osk) \leftarrow \mathsf{Ovk}(gk)$, $\sigma \leftarrow \mathsf{Sign}(sk, osk, m)$, it must hold that $1 \leftarrow \mathsf{Vrf}(vk, ovk, m, \sigma)$.

The security notion considered in [2,16] is defined with respect to a single long-term key pair. Here we extend the notion to multiple key pairs.

Definition 11. (*Multi-Key Partial One-time Chosen-Message Attack for POS*) A partially one-time signature scheme, $\mathsf{POS} = \{\mathsf{Setup}, \mathsf{Key}, \mathsf{Ovk}, \mathsf{Sign}, \mathsf{Vrf}\}$, is unforgeable against multi-key non-adaptive partial one-time chosen-message attacks ($\mathsf{MK\text{-}OT\text{-}NACMA}$), if for any polynomial time adversary $\mathcal{A}$ the advantage function $\mathrm{Adv}^{\mathsf{mk\text{-}ot\text{-}nacma}}_{\mathsf{POS},\mathcal{A}}(\lambda)$ defined by

$$\Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda), \\ (ovk^\dagger, \sigma^\dagger, m^\dagger) \leftarrow \mathcal{A}^{\mathcal{O}_k, \mathcal{O}_s}(gk) \end{array} : \begin{array}{l} (vk^\dagger, ovk^\dagger, *) \in Q_{mv} \wedge \\ (vk^\dagger, ovk^\dagger, m^\dagger) \notin Q_{mv} \wedge \\ 1 = \mathsf{Vrf}(vk^\dagger, ovk^\dagger, m^\dagger, \sigma^\dagger) \end{array}\right] \tag{4}$$

is negligible. Oracle $\mathcal{O}_k$ is the key generation oracle that, on receiving the i-th request from $\mathcal{A}$, generates $(vk^{[i]}, sk^{[i]}) \leftarrow \mathsf{Key}(gk)$, and returns $vk^{[i]}$. Oracle $\mathcal{O}_s$ is the signing oracle that, given $m \in \mathcal{M}$ and $vk^{[i]}$ generated by $\mathcal{O}_k$, executes $(ovk^{(j)}, osk^{(j)}) \leftarrow \mathsf{Ovk}(gk)$, $\sigma \leftarrow \mathsf{Sign}(sk^{[i]}, osk^{(j)}, m)$, records $(vk^{[i]}, ovk^{(j)}, m)$ in Q_{mv}, and returns $(\sigma, ovk^{(j)})$.

The following concrete scheme is taken from [2] with a trivial modification in the signing algorithm so that the signature elements are computed more efficiently.

[**Partially One-time Signature Scheme:** POS]

$\mathsf{Setup}(1^\lambda)$: Run $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G}) \leftarrow \mathcal{G}(1^\lambda)$. Set message space $\mathcal{M}$ to $\tilde{\mathbb{G}}^{\ell_{\mathsf{pos}}}$ for a fixed positive integer ℓ_{pos}.

$\mathsf{Key}(gk)$: Take generators G and $\tilde{G}$ from gk. Choose w_z randomly from $\mathbb{Z}_p^*$, and compute $G_z := G^{w_z}$. For $i = 1, \ldots, \ell_{\mathsf{pos}}$, uniformly choose χ_i from $\mathbb{Z}_p$ and compute $G_i := G^{\chi_i}$. Output $vk := (G_z, G_1, \ldots, G_{\ell_{\mathsf{pos}}}) \in \mathbb{G}_1^{\ell_{\mathsf{pos}}+1}$ and $sk := (\chi_1, \ldots, \chi_{\ell_{\mathsf{pos}}}, w_z)$.

$\mathsf{Ovk}(gk)$: Choose $a \leftarrow \mathbb{Z}_p$ and output $ovk = A := G^a$, and $osk := a$.

$\mathsf{Sign}(sk, osk, m)$: Parse m into $(\tilde{M}_1, \cdots, \tilde{M}_{\ell_{\mathsf{pos}}}) \in \tilde{\mathbb{G}}^{\ell_{\mathsf{pos}}}$. Take a and w_z from osk and sk, respectively. Choose ζ randomly from $\mathbb{Z}_p^*$ and compute the signature as $(\tilde{Z}, \tilde{R})$ where $\tilde{Z} = \tilde{G}^\zeta$, $\tilde{R} = \tilde{G}^{a-\zeta\, w_z} \prod_{i=1}^{\ell_{\mathsf{pos}}} \tilde{M}_i^{-\chi_i}$.

$\mathsf{Vrf}(vk, ovk, m, \sigma)$: Parse σ as $(\tilde{Z}, \tilde{R}) \in \tilde{\mathbb{G}}^2$, m as $(\tilde{M}_1, \ldots, \tilde{M}_{\ell_{\mathsf{pos}}}) \in \tilde{\mathbb{G}}^{\ell_{\mathsf{pos}}}$, and ovk as A. Return 1, if $e(A, \tilde{G}) = e(G_z, \tilde{Z})\, e(G, \tilde{R}) \prod_{i=1}^{\ell_{\mathsf{pos}}} e(G_i, \tilde{M}_i)$ holds. Return 0, otherwise.

In [2], the security of the above scheme is proven based on the DBP assumption in $\mathbb{G}$ with respect to a single long-term key, i.e., under the constraint that $\mathcal{O}_k$ is accessible only once. However, it is easy to show that the scheme is indeed $\mathsf{MK\text{-}OT\text{-}NACMA}$ as stated below thanks to the random self-reducibility of the DBP problem. (The scheme satisfies even stronger security where $\mathcal{A}$ is allowed to access Ovk and Sign separately.)

Theorem 1. POS *is strongly unforgeable against* $\mathsf{MK\text{-}OT\text{-}NACMA}$ *if* DBP *in* $\mathbb{G}$ *holds. In particular, for all p.p.t. algorithms* $\mathcal{A}$ *there exists a p.p.t. algorithm* $\mathcal{B}$ *such that* $\mathrm{Adv}^{\mathsf{mk\text{-}ot\text{-}nacma}}_{\mathsf{POS},\mathcal{A}}(\lambda) \leq \mathrm{Adv}_{\mathsf{DBP},\mathcal{B}}(\lambda) + 1/p(\lambda)$, *where* $p(\lambda)$ *is the size of the groups*

produced by $\mathcal{G}$. Moreover, the run-time overhead of the reduction $\mathcal{B}$ is a small number of multi-exponentiations per signing or key query.

2.6. *xRMA-Secure Fully Structure-Preserving Signature Scheme*

We follow the notion of extended random message attacks and take one of the schemes in [2]. The definition is relative to a message sampling algorithm, $\mathsf{SampleM}$, that takes gk and outputs messages $\boldsymbol{m}$ with some auxiliary information ω.

Definition 12. (*Unforgeability against Extended Random Message Attacks*) A signature scheme, $\mathsf{xSIG} = \{\mathsf{Setup}, \mathsf{Key}, \mathsf{Sign}, \mathsf{Vrf}\}$, is unforgeable against extended random message attacks ($\mathsf{UF\text{-}XRMA}$) with respect to message sampling algorithm $\mathsf{SampleM}$ if for any polynomial time adversary $\mathcal{A}$

$$\mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{xSIG},\mathcal{A}}(\lambda) := \Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda), \\ (vk, sk) \leftarrow \mathsf{Key}(gk), \\ (\boldsymbol{m}, \omega) \leftarrow \mathsf{SampleM}(gk), \\ \boldsymbol{\sigma} \leftarrow \mathsf{Sign}(sk, \boldsymbol{m}), \\ (\sigma^\dagger, m^\dagger) \leftarrow \mathcal{A}(vk, \boldsymbol{\sigma}, \boldsymbol{m}, \omega) \end{array} : \begin{array}{l} m^\dagger \notin \boldsymbol{m} \wedge \\ 1 = \mathsf{Vrf}(vk, m^\dagger, \sigma^\dagger) \end{array}\right] \tag{5}$$

is negligible.

The next scheme is taken from [2] with two modifications to fit to our construction. First, the message space is extended to sign messages consisting of $\ell_{\mathsf{x}} \geq 1$ message blocks. Second, it takes randomness from $\mathbb{Z}_p$ rather than $\mathbb{Z}_p^*$ in the key generation. Those differences make no changes in their security properties.

[**Signature Scheme:** xSIG]

$\mathsf{Setup}(1^\lambda)$: Run $(p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G}) \leftarrow \mathcal{G}(1^\lambda)$. For some fixed $\ell_{\mathsf{x}} \geq 1$, choose $u_1, \cdots, u_{\ell_{\mathsf{x}}}, \varrho, \delta$ randomly from $\mathbb{Z}_p^*$ and compute $F_1 := G^\varrho$, $F_2 := G^\delta$, $\tilde{F}_1 := \tilde{G}^\varrho$, $\tilde{F}_2 := \tilde{G}^\delta$, $U_i := G^{u_i}$, and $\tilde{U}_i := \tilde{G}^{u_i}$. Output $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G}, F_1, F_2, \tilde{F}_1, \tilde{F}_2, \{U_i, \tilde{U}_i\}_{i=1}^{\ell_{\mathsf{x}}})$. This defines the message space $\mathcal{M} = \{(\tilde{M}_{11}, \tilde{M}_{12}, \tilde{M}_{13}), \ldots, (\tilde{M}_{\ell_{\mathsf{x}}1}, \tilde{M}_{\ell_{\mathsf{x}}2}, \tilde{M}_{\ell_{\mathsf{x}}3}) \mid {}^\forall i, {}^\exists m_i \in \mathbb{Z}_p \text{ s.t. } (\tilde{M}_{i1}, \tilde{M}_{i2}, \tilde{M}_{i3}) = (\tilde{F}_1^{m_i}, \tilde{F}_2^{m_i}, \tilde{U}_i^{m_i})\}$.

$\mathsf{Key}(gk)$: On input gk, choose $\tau_1, \tau_2, \tau_3, \rho, a, b, \alpha$ from $\mathbb{Z}_p$, and compute

$$\begin{array}{llll} \tilde{V}_1 := \tilde{G}^b, & \tilde{V}_2 := \tilde{G}^a, & \tilde{V}_3 := \tilde{G}^{ba}, & \tilde{V}_4 := \tilde{G}^{\tau_1 + a\tau_2}, \\ \tilde{V}_5 := \tilde{V}_4^b, & \tilde{V}_6 := \tilde{G}^{\tau_3}, & V_7 := G^\rho, & \tilde{V}_8 := \tilde{G}^{\alpha b/\rho}, \\ K_1 := G^\alpha, & K_2 := G^b, & K_3 := G^{\tau_1}, & K_4 := G^{\tau_2}. \end{array} \tag{6}$$

(For completeness of description, pick $\tilde{V}_8$ uniformly from $\tilde{\mathbb{G}}$ if $\rho = 0$.) Output $vk := (gk, \tilde{V}_1, \tilde{V}_2, \tilde{V}_3, \tilde{V}_4, \tilde{V}_5, \tilde{V}_6, V_7, \tilde{V}_8)$ and $sk := (vk, K_1, K_2, K_3, K_4)$.

$\mathsf{Sign}(sk, M)$: Parse message M into $\{(\tilde{M}_{11}, \tilde{M}_{12}, \tilde{M}_{13}), \cdots, (\tilde{M}_{\ell_{\mathsf{x}}1}, \tilde{M}_{\ell_{\mathsf{x}}2}, \tilde{M}_{\ell_{\mathsf{x}}3})\} \in \mathcal{M}$. Select $r_1, r_2, z \leftarrow \mathbb{Z}_p$, set $r := r_1 + r_2$, compute

$$\begin{array}{lll} \tilde{S}_0 := (\tilde{V}_6 \prod_{i=1}^{\ell_{\mathsf{x}}} \tilde{M}_{i3})^{r_1}, & S_1 := K_1 K_3{}^r, & S_2 := K_4{}^r G^{-z}, \\ S_3 := K_2{}^z, & S_4 := K_2{}^{r_2}, & S_5 := G^{r_1}. \end{array} \tag{7}$$

Output $\sigma := (\tilde{S}_0, \ldots, S_5) \in \tilde{\mathbb{G}} \times \mathbb{G}^5$.

$\mathsf{Vrf}(vk, M, \sigma)$: Output 1 if the following relations hold:

$$e\left(S_5, \tilde{V}_6 \prod_{i=1}^{\ell_{\mathsf{x}}} \tilde{M}_{i3}\right) = e(G, \tilde{S}_0), \tag{8}$$

$$e(S_1, \tilde{V}_1) e(S_2, \tilde{V}_3) e(S_3, \tilde{V}_2) = e(S_4, \tilde{V}_4) e(S_5, \tilde{V}_5) e(V_7, \tilde{V}_8),$$

$$e(F_1, \tilde{M}_{i3}) = e(U_i, \tilde{M}_{i1}), \quad e(F_2, \tilde{M}_{i3}) = e(U_i, \tilde{M}_{i2}) \quad \text{for } i = 1, \cdots, \ell_{\mathsf{x}}.$$

Output 0, otherwise.

The above scheme comes with trivial modifications from the original in [2]. First it is extended to sign random messages consisting of $\ell_{\mathsf{x}} \geq 1$ message blocks, and second it takes randomness from $\mathbb{Z}_p$ rather than $\mathbb{Z}_p^*$ in the key generation. Those changes have no effect on the security that we recall below.

Theorem 2. ([2]) *If the* DDH_2, XDLIN_1, *and* $\mathsf{co\text{-}CDH}_2$ *assumptions hold, then the above* xSIG *is* UF-XRMA *with respect to any message sampling algorithm that takes gk as input and returns message block* $(\tilde{F}_1^{m_i}, \tilde{F}_2^{m_i}, \tilde{U}_i^{m_i})$ *with auxiliary information* m_i *for* $i = 1, \ldots, \ell_{\mathsf{x}}$.

Theorem 3. *The above* xSIG *is fully structure preserving.*

Proof. By inspection, it is clear that vk (modulo group description in gk), sk, M, and σ consist of source group elements, and xSIG.Vrf consists of evaluating PPEs.

Next we show that the following PPEs are satisfied if and only if the verification key and the secret key are in the range of xSIG.Key.

$$\begin{array}{l} e(K_2, \tilde{G}) = e(G, \tilde{V}_1), \quad e(G, \tilde{V}_3) = e(K_2, \tilde{V}_2), \quad e(K_1, \tilde{V}_1) = e(V_7, \tilde{V}_8), \\ e(K_2, \tilde{V}_4) = e(G, \tilde{V}_5), \quad e(K_3, \tilde{G})\, e(K_4, \tilde{V}_2) = e(G, \tilde{V}_4). \end{array} \tag{9}$$

Showing correctly generated keys satisfy the above relations is trivial. We argue the other direction as follows. Variables that define a key pair are a, b, α, τ_1, τ_2, τ_3 and ρ. They are uniquely determined by $\tilde{V}_2$, $\tilde{V}_1$, K_1, K_3, K_4, $\tilde{V}_6$, and V_7, respectively. We verify that the remaining $\tilde{V}_3$, $\tilde{V}_4$, $\tilde{V}_5$, $\tilde{V}_8$, and K_2 are in the support of the correct distribution if the above relations are satisfied. The first equation is $e(K_2, \tilde{G}) = e(G, \tilde{G})^b$ that defines $K_2 = G^b$. The second equation is $e(G, \tilde{V}_3) = e(G, \tilde{G})^{ba}$ that defines $\tilde{V}_3 = \tilde{G}^{ba}$. The third equation is $e(G, \tilde{G})^{\alpha b} = e(G, \tilde{V}_8)^{\rho}$ that defines $\tilde{V}_8 = \tilde{G}^{\alpha b/\rho}$ for $\rho \neq 0$. If $\rho = 0$, $\tilde{V}_8$ can be an arbitrary value as described in the key generation. The

fourth equation is $e(G, \tilde{V}_4)^b = e(G, \tilde{V}_5)$ that defines $\tilde{V}_5 = \tilde{V}_4^b$. The last equation is $e(G, \tilde{G})^{\tau_1 + a\tau_2} = e(G, \tilde{V}_4)$ that defines $\tilde{V}_4 = \tilde{G}^{\tau_1 + a\tau_2}$ as prescribed. □

Proving a Correct Secret Key for a Signature In the above xSIG, there are several secret keys for a verification key, and each secret key yields signatures with a different distribution. It is possible to efficiently prove one's possession of a secret key used to create the signature in question by proving the following relation.

$$\begin{aligned} &e(\underline{K_2}, \underline{\tilde{G}^r}) = e(S_4, \tilde{G})e(S_5, \tilde{V}_1), \quad e(S_1, \tilde{G}) = e(\underline{K_1}, \tilde{G})\, e(\underline{K_3}, \underline{\tilde{G}^r}), \\ &e(S_3, \tilde{G}) = e(\underline{G^z}, \tilde{V}_1), \qquad\qquad e(S_2, \tilde{G})\, e(\underline{G^z}, \tilde{G}) = e(\underline{K_4}, \underline{\tilde{G}^r}) \end{aligned} \tag{10}$$

Here, z and r are the random coins used for the signature. A Groth–Sahai zero-knowledge proof for the underlined group elements as witnesses can be constructed using techniques from [20,29].

Consider a verification key and a signature that satisfy the verification equations in (8) and a secret key that satisfies (9) with respect to the verification key. Suppose that they also satisfy the equations in (10). Define r_1 and r_2 by $r_1 = \log_G S_5$ and $r_2 = \log_{K_2} S_4$. Parameter b is defined by $b = \log_G K_2 = \log_{\tilde{G}} \tilde{V}_1$. In the exponent, the first relation in (10) is read as $br = br_2 + br_1$ that ensures correctness of G^r with respect to S_4 and S_5. The second relation in (10) then guarantees $S_1 = K_1 K_3^r$ for this r, K_1, and K_3. The third relation in (10) proves that $S_3 = G^{z \log_{\tilde{G}} \tilde{V}_1} = G^{zb} = {K_2}^z$ for some z determined by G^z. Finally, the last relation in (10) is for $S_2 = {K_4}^r \tilde{G}^{-z}$. Thus, the secret key fulfilling all relations in (10) satisfies relations in (7) with respect to the signature and the verification key. Namely, the secret key is the one used to create the signature.

3. Trapdoor Commitment Schemes

In this section, we construct a structure-preserving shrinking trapdoor commitment scheme as defined in Sect. 3.1. We first construct a commitment scheme in Sect. 3.2 that is almost structure preserving in the sense that the messages for computing commitments are not group elements but scalar values. This slight relaxation allows to implement the shrinking property by using the one-way nature from the scalar values to group elements. Then a complete scheme is constructed in Sect. 3.3 by combining the building block from Sect. 3.2 with a one-time structure-preserving signature scheme that actually binds messages to commitments.

3.1. *Definitions*

We adopt the following standard syntax for trapdoor commitment schemes.

Definition 13. (*Trapdoor Commitment Scheme*) A trapdoor commitment scheme TC is a tuple of polynomial time algorithms TC = {Setup, Key, Com, Vrf, SimCom, Equiv} that:

$\mathsf{Setup}(1^\lambda) \to gk$: A common parameter generation algorithm that takes security parameter λ and outputs a common parameter, gk. It determines the message space $\mathcal{M}$, the commitment space $\mathcal{C}$, and opening space $\mathcal{I}$.

$\mathsf{Key}(gk) \to (ck, tk)$: A key generation algorithm that takes gk as input and outputs a commitment key, ck, and a trapdoor key, tk.

$\mathsf{Com}(ck, m) \to (com, open)$: A commitment algorithm that takes ck and message $m \in \mathcal{M}$ and outputs a commitment, $com \in \mathcal{C}$, and opening information, $open \in \mathcal{I}$.

$\mathsf{Vrf}(ck, com, m, open) \to 1/0$: A verification algorithm that takes ck, com, m, and $open$ as input and outputs 1 or 0 representing acceptance or rejection, respectively.

$\mathsf{SimCom}(gk) \to (com, ek)$: A sampling algorithm that takes common parameter gk and outputs commitment com and equivocation key ek.

$\mathsf{Equiv}(m, ek, tk) \to open$: An algorithm that takes ck, ek, tk and $m \in \mathcal{M}$ as input and returns $open$.

The trapdoor commitment scheme is correct if, for all $\lambda \in \mathbb{N}$, $gk \leftarrow \mathsf{Setup}(1^\lambda)$, $(ck, tk) \leftarrow \mathsf{Key}(gk)$, $m \leftarrow \mathcal{M}$, $(com, open) \leftarrow \mathsf{Com}(ck, m)$, it holds that $1 = \mathsf{Vrf}(ck, com, m, open)$. Furthermore, it is statistical trapdoor if for any unbounded stateful adversary $\mathcal{A}$ outputting $m \in \mathcal{M}$

$$\Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda);\ (ck, tk) \leftarrow \mathsf{Key}(gk);\ m \leftarrow \mathcal{A}(ck, tk) \\ \quad (com_0, open_0) \leftarrow \mathsf{Com}(ck, m);\ b \leftarrow \{0, 1\} \qquad : \mathcal{A}(com_b, open_b) = b \\ (com_1, ek) \leftarrow \mathsf{SimCom}(gk);\ open_1 \leftarrow \mathsf{Equiv}(m, ek, tk) \end{array}\right] - \frac{1}{2}$$

is negligible in λ.

A trapdoor commitment scheme is structure-preserving relative to group generator $\mathcal{G}$ if its common parameter gk includes a description of bilinear groups generated by $\mathcal{G}$ and its commitment keys, messages, commitments, and opening information consist only of source group elements, and the verification function consists only of evaluating group membership and relations described by pairing product equations.

From now, we focus on the binding property, which is important for our purpose. The standard binding property requires that it is infeasible for any polynomial time adversary to find two distinct messages and openings for a single commitment value com. As we use a commitment scheme mostly as a hash function, we refer the binding property as collision resistance using terminology for hash functions. A weaker notion known as target collision resistance asks the adversary to find a collision on a given message. An intermediate notion is introduced in [28] as honest-sender binding where the adversary chooses the message for which an honest commitment is made. Thus, the adversary does not choose the randomness used to create the target commitment, but gets to see it and try to create a different opening to a different message. Following [9], we use a refined notion of [28] called *chosen-message target collision resistance* (CMTCR) that handles an arbitrary number of messages.

Definition 14. (*Chosen-Message Target Collision Resistance*) For a trapdoor commitment scheme, TC, let $\mathcal{O}_{ck}$ denote an oracle that, given $m \in \mathcal{M}$, executes $(com, open) \leftarrow \mathsf{Com}(ck, m)$, records (com, m) in Q, and returns $(com, open)$. We say TC is chosen-

message target collision resistant if for any polynomial time adversary $\mathcal{A}$ the advantage defined by

$$\mathrm{Adv}^{\mathsf{cmtcr}}_{\mathsf{TC},\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda), \\ (ck, tk) \leftarrow \mathsf{Key}(gk), \\ (com^\dagger, m^\dagger, open^\dagger) \leftarrow \mathcal{A}^{\mathcal{O}_{ck}}(ck) \end{array} : \begin{array}{l} (com^\dagger, *) \in Q \wedge (com^\dagger, m^\dagger) \notin Q \wedge \\ 1 = \mathsf{Vrf}(ck, com^\dagger, m^\dagger, open^\dagger) \end{array}\right] \tag{11}$$

is negligible in security parameter λ.

A structure-preserving commitment scheme is shrinking if number of group elements in com is strictly less than that in m. The impossibility argument of [8] shows that if a structure-preserving commitment scheme is shrinking, then there exists an adversary that can find two openings and messages that are consistent to a commitment. It is essential for the impossibility argument that the adversary is allowed to choose the randomness used for the target commitment. We stress that it is not the case for the adversary in the game of CMTCR.

3.2. *Message-Transposing Commitment Scheme*

We introduce a commitment scheme with a property that the message space $\mathcal{M}^{com}$ for creating a commitment and the space $\mathcal{M}^{ver}$ for verification differ, and there exists an efficiently computable bijection $\gamma : \mathcal{M}^{com} \to \mathcal{M}^{ver}$. As a message in $\mathcal{M}^{com}$ is bound to a commitment through function γ, we call such a scheme a message-transposing commitment scheme.[1] A formal definition is as follows.

Definition 15. (*Message-Transposing Commitment Scheme*) A message-transposing commitment scheme is a set of algorithms $\mathsf{MTC} = \{\mathsf{Setup}, \mathsf{Key}, \mathsf{Com}, \mathsf{Vrf}, \mathsf{SimCom}, \mathsf{Equiv}\}$ such that

- $\mathsf{Setup}(1^\lambda) \to gk$: A setup function that, given a security parameter λ, generates common parameter gk, which defines message spaces $\mathcal{M}^{com}$ for commitment generation and $\mathcal{M}^{ver}$ for verification, and an efficiently computable bijection $\gamma : \mathcal{M}^{com} \to \mathcal{M}^{ver}$. It also determines the commitment space $\mathcal{C}$, and the opening space $\mathcal{I}$.
- $\mathsf{Key}(gk) \to (ck, tk)$: A key generation algorithm that takes gk and outputs a public commitment key, ck, and a trapdoor key, tk.
- $\mathsf{Com}(ck, m) \to (com, open)$: A commitment algorithm that takes ck and message $m \in \mathcal{M}^{com}$ and outputs a commitment, $com \in \mathcal{C}$, and an opening information, $open \in \mathcal{I}$.
- $\mathsf{Vrf}(ck, com, M, open) \to 1/0$: A verification algorithm that takes ck, com, $M \in \mathcal{M}^{ver}$, and $open$ as inputs, and outputs 1 or 0 representing acceptance or rejection, respectively.

[1] Referred as a γ-binding commitment scheme in [9].

$\mathsf{SimCom}(gk) \to (com, ek)$: A sampling algorithm that takes common parameter gk and outputs commitment com and equivocation key ek.

$\mathsf{Equiv}(M, ek, tk) \to open$: An algorithm that takes ck, ek, tk, and $M \in \mathcal{M}^{ver}$ as input and returns $open$.

Correctness and statistical trapdoor are the same as Definition 13 with trivial adaptation to the message spaces.

We say that a message-transposing commitment scheme is structure preserving with respect to verification if ck, com, $open$, and $\mathcal{M}^{ver}$ consist of source group elements of bilinear groups and the verification function consists only of evaluating group membership and pairing product equations. It is shrinking if the number of group elements in a commitment is strictly less than that in the corresponding message for verification.

Next we formally define the security notions, message-transposing target collision resistance and message-transposing collision resistance. As well as ordinary notions of collision resistance, message-transposing collision resistance implies message-transposing target collision resistance.

Definition 16. (*Message-Transposing Target Collision Resistance*) For a message-transposing commitment scheme, MTC, let $\boldsymbol{com}$ and $\boldsymbol{open}$ denote vectors of commitment and openings produced by Com for uniformly sampled messages $\boldsymbol{m}$. We say MTC is message-transposing target collision resistant if for any polynomial time adversary $\mathcal{A}$ the advantage function

$$\mathrm{Adv}^{\mathsf{tcr}}_{\mathsf{MTC},\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda), (ck, tk) \leftarrow \mathsf{Key}(gk), \\ \boldsymbol{m} \leftarrow \mathcal{M}^{com}, (\boldsymbol{com}, \boldsymbol{open}) \leftarrow \mathsf{Com}(ck, \boldsymbol{m}), \\ (com, M, open) \leftarrow \mathcal{A}(ck, \boldsymbol{m}, \boldsymbol{com}, \boldsymbol{open}) \end{array} : \begin{array}{l} com \in \boldsymbol{com} \ \wedge \ M \notin \gamma(\boldsymbol{m}) \wedge \\ 1 = \mathsf{Vrf}(ck, com, M, open) \end{array}\right]$$

is negligible in security parameter λ.

Definition 17. (*Message-Transposing Collision Resistance*) A message-transposing commitment scheme, MTC, is message-transposing collision resistant if for any polynomial time adversary $\mathcal{A}$ the advantage

$$\mathrm{Adv}^{\mathsf{cr}}_{\mathsf{MTC},\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{l} gk \leftarrow \mathsf{Setup}(1^\lambda), (ck, tk) \leftarrow \mathsf{Key}(gk), \\ (com, M_1, open_1, M_2, open_2) \leftarrow \mathcal{A}(ck) \end{array} : \begin{array}{l} M_1 \neq M_2 \wedge \\ 1 = \mathsf{Vrf}(ck, com, M_1, open_1) \wedge \\ 1 = \mathsf{Vrf}(ck, com, M_2, open_2) \end{array}\right]$$

is negligible in the security parameter λ.

Now we present a concrete scheme for a structure-preserving message-transposing trapdoor commitment scheme for $\gamma : \mathbb{Z}_p \to \mathbb{G}$. For our purpose, we only require target collision resistance but the construction satisfies the stronger notion.

[**Message-Transposing Trapdoor Commitment Scheme:** MTC]

Setup(1^λ): Run $\mathcal{G}(1^\lambda)$ and obtain $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G})$. It defines message spaces $\mathcal{M}^{com} := \mathbb{Z}_p^\ell$, $\mathcal{M}^{ver} := \mathbb{G}^\ell$ for fixed $\ell \geq 1$ and bijection $\gamma : \mathbb{Z}_p^\ell \to \mathbb{G}^\ell$ by $\gamma(m_1, \ldots, m_\ell) = (G^{m_1}, \ldots, G^{m_\ell})$. Output gk.

Key(gk): For $i = 1, \ldots, \ell$, choose $\rho_i \leftarrow \mathbb{Z}_p^*$ and compute $\tilde{X}_i := \tilde{G}^{\rho_i}$. Output $ck := (gk, \tilde{X}_1, \ldots, \tilde{X}_\ell)$ and $tk := (gk, \rho_1, \ldots, \rho_\ell)$.

Com(ck, m): Parse m into $(m_1, \cdots, m_\ell) \in \mathbb{Z}_p^\ell$. Choose $\zeta \leftarrow \mathbb{Z}_p^*$ and compute

$$\tilde{G}_u := \tilde{G}^\zeta \prod_{i=1}^{\ell} \tilde{X}_i^{m_i}, \quad \text{and} \quad R := G^\zeta.$$

Output $com := \tilde{G}_u$ and $open := R$.

Vrf($ck, com, M, open$): Parse $ck = (gk, \tilde{X}_1, \ldots, \tilde{X}_\ell)$, $open = R$, $M = (M_1, \ldots, M_\ell) \in \mathbb{G}^\ell$, and $com = \tilde{G}_u$, respectively. Take generators $(G, \tilde{G})$ from gk. Return 1 if

$$e(G, \tilde{G}_u) = e(R, \tilde{G}) \prod_{i=1}^{\ell} e(M_i, \tilde{X}_i) \tag{12}$$

holds. Return 0, otherwise.

SimCom(gk): Choose $\omega_u \in \mathbb{Z}_p^*$. Compute $\tilde{G}_u := \tilde{G}^{\omega_u}$ and output $com := \tilde{G}_u$ and $ek := \omega_u$.

Equiv(M, ek, tk): Parse $tk = (gk, \rho_1, \ldots, \rho_\ell)$, $ek = \omega_u$, and $M = (M_1, \ldots, M_\ell)$. Compute $R := G^{\omega_u} \prod_{i=1}^{\ell} M_i^{-\rho_i}$. Then output $open := R$.

Theorem 4. MTC *is correct, statistical trapdoor, and structure preserving with respect to verification. It is message-transposing collision resistant if the* DBP *assumption holds.*

Proof. Correctness is verified as

$$\begin{aligned} e(R, \tilde{G}) \prod_{i=1}^{\ell} e(M_i, \tilde{X}_i) &= e(G^\zeta, \tilde{G}) \prod_{i=1}^{\ell} e(G^{m_i}, \tilde{X}_i) \\ &= e(G, \tilde{G}^\zeta)\, e(G, \prod_{i=1}^{\ell} \tilde{X}_i^{m_i}) \ = e(G, \tilde{G}_u). \end{aligned}$$

To see if it is statistically trapdoor, observe that SimCom outputs $\tilde{G}_u$ distributed uniformly over $\tilde{\mathbb{G}}^*$ whereas that from Com distributes statistically close to uniform over $\tilde{\mathbb{G}}$. Then R from Equiv is the one that is uniquely determined by the verification equation since it satisfies

$$e(R,\tilde{G})\prod_{i=1}^{\ell}e(M_i,\tilde{X}_i)=e\left(G^{\omega_u}\prod_{i=1}^{\ell}M_i^{-x_i},\tilde{G}\right)\prod_{i=1}^{\ell}e(M_i,\tilde{G}^{x_i})=e(G,\tilde{G}_u).$$

Finally, it is obviously structure preserving with respect to verification due to verification equation (12).

Next we prove the message-transposing collision resistance. Let $\mathcal{A}$ be an adversary that breaks the CR security of MTC. We show algorithm $\mathcal{B}$ that attacks the DBP with black box access to $\mathcal{A}$. Given an instance $(e,\mathbb{G},\tilde{\mathbb{G}},G,\tilde{G},\tilde{G}_z)$ of the DBP, algorithm $\mathcal{B}$ sets up key ck as follows. Set $gk:=(p,\mathbb{G},\tilde{\mathbb{G}},\mathbb{G}_T,e,G,\tilde{G})$. For $i=1,\ldots,\ell$, choose $\xi_i,\varphi_i\leftarrow(\mathbb{Z}_p^*)^2$ and set $\tilde{X}_i:=(\tilde{G}_z)^{\xi_i}\tilde{G}^{\varphi_i}$. Then give $ck:=(gk,\tilde{X}_1,\ldots,\tilde{X}_\ell)$ to $\mathcal{A}$.

Suppose that $\mathcal{A}$ outputs $(\tilde{G}_u,R_1,M_1,R_2,M_2)$ that passes the verification as required. $\mathcal{B}$ then outputs $(Z^\dagger,R^\dagger)$ where

$$R^\dagger:=\frac{R_1}{R_2}\prod_{i=1}^{\ell}\left(\frac{M_{1i}}{M_{2i}}\right)^{\varphi_i},\ \text{and}\quad Z^\dagger:=\prod_{i=1}^{\ell}\left(\frac{M_{1i}}{M_{2i}}\right)^{\xi_i},\tag{13}$$

as the answer to the DBP. This completes the description of $\mathcal{B}$.

We first verify that the simulated ck is correctly distributed. In the key generation, gk is set legitimately to the given output of $\mathcal{G}$. Each simulated $\tilde{X}_i$ distributes uniformly over $\tilde{\mathbb{G}}$, whereas the real one distributes uniformly over $\tilde{\mathbb{G}}^*$. Thus, the simulated ck is statistically close to the real one.

We then argue that the resulting $(Z^\dagger,R^\dagger)$ is a valid answer to the given instance of the DBP. Since the output from $\mathcal{A}$ satisfies the verification equation, we have

$$1=e\left(\frac{R_1}{R_2},\tilde{G}\right)\prod_{i=1}^{\ell}e\left(\frac{M_{1i}}{M_{2i}},(\tilde{G}_z)^{\xi_i}\tilde{G}^{\varphi_i}\right)\tag{14}$$

$$=e\left(\prod_{i=1}^{\ell}\left(\frac{M_{1i}}{M_{2i}}\right)^{\xi_i},\tilde{G}_z\right)e\left(\frac{R_1}{R_2}\prod_{i=1}^{\ell}\left(\frac{M_{1i}}{M_{2i}}\right)^{\varphi_i},\tilde{G}\right)=e(Z^\dagger,\tilde{G}_z)\,e(R^\dagger,\tilde{G}).\tag{15}$$

Observe that every ξ_i is independent of the view of $\mathcal{A}$ as it is information-theoretically hidden in $\tilde{X}_i$. Since a valid output from $\mathcal{A}$ satisfies $M_1\neq M_2$, there exists an index $i^\dagger\in\{1,\ldots,\ell\}$ that $M_{1i^\dagger}\neq M_{2i^\dagger}$. Thus, $Z^\dagger$ distributes as well as $(M_{1i}/M_{2i})^{\xi_i}$ at $i=i^\dagger$. Since $M_{1i^\dagger}/M_{2i^\dagger}\neq 1$ and $\xi_{i^\dagger}$ is uniform over $\mathbb{Z}_p^*$, we conclude that $Z^\dagger=1$ occurs only with negligible probability.

Thus, $\mathcal{B}$ breaks the DBP assumption with almost the same probability and running time of $\mathcal{A}$ breaking the message-transposing collision resistance of MTC. □

3.3. *Structure-Preserving Shrinking Trapdoor Commitment Scheme*

We construct trapdoor commitment scheme TC by combining partially one-time signature scheme POS and message-transposing trapdoor commitment scheme MTC. A key idea is to commit to both long-term and one-time *secret keys* of POS by using shrinking MTC and allow verification of the commitment using corresponding public

keys as opening information. For this to be possible, the bijection γ must correspond to the mapping from the secret key space to the public key space of POS. More precisely, we require the following properties satisfied by POS and MTC. Let $\mathcal{M}_{\mathsf{pos}}$ be the message space of POS defined with respect to gk. We denote the key spaces as $\mathcal{K}^{vk}_{\mathsf{pos}}$, $\mathcal{K}^{sk}_{\mathsf{pos}}$, $\mathcal{K}^{ovk}_{\mathsf{pos}}$, and $\mathcal{K}^{osk}_{\mathsf{pos}}$ in a self-explanatory manner. There must exist efficiently computable bijections $\gamma_{sk} : \mathcal{K}^{sk}_{\mathsf{pos}} \to \mathcal{K}^{vk}_{\mathsf{pos}}$ and $\gamma_{osk} : \mathcal{K}^{osk}_{\mathsf{pos}} \to \mathcal{K}^{ovk}_{\mathsf{pos}}$, and the MTC is for $\gamma := \gamma_{sk} \times \gamma^{(1)}_{osk} \times \cdots \times \gamma^{(k)}_{osk}$. It is also required that POS and MTC have a common setup function, Setup, that outputs gk based on POS.Setup and MTC.Setup, as mentioned in Sect. 2.3. (When instantiated from POS in Sect. 2.5 and MTC from Sect. 3.2, Setup is as simple as running $gk \leftarrow \mathcal{G}(1^\lambda)$.) The construction is as follows.

[**Trapdoor Commitment Scheme:** TC]

$\mathsf{Setup}(1^\lambda)$: It the same as the common setup function for POS and MTC. The commitment message space $\mathcal{M}^{com}_{\mathsf{mtc}}$ and verification message space $\mathcal{M}^{ver}_{\mathsf{mtc}}$ for MTC are set to $\mathcal{M}^{com}_{\mathsf{mtc}} := \mathcal{K}^{sk}_{\mathsf{pos}} \times (\mathcal{K}^{osk}_{\mathsf{pos}})^k$ and $\mathcal{M}^{ver}_{\mathsf{mtc}} := \mathcal{K}^{vk}_{\mathsf{pos}} \times (\mathcal{K}^{ovk}_{\mathsf{pos}})^k$. The message space for TC is $\mathcal{M} := (\mathcal{M}_{\mathsf{pos}})^k$ for some integer $k > 0$.

$\mathsf{Key}(gk)$: Run $(ck_{\mathsf{mtc}}, tk_{\mathsf{mtc}}) \leftarrow \mathsf{MTC.Key}(gk)$. Output $ck := ck_{\mathsf{mtc}}$ and $tk := tk_{\mathsf{mtc}}$. It is assumed that gk is included in ck.

$\mathsf{Com}(ck, M)$: Parse $ck := ck_{\mathsf{mtc}}$ and $M := (M^{(1)}, \cdots, M^{(k)}) \in (\mathcal{M}_{\mathsf{pos}})^k$. Take gk from ck. Run

$$\begin{aligned}
(vk_{\mathsf{pos}}, sk_{\mathsf{pos}}) &\leftarrow \mathsf{POS.Key}(gk),\\
(ovk^{(j)}_{\mathsf{pos}}, osk^{(j)}_{\mathsf{pos}}) &\leftarrow \mathsf{POS.Ovk}(gk) \text{ for } j = 1, \ldots, k,\\
\sigma^{(j)}_{\mathsf{pos}} &\leftarrow \mathsf{POS.Sign}(sk_{\mathsf{pos}}, osk^{(j)}_{\mathsf{pos}}, M^{(j)}) \text{ for } j = 1, \ldots, k, \text{ and}\\
(com_{\mathsf{mtc}}, open_{\mathsf{mtc}}) &\leftarrow \mathsf{MTC.Com}(ck_{\mathsf{mtc}}, (sk_{\mathsf{pos}}, osk^{(1)}_{\mathsf{pos}}, \cdots, osk^{(k)}_{\mathsf{pos}})).
\end{aligned}$$

Output $com := com_{\mathsf{mtc}}$ and $open := (open_{\mathsf{mtc}}, vk_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \cdots, ovk^{(k)}_{\mathsf{pos}}, \sigma^{(1)}_{\mathsf{pos}}, \cdots, \sigma^{(k)}_{\mathsf{pos}})$.

$\mathsf{Vrf}(ck, com, M, open)$: Parse $com = com_{\mathsf{mtc}}$, $M = (M^{(1)}, \cdots, M^{(k)}) \in (\mathcal{M}_{\mathsf{pos}})^k$ and $open = (open_{\mathsf{mtc}}, vk_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \cdots, ovk^{(k)}_{\mathsf{pos}}, \sigma^{(1)}_{\mathsf{pos}}, \cdots, \sigma^{(k)}_{\mathsf{pos}})$. Execute

$$\begin{aligned}
b_0 &\leftarrow \mathsf{MTC.Vrf}(ck_{\mathsf{mtc}}, com_{\mathsf{mtc}}, (vk_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \cdots, ovk^{(k)}_{\mathsf{pos}}), open_{\mathsf{mtc}}), \text{ and}\\
b_j &\leftarrow \mathsf{POS.Vrf}(vk_{\mathsf{pos}}, ovk^{(j)}_{\mathsf{pos}}, M^{(j)}, \sigma^{(j)}_{\mathsf{pos}}) \text{ for } j = 1, \ldots, k.
\end{aligned}$$

Output 1 if $b_j = 1$ for all $j = 0, \ldots, k$. Output 0, otherwise.

$\mathsf{SimCom}(gk)$: Take gk_{mtc} from gk and run $(com_{\mathsf{mtc}}, ek_{\mathsf{mtc}}) \leftarrow \mathsf{MTC.SimCom}(gk_{\mathsf{mtc}})$ and output $com := com_{\mathsf{mtc}}$ and $ek := (com_{\mathsf{mtc}}, ek_{\mathsf{mtc}})$.

$\mathsf{Equiv}(M, ek, tk)$: The same as TC.Com except that, MTC.Com is replaced by $open_{\mathsf{mtc}} \leftarrow \mathsf{MTC.Equiv}((vk_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \cdots, ovk^{(k)}_{\mathsf{pos}}), ek_{\mathsf{mtc}}, tk_{\mathsf{mtc}})$ and com_{mtc} included in ek.

Theorem 5. *The commitment scheme* TC *described above is* CMTCR *if* POS *is* MK-OT-NACMA*, and* MTC *is message-transposing target collision resistant.*

Proof. We follow the game transition framework. Let Game 0 be the CMTCR game launched by adversary $\mathcal{A}$. By $com^\dagger = com^\dagger_{\mathsf{mtc}}$, $open^\dagger = (open^\dagger_{\mathsf{mtc}}, vk^\dagger_{\mathsf{pos}}, ovk^{\dagger\,(1)}_{\mathsf{pos}}, \ldots, ovk^{\dagger\,(k)}_{\mathsf{pos}}, \sigma^{\dagger\,(1)}_{\mathsf{pos}}, \ldots, \sigma^{\dagger\,(k)}_{\mathsf{pos}})$ and $M^\dagger = (M^{\dagger(1)}, \ldots, M^{\dagger(k)})$, we denote the collision $\mathcal{A}$ outputs.

In Game 1, abort if $(vk^\dagger_{\mathsf{pos}}, ovk^{\dagger\,(1)}_{\mathsf{pos}}, \ldots, ovk^{\dagger\,(k)}_{\mathsf{pos}})$ is different from any of $(vk^{[i]}_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \ldots, ovk^{(k)}_{\mathsf{pos}})$ observed by the signing oracle. We claim that if such an abort happens, then MTC is broken. It is shown by constructing adversary $\mathcal{B}$ attacking the message-transposing target collision resistance of MTC. Adversary $\mathcal{B}$ is given ck_{mtc} and q_s reference commitments com_{mtc} and opening $open_{\mathsf{mtc}}$ for random messages of the form $(sk^{[i]}_{\mathsf{pos}}, osk^{(1)}_{\mathsf{pos}}, \ldots, osk^{(k)}_{\mathsf{pos}})$. Each message is uniquely mapped to $(vk^{[i]}_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \ldots, ovk^{(k)}_{\mathsf{pos}})$ by bijection γ. Adversary $\mathcal{B}$ invokes $\mathcal{A}$ with $ck := ck_{\mathsf{mtc}}$ as input. For every commitment query M, adversary $\mathcal{B}$ takes a fresh sample $(sk^{[i]}_{\mathsf{pos}}, osk^{(1)}_{\mathsf{pos}}, \ldots, osk^{(k)}_{\mathsf{pos}})$ with its commitment com_{mtc} and opening $open_{\mathsf{mtc}}$, and computes $\sigma^{(j)}_{\mathsf{pos}} \leftarrow \mathsf{POS.Sign}(sk_{\mathsf{pos}}, osk^{(j)}_{\mathsf{pos}}, M^{(j)})$ for $j = 1, \ldots, k$. It then returns $com := com_{\mathsf{mtc}}$ and $open := (open_{\mathsf{mtc}}, vk_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \cdots, ovk^{(k)}_{\mathsf{pos}}, \sigma^{(1)}_{\mathsf{pos}}, \ldots, \sigma^{(k)}_{\mathsf{pos}})$. If $\mathcal{A}$ eventually outputs a collision, $\mathcal{B}$ outputs $com^\dagger_{\mathsf{mtc}} := com^\dagger_{\mathsf{mtc}}, open^\dagger_{\mathsf{mtc}} := open^\dagger_{\mathsf{mtc}}$ and $M^\dagger := (vk^\dagger_{\mathsf{pos}}, ovk^{\dagger\,(1)}_{\mathsf{pos}}, \ldots, ovk^{\dagger\,(k)}_{\mathsf{pos}})$. This completes the description of $\mathcal{B}$.

The simulated commitments *com* and openings *open* have the same distributions as the real ones since every $osk^{(j)}_{\mathsf{pos}}$ is sampled legitimately by the challenger and the commitment generation procedure is the genuine one. Furthermore, the output of $\mathcal{B}$ is a valid collision against MTC since $\mathcal{A}$ must have chosen $com^\dagger (= com^\dagger_{\mathsf{mtc}})$ from previously used commitments and $M^\dagger$ must be fresh for the attack being successful by definition. Accordingly, we have $|\Pr[\text{Game 0}] - \Pr[\text{Game 1}]| \leq \mathrm{Adv}^{\mathsf{tcr}}_{\mathsf{MTC},\mathcal{B}}(\lambda)$.

We then argue that $\mathcal{A}$ wins in Game 1 only if POS is broken. Let $\mathcal{C}$ be an adversary attacking the MK-OT-NACMA property of POS. Given gk, it executes $(ck_{\mathsf{mtc}}, tk_{\mathsf{mtc}}) \leftarrow \mathsf{MTC.Key}(gk)$. Then it invokes $\mathcal{A}$ with input $ck := ck_{\mathsf{mtc}}$. For each i-th query $M^{[i]} = (M^{[i],(1)}, \ldots, M^{[i],(k)})$, $\mathcal{C}$ makes a key generation query to $\mathcal{O}_k$ to obtain $vk^{(j)}_{\mathsf{pos}}$, and then makes signing queries to $\mathcal{O}_s$ for $(M^{[i],(1)}, \ldots, M^{[i],(k)})$ with respect to $vk^{[i]}_{\mathsf{pos}}$. On receiving corresponding signatures from $\mathcal{O}_s$, $\mathcal{C}$ computes $(com_{\mathsf{mtc}}, ek_{\mathsf{mtc}}) \leftarrow \mathsf{MTC.SimCom}(gk)$ and $open_{\mathsf{mtc}} \leftarrow \mathsf{MTC.Equiv}((vk^{[i]}_{\mathsf{pos}}, ovk_{\mathsf{pos}}{}^{(1)}, \cdots, ovk_{\mathsf{pos}}{}^{(k)}), ek_{\mathsf{mtc}}, tk_{\mathsf{mtc}})$ and outputs $com := com_{\mathsf{mtc}}$ and $open := (open_{\mathsf{mtc}}, vk^{[i]}_{\mathsf{pos}}, ovk^{(1)}_{\mathsf{pos}}, \ldots, ovk^{(k)}_{\mathsf{pos}}, \sigma^{(1)}_{\mathsf{pos}}, \ldots, \sigma^{(k)}_{\mathsf{pos}})$. On receiving a collision from $\mathcal{A}$, $\mathcal{C}$ searches for $vk^{[i]^\dagger}_{\mathsf{pos}}$ that $vk_{\mathsf{pos}} = vk^{[i]^\dagger}_{\mathsf{pos}}$. Note that this search always succeeds if the game does not abort. It then finds $j^\dagger$ that $M^{\dagger(j)^\dagger} \neq M^{[i]^\dagger,(j)^\dagger}$ (such an index must exist since $M^\dagger$ differs from any queried messages with respect to $i^\dagger$) and outputs $vk^{[i]^\dagger}_{\mathsf{pos}}, ovk^{(j)^\dagger}_{\mathsf{pos}}$ and $M^\dagger := M^{\dagger\,[i]^\dagger,(j)^\dagger}$. This completes the description of $\mathcal{C}$. The simulated signatures are statistically close to the real ones due to the statistical trapdoor property of MTC.SimCom and MTC.Equiv.

Thus, we have $\Pr[\text{Game } 1] - \epsilon_{\mathsf{sim}} \leq \mathrm{Adv}^{\mathsf{mk\text{-}ot\text{-}nacma}}_{\mathsf{POS},\mathcal{C}}(\lambda)$, where ϵ_{sim} is the statistical loss by switching from MTC.SimCom to MTC.Equiv.

All in all, we have

$$\mathrm{Adv}^{\mathsf{cmtcr}}_{\mathsf{TC},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{tcr}}_{\mathsf{MTC},\mathcal{B}}(\lambda) + \mathrm{Adv}^{\mathsf{mk\text{-}ot\text{-}nacma}}_{\mathsf{POS},\mathcal{C}}(\lambda) + \epsilon_{\mathsf{sim}},$$

which proves the statement. □

The following is immediate from the construction. In particular, correctness holds due to the correctness of MTC and POS and the existence of a bijection from the secret keys of POS to the verification keys.

Theorem 6. TC *given above is a structure-preserving trapdoor commitment scheme if* MTC *is structure preserving with respect to verification, and* POS *is structure preserving.*

4. Fully Structure-Preserving Signatures

We argue that constructing an FSPS requires a different approach than those for all known constructions of SPS. The verification equations of existing structure-preserving constant-size signatures on message vectors $(G^{m_1}, \ldots, G^{m_L})$ involve pairings such as $\prod e(G^{x_i}, G^{m_i})$, where G^{x_i} is a public key element and G^{m_i} is a message element. The message is squashed into a signature element, say S, in such a form that $S := A \cdot \prod_{i=1}^{L} G^{m_i x_i}$ where x_i is a signing key component and A is computed from inputs other than the message. Such a structure requires either m_i or x_i to be known to a signing algorithm that uses generic group operations. In FSPS, however, neither is given to the signing function.

Our starting point is the FSPS scheme in Sect. 2.6. The following sections present constructions that upgrade the security to UF-CMA by incorporating one-time signatures or trapdoor commitments.

4.1. *Warm-Up*

Our first approach is to take random x_i instead of the signing key. That is, x_i works as a random one-time key and G^{x_i} is regarded as a one-time public key, which is then authenticated by an FSPS with a long-term key that is secure against extended random message attacks. This results in a combination of a weaker signature scheme with OTS, which is well known as a method for upgrading the security of the underlying signature scheme. This in fact can be seen as a special case of the construction of SPS by Abe et al. [2]. We nevertheless work out the scheme in detail to discuss our motivation for our main scheme and settle a basis for comparison. Let OTS and xSIG be a one-time and an ordinary signature scheme that have common setup function Setup. We construct FSP1 as follows.

[**Signature Scheme:** FSP1]

$\mathsf{Setup}(1^\lambda)$: It is the same as Setup for OTS and xSIG. It outputs $gk \leftarrow \mathsf{Setup}(1^\lambda)$, and sets $\mathcal{M}_{\mathsf{xsig}} := \mathcal{K}^{vk}_{\mathsf{ots}}$ and $\mathcal{M} := \mathcal{M}_{\mathsf{ots}}$.

$\mathsf{Key}(gk)$: Run $(vk_{\mathsf{xsig}}, sk_{\mathsf{xsig}}) \leftarrow \mathsf{xSIG.Key}(gk)$. (It is assumed that gk is included in vk_{xsig} and sk_{xsig}.) Output $(vk, sk) := (vk_{\mathsf{xsig}}, sk_{\mathsf{xsig}})$.

$\mathsf{Sign}(sk, M)$: Take sk_{xsig} and gk from sk. Compute

$$\begin{aligned}(ovk_{\mathsf{ots}}, osk_{\mathsf{ots}}) &\leftarrow \mathsf{OTS.Key}(gk),\\ \sigma_{\mathsf{xsig}} &\leftarrow \mathsf{xSIG.Sign}(sk_{\mathsf{xsig}}, ovk_{\mathsf{ots}}), \text{ and}\\ \sigma_{\mathsf{ots}} &\leftarrow \mathsf{OTS.Sign}(osk_{\mathsf{ots}}, M).\end{aligned}$$

Output $\sigma := (\sigma_{\mathsf{xsig}}, \sigma_{\mathsf{ots}}, ovk_{\mathsf{ots}})$

$\mathsf{Vrf}(vk, M, \sigma)$: : Take vk_{xsig} and $(\sigma_{\mathsf{xsig}}, \sigma_{\mathsf{ots}}, ovk_{\mathsf{ots}})$ from the input. Output 1 if

$$1 = \mathsf{OTS.Vrf}(vk_{\mathsf{ots}}, M, \sigma_{\mathsf{ots}}) \quad \text{and} \quad 1 = \mathsf{xSIG.Vrf}(vk_{\mathsf{xsig}}, vk_{\mathsf{ots}}, \sigma_{\mathsf{xsig}}).$$

Output 0, otherwise.

Theorem 7. *If* OTS *is a* UF-NACMA *secure SPS and* xSIG *is a* UF-XRMA *secure FSPS, then* FSP1 *is a* UF-CMA *secure FSPS scheme.*

Proof. Since the syntactical consistency and correctness are trivial from the construction, we only show that the scheme is fully structure preserving. The public component of FSP1 is $(vk, \sigma, M) = (vk_{\mathsf{xsig}}, (\sigma_{\mathsf{xsig}}, \sigma_{\mathsf{ots}}, ovk_{\mathsf{ots}}), M)$, which consists of public components of xSIG.Key and the OTS. Also, the signing key of FSP1 consists of sk_{xsig}. Thus, both public and private components of FSP1 consist of group elements since xSIG is FSPS and the OTS is SPS. Furthermore, FSP1.Vrf evaluates OTS.Vrf and xSIG.Vrf that evaluate PPEs. Thus, FSP1 is FSPS.

We next prove the UF-CMA security of FSP1 by following the standard game transition technique. Let $\mathcal{A}$ be an adversary against FSP1. By Pr[Game i] we denote probability that $\mathcal{A}$ eventually outputs a valid forgery as defined in Definition 2. Let Game 0 be the UF-CMA game that $\mathcal{A}$ is playing. By definition, $\Pr[\text{Game } 0] = \mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{FSP1},\mathcal{A}}(\lambda)$. Let $(\sigma^\dagger, M^\dagger)$ be a forgery $\mathcal{A}$ outputs. Let $\sigma^\dagger := (\sigma^\dagger_{\mathsf{xsig}}, \sigma^\dagger_{\mathsf{ots}}, vk^\dagger_{\mathsf{ots}})$.

In Game 1, abort the game if $(\sigma^\dagger, M^\dagger)$ is a valid forgery and $vk^\dagger_{\mathsf{ots}}$ is never used by the signing oracle. We show that this event occurs only if the UF-XRMA security of xSIG is broken. Let $\mathcal{B}$ be an adversary against xSIG launching an XRMA attack. $\mathcal{B}$ is given a public key vk_{xsig}, message $m^{(j)} := vk^{(j)}_{\mathsf{ots}}$, signature $\sigma^{(j)}_{\mathsf{xsig}})$ for $j = 1, \ldots, q_s$. It is also given random coin $\omega^{(j)}$ for each j used to generate $vk^{(j)}_{\mathsf{ots}}$ using OTS.Key as the message sampler. $\mathcal{B}$ first computes $sk^{(j)}_{\mathsf{ots}}$ from $\omega^{(j)}$ by executing OTS.Key by itself. Then it invokes $\mathcal{A}$ with input $vk := vk_{\mathsf{xsig}}$. On receiving $M^{(j)}$ from $\mathcal{A}$ for signing, $\mathcal{B}$ computes $\sigma^{(j)}_{\mathsf{ots}} \leftarrow \mathsf{OTS.Sign}(sk^{(j)}_{\mathsf{ots}}, M^{(j)})$ and returns $\sigma^{(j)} := (\sigma^{(j)}_{\mathsf{xsig}}, \sigma^{(j)}_{\mathsf{ots}}, vk^{(j)}_{\mathsf{ots}})$. When $\mathcal{A}$ outputs forgery $\sigma^\dagger := (\sigma^\dagger_{\mathsf{xsig}}, \sigma^\dagger_{\mathsf{ots}}, vk^\dagger_{\mathsf{ots}})$ for some message $M^\dagger$, $\mathcal{B}$ outputs

$\sigma^{\dagger}_{\mathsf{xsig}} := \sigma^{\dagger}_{\mathsf{xsig}}$ and $m^{\dagger} := vk^{\dagger}_{\mathsf{ots}}$. This is a valid forgery since $\mathcal{A}$'s forgery is supposed to satisfy $vk^{\dagger}_{\mathsf{ots}} \neq vk^{(j)}_{\mathsf{ots}}$. Thus, we have $|\Pr[\text{Game 0}] - \Pr[\text{Game 1}]| \leq \mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{xSIG},\mathcal{B}}(\lambda)$.

Next we show that $\mathcal{A}$ wins Game 1 only if OTS is broken. Let $\mathcal{C}$ be an adversary attacking OTS with NACMA. Given gk from outside, $\mathcal{C}$ first chooses a random index $i \leftarrow \{1, \ldots, q_s\}$. It then executes $(vk, sk) \leftarrow \mathsf{FSP1.Key}(gk)$. Given $j(\neq i)$-th query $M^{(j)}$ from $\mathcal{A}$, $\mathcal{C}$ runs $\sigma^{(j)} \leftarrow \mathsf{FSP1.Sign}(sk, M^{(j)})$ and returns $\sigma^{(j)}$. Given $j(= i)$-th query, $\mathcal{C}$ forwards $M^{(j)}$ to the signing oracle of OTS and receive $\sigma^{(j)}_{\mathsf{ots}}$ and $vk^{(j)}_{\mathsf{ots}}$. Then $\mathcal{C}$ executes $\sigma_{\mathsf{xsig}} \leftarrow \mathsf{xSIG.Sign}(sk_{\mathsf{xsig}}, vk^{(j)}_{\mathsf{ots}})$ and returns $\sigma := (\sigma^{(j)}_{\mathsf{xsig}}, \sigma^{(j)}_{\mathsf{ots}}, vk^{(j)}_{\mathsf{ots}})$ to $\mathcal{A}$. When $\mathcal{A}$ outputs forgery $\sigma^{\dagger} := (\sigma^{\dagger}_{\mathsf{xsig}}, \sigma^{\dagger}_{\mathsf{ots}}, vk^{\dagger}_{\mathsf{ots}})$ and $M^{\dagger}$, $\mathcal{C}$ aborts if $vk^{\dagger}_{\mathsf{ots}} \neq vk^{(i)}_{\mathsf{ots}}$. Otherwise, $\mathcal{C}$ outputs $\sigma^{\dagger}_{\mathsf{xsig}} := \sigma^{\dagger}_{\mathsf{ots}}$ and $m^{\dagger} := M^{\dagger}$. This is a valid forgery since $M^{\dagger} \neq M^{(j)}$ for all j including the case $j = i$. Thus, we have $\Pr[\text{Game 1}] \leq q_s \cdot \mathrm{Adv}^{\mathsf{uf\text{-}nacma}}_{\mathsf{OTS},\mathcal{C}}(\lambda)$.

In total, we have

$$\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{FSP1},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{xSIG},\mathcal{B}}(\lambda) + q_s \cdot \mathrm{Adv}^{\mathsf{uf\text{-}nacma}}_{\mathsf{OTS},\mathcal{C}}(\lambda),$$

which proves the statement. □

Though the above reduction involves a loss factor of q_s, it will vanish if OTS is based on a random self-reducible problem like SDP.

The above construction requires $\mathcal{K}^{vk}_{\mathsf{ots}}$ to match $\mathcal{M}_{\mathsf{xsig}}$. When they are instantiated with the concrete schemes from previous sections (using the POS in Sect. 2.5 as OTS by swapping $\mathbb{G}$ and $\tilde{\mathbb{G}}$, and using xSIG in Sect. 2.6), the space adjustment is done as follows.

[Procedure: Matching $\mathcal{K}^{vk}_{\mathsf{ots}}$ to $\mathcal{M}_{\mathsf{xsig}}$]

Setup: It runs xSIG.Setup and sets $(F_1, \tilde{F}_1)$ as default generators $(G, \tilde{G})$ for OTS. It also provide extra generators $(F_2, U_1, \ldots, U_{L+2})$ to OTS for the following procedures to work.

OTS.Key: It runs POS.Key and POS.Ovk in sequence and set $vk_{\mathsf{ots}} := (vk_{\mathsf{pos}}, ovk_{\mathsf{pos}})$. The key spaces are adjusted as follows.

- POS.Key On top of the legitimate procedure with $G := F_1$ to obtain $(G^{w_z}, G^{\chi_1}, \ldots, G^{\chi_L})$, it computes the extended part as $G_{i2} := F_2^{\chi_i}$ $G_{i3} := U_i^{\chi_i}$ for $i = 1, \ldots, L$, and $G_{z2} := F_2^{w_z}$, $G_{z3} := U_{L+1}^{w_z}$, and include all of them to vk_{pos}.
- POS.Ovk On top of legitimate procedure with $G := F_1$ that computes $A := G^a$, it computes extra parts $A_2 := F_2^a$ and $A_3 := U_{L+2}^a$ and includes them to ovk_{pos}.

Then those extended vk_{pos} and ovk_{pos} constitute a message $((G_z, G_{z2}, G_{z3}), (G_1, G_{12}, G_{13}), \ldots, (G_L, G_{L2}, G_{L3}), (A, A_2, A_3))$ given to xSIG to sign. We present a summary of the resulting instantiation of FSP1 below.

Common Parameter	$(G, \tilde{G}, F_1, F_2, \tilde{F}_1, \tilde{F}_2, \{U_i, \tilde{U}_i\}_{i=1}^{L+2})$
Public-key	$(\tilde{V}_1, \tilde{V}_2, \tilde{V}_3, \tilde{V}_4, \tilde{V}_5, \tilde{V}_6, V_7, \tilde{V}_8)$
Secret-key	(K_1, K_2, K_3, K_4)
Message	$(M_1, \ldots, M_L)$
Signature	$(\tilde{S}_0, S_1, \ldots, S_5, \tilde{A}, \tilde{A}_2, \tilde{A}_3, \tilde{G}_z, \tilde{G}_{z2}, \tilde{G}_{z3}, \{\tilde{G}_i, \tilde{G}_{i2}, \tilde{G}_{i3}\}_{i=1}^{L}, Z, R)$
Verification PPEs	$e(G, \tilde{A}) = e(Z, \tilde{G}_z)\, e(R, \tilde{G}) \prod_{i=1}^{L} e(M_i, \tilde{G}_i)$,
	$e(S_5, \tilde{V}_6\, \tilde{A}_3\, \tilde{G}_{z3} \prod_{i=1}^{L} \tilde{G}_{i3}) = e(G, \tilde{S}_0)$,
	$e(S_1, \tilde{V}_1)\, e(S_2, \tilde{V}_3)\, e(S_3, \tilde{V}_2) = e(S_4, \tilde{V}_4)\, e(S_5, \tilde{V}_5)\, e(V_7, \tilde{V}_8)$,
	$e(F_1, \tilde{A}_3) = e(U_{L+2}, \tilde{A})$, $\quad e(F_2, \tilde{A}_3) = e(U_{L+2}, \tilde{A}_2)$
	$e(F_1, \tilde{G}_{z3}) = e(U_{L+1}, \tilde{G}_z)$, $\; e(F_2, \tilde{G}_{z3}) = e(U_{L+1}, \tilde{G}_{z2})$
	$e(F_1, \tilde{G}_{i3}) = e(U_i, \tilde{G}_i)$, $\quad e(F_2, \tilde{G}_{i3}) = e(U_i, \tilde{G}_{i2})$ (for $i = 1, \ldots, L$).

Motivation for Improvement Since an SPS is an OTS, construction FSP1 can be seen as a generic conversion from any SPS to an FSPS. In exchange for the generality, the construction has several shortcomings when instantiated with current building blocks.

- ($O(|m|)$-size signatures) The resulting signature σ includes the one-time verification key ovk_{ots}, which is linear in the size of messages in all current instantiations of OTS.
- (Factor 3 expansion in xSIG) As shown above, the message space of xSIG must cover ovk_{ots}, which is linear in the size of the message. Even worse, the currently known instantiation of xSIG suffers from an expansion factor of $\mu = 3$ for messages. That is, to sign a message consisting of a group element, say G^x, it is required to represent the message with two more extra elements F_2^x and U_i^x for given bases F_2 and U_i. Thus, the size of ovk_{ots} will actually be μ times larger than the one-time verification key that OTS originally requires.

The above shortcomings amplify each other. Finding an instantiation of xSIG with a smaller expansion factor is one direction of improvement. We leave it as an interesting open problem and focus on a generic approach in the next section.

4.2. *Main Construction*

Our idea is to avoid signing any components whose size grows to the size of messages directly with xSIG. We achieve this by committing to the message using a shrinking commitment scheme and signing the commitment with xSIG. Combining a trapdoor commitment scheme (or a chameleon hash) and a signature scheme to achieve such an improvement is a known approach. What is important here is to clarify the required security for each building block. We show that chosen-message target collision resistance is sufficient for TC to reach UF-CMA in combination with an XRMA-secure signature scheme.

Let xSIG be a UF-XRMA secure FSPS scheme and TC be a CMTCR secure trapdoor commitment scheme with common setup function Setup. We construct our FSPS scheme FSP2 from xSIG and TC as follows.

[**Signature Scheme:** FSP2]

$\mathsf{Setup}(1^\lambda)$: Run common setup $gk \leftarrow \mathsf{Setup}(1^\lambda)$ and output gk. Set the message spaces $\mathcal{M}_{\mathsf{xsig}} := \mathcal{C}_{\mathsf{tc}}$ and $\mathcal{M} := \mathcal{M}_{\mathsf{tc}}$.

$\mathsf{Key}(gk)$: Run $(vk_{\mathsf{xsig}}, sk_{\mathsf{xsig}}) \leftarrow \mathsf{xSIG.Key}(gk)$, and $(ck_{\mathsf{tc}}, tk_{\mathsf{tc}}) \leftarrow \mathsf{TC.Key}(gk)$. Set $vk := (vk_{\mathsf{xsig}}, ck_{\mathsf{tc}})$, $sk := sk_{\mathsf{xsig}}$. Output (vk, sk)

$\mathsf{Sign}(sk, M)$: Parse sk into sk_{xsig}. Run $(com_{\mathsf{tc}}, open_{\mathsf{tc}}) \leftarrow \mathsf{TC.Com}(ck_{\mathsf{tc}}, M)$ and $\sigma_{\mathsf{xsig}} \leftarrow \mathsf{xSIG.Sign}(sk_{\mathsf{xsig}}, com_{\mathsf{tc}})$. Output $\sigma := (\sigma_{\mathsf{xsig}}, open_{\mathsf{tc}}, com_{\mathsf{tc}})$.

$\mathsf{Vrf}(vk, M, \sigma)$: Parse $vk = (vk_{\mathsf{xsig}}, ck_{\mathsf{tc}})$ and $\sigma = (\sigma_{\mathsf{xsig}}, open_{\mathsf{tc}}, com_{\mathsf{tc}})$. Output 1 if $1 = \mathsf{TC.Vrf}(ck_{\mathsf{tc}}, com_{\mathsf{tc}}, M, open_{\mathsf{tc}})$ and $1 = \mathsf{xSIG.Vrf}(vk_{\mathsf{xsig}}, com_{\mathsf{tc}}, \sigma_{\mathsf{xsig}})$. Output 0, otherwise.

Note that trapdoor tk_{tc} is not included in sk but used only in the security proof. It is the point that makes the scheme fully structure preserving.

Theorem 8. *If* TC *is a* CMTCR *secure SPTC, and* xSIG *is a* UF-XRMA *secure FSPS relative to* TC.SimCom *as a message sampler, then* FSP2 *is a* UF-CMA *FSPS.*

Proof. Correctness holds trivially from those of the underlying TC and xSIG. Regarding the full structure-preserving property, observe that sk consists of sk_{xsig}, and it consists only of source group elements since xSIG is fully structure preserving. The same is true for public components, i.e., public keys, messages, and signatures consist only of source group elements because both tk_{tc} and xSIG are structure preserving. The verification only evaluates verification functions of these underlying building blocks, which evaluate PPEs. Thus, FSP2 is FSPS.

We next prove the security property. Let $\mathcal{A}$ be an adversary against FSP2. Let Game 0 be the UF-CMA game that $\mathcal{A}$ is playing. By definition, $\Pr[\text{Game } 0] = \mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{FSP2},\mathcal{A}}(\lambda)$. Let $(\sigma^\dagger, m^\dagger)$ be a forgery $\mathcal{A}$ outputs. Let $\sigma^\dagger := (\sigma^\dagger_{\mathsf{xsig}}, open^\dagger_{\mathsf{tc}}, com^\dagger_{\mathsf{tc}})$.

In Game 1, abort the game if $(\sigma^\dagger, m^\dagger)$ is a valid forgery and $com^\dagger_{\mathsf{tc}}$ is never queried by the signing oracle. We show that this event occurs only if the UF-XRMA security of xSIG is broken. Let $\mathcal{B}$ be an adversary against xSIG launching an XRMA attack. The message sampler for XRMA is TC.SimCom. That is, the challenger samples random messages by $(com_{\mathsf{tc}}, ek_{\mathsf{tc}}) \leftarrow \mathsf{TC.SimCom}(gk; \omega)$ with random coin ω and gives com_{tc} and ω with signature σ_{xsig} on com_{tc} as a message. Let $sample^{[i]}$ be the i-th sample, i.e., $sample^{[i]} := (com^{[i]}_{\mathsf{tc}}, \omega^{[i]}, \sigma^{[i]}_{\mathsf{xsig}})$. Given $(vk_{\mathsf{xsig}}, sample^{[1]}, \ldots, sample^{[q_s]})$ as input, $\mathcal{B}$ runs as follows. It first takes gk from vk_{xsig} and recovers every $ek^{[i]}_{\mathsf{tc}}$ from $\omega^{[i]}$ by $(com_{\mathsf{tc}}, ek_{\mathsf{tc}}) \leftarrow \mathsf{TC.SimCom}(gk; \omega)$. It then runs $(ck_{\mathsf{tc}}, tk_{\mathsf{tc}}) \leftarrow \mathsf{TC.Key}(gk)$ and invokes $\mathcal{A}$ with input $vk := (vk_{\mathsf{xsig}}, ck_{\mathsf{tc}})$. Given the i-th signing query $m^{[i]}$ from $\mathcal{A}$, it executes $open^{[i]}_{\mathsf{tc}} \leftarrow \mathsf{TC.Equiv}(m^{[i]}, tk^{[i]}_{\mathsf{tc}}, ek^{[i]}_{\mathsf{tc}},)$ and returns $\sigma := (\sigma^{[i]}_{\mathsf{xsig}}, open^{[i]}_{\mathsf{tc}}, com^{[i]}_{\mathsf{tc}})$ to $\mathcal{A}$. If $\mathcal{A}$ eventually outputs a forgery, $\sigma^\dagger = (\sigma^\dagger_{\mathsf{xsig}}, open^\dagger_{\mathsf{tc}}, com^\dagger_{\mathsf{tc}})$ and $m^\dagger$, it outputs $\sigma^\dagger_{\mathsf{xsig}} := \sigma^\dagger_{\mathsf{xsig}}$ and $m^\dagger := com^\dagger_{\mathsf{tc}}$ as a forgery with respect to xSIG.

Correctness of the above reduction holds because of the statistically close distribution of simulated $com^{[i]}_{\mathsf{tc}}$, and $open^{(j)}_{\mathsf{tc}}$. The output $(\sigma^\dagger_{\mathsf{xsig}}, m^\dagger)$ is also a valid forgery since

$com_{\mathsf{tc}}^{\dagger}$ differs from any $com_{\mathsf{tc}}^{[i]}$. Letting ϵ_{sim} denote the statistical distance, we have $|\Pr[\text{Game } 0] - \Pr[\text{Game } 1]| \leq \mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{xSIG},\mathcal{B}}(\lambda) + \epsilon_{\mathsf{sim}}$.

Now we claim that $\mathcal{A}$ winning in Game 1 occurs only if the CMTCR security of TC is broken. The reduction from successful $\mathcal{A}$ in Game 1 to adversary $\mathcal{C}$ that breaks TC is straightforward. Given ck_{tc}, $\mathcal{C}$ runs $(vk_{\mathsf{xsig}}, sk_{\mathsf{xsig}}) \leftarrow \mathsf{xSIG.Key}(gk)$ and invokes $\mathcal{A}$ with $vk := (vk_{\mathsf{xsig}}, ck_{\mathsf{tc}})$. Then, given message $m^{(j)}$, forward it to the oracle of TC and obtain $(com_{\mathsf{tc}}^{[i]}, open_{\mathsf{tc}}^{[i]})$. Then sign $com_{\mathsf{tc}}^{[i]}$ using sk_{xsig} to obtain $\sigma_{\mathsf{xsig}}^{[i]}$ and return $(\sigma_{\mathsf{xsig}}^{[i]}, open_{\mathsf{tc}}^{[i]}, com_{\mathsf{tc}}^{[i]})$ to $\mathcal{A}$. Given a forged signature $(\sigma_{\mathsf{xsig}}^{\dagger}, open_{\mathsf{tc}}^{\dagger}, com_{\mathsf{tc}}^{\dagger})$ and $m^{\dagger}$, output $open_{\mathsf{tc}}^{\dagger} := open_{\mathsf{tc}}^{\dagger}$ and $m^{\dagger} := m^{\dagger}$. It is a valid forgery since $m^{\dagger} \neq m^{[i]}$ for all i. We thus have $\Pr[\text{Game } 1] = \mathrm{Adv}^{\mathsf{cmtcr}}_{\mathsf{TC},\mathcal{C}}(\lambda)$.

By summing up the differences, we have

$$\mathrm{Adv}^{\mathsf{uf\text{-}cma}}_{\mathsf{FSP2},\mathcal{A}}(\lambda) \leq \mathrm{Adv}^{\mathsf{uf\text{-}xrma}}_{\mathsf{xSIG},\mathcal{B}}(\lambda) + \mathrm{Adv}^{\mathsf{cmtcr}}_{\mathsf{TC},\mathcal{C}}(\lambda) + \epsilon_{\mathsf{sim}}, \tag{16}$$

which proves the statement. □

To instantiate this construction with the building blocks from previous sections, we again need to duplicate $com_{\mathsf{mtc}} = \tilde{G}_u = \tilde{G}^{\zeta} \prod_{i=1}^{\ell} \tilde{X}_i^{m_i}$ to a triple with respect to bases $\tilde{G} = \tilde{F}_2, \tilde{F}_3$ and $\tilde{U}_1$ as follows. To be able to do so without holding the discrete logarithms of the $\tilde{X}_i$'s, we need to duplicate $\tilde{X}$ to the same set of bases as well. Details are shown in the following.

[**Procedure: Matching $\mathcal{C}_{\mathsf{mtc}}$ to $\mathcal{M}_{\mathsf{xsig}}$**]

Setup: It runs xSIG.Setup and sets $(F_1, \tilde{F}_1)$ as default generators $(G, \tilde{G})$ for MTC with extra generators (F_2, U_1) as well.

MTC.Key: On top of the legitimate procedure with $G := \tilde{F}_1$ to obtain $\tilde{X}_i := G^{\rho_i}$, additionally compute $\tilde{X}_{i2} := \tilde{F}_2^{\rho_i}$ and $\tilde{X}_{i3} := \tilde{U}_1^{\rho_i}$ for $i = 1, \ldots, \ell$ and include them to ck_{mtc}.

MTC.Com: On top of the legitimate procedure that computes $\tilde{G}_u = \tilde{G}^{\zeta} \prod_{i=1}^{\ell} \tilde{X}_i^{m_i}$ for $\tilde{G} := \tilde{F}_1$, compute $\tilde{G}_{u2} := \tilde{F}_2^{\zeta} \prod_{i=1}^{\ell} \tilde{X}_{i1}^{m_i}$ and $\tilde{G}_{u3} := \tilde{U}_1^{\zeta} \prod_{i=1}^{\ell} \tilde{X}_{i3}^{m_i}$ and include them to com_{mtc}.

MTC.SimCom: Compute the above extra components as $\tilde{G}_{u2} := \tilde{F}_2^{\omega_u}$, and $\tilde{G}_{u3} := U_1^{\omega_u}$.

The result is an extended commitment $com_{\mathsf{mtc}} = (\tilde{G}_u, \tilde{G}_{u2}, \tilde{G}_{u3})$ that matches the message space of xSIG with $\ell = 1$. Note that the duplicated keys have no effect on the security of POS nor MTC since they can be easily simulated when the discrete-logs of the extra bases to the original base $\tilde{G}$ are known.

We summarize the instantiation of FSP2 in the following. Let $k = \lceil \frac{L}{\ell_{\mathsf{pos}}} \rceil$ and $\ell_{\mathsf{mtc}} = 1 + k + \ell_{\mathsf{pos}}$.

Table 1. Size of a secret key, a verification key, a signature, and the number of PPEs in verification for a unilateral message of size $L = k\ell$. $(a, b) : a$ and b elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively.

Scheme	$\lvert sk\rvert$	$\lvert gk\rvert$	$\lvert vk\rvert$	$\lvert M\rvert$	$\lvert\sigma\rvert$	# PPE
FSP1	$(4, 0)$	$(5 + L, 5 + L)$	$(1, 7)$	$(L, 0)$	$(7, 7 + 3L)$	$7 + 2L$
FSP2	$(4, 0)$	$(4, 4)$	$(1, 10 + 3(k + \ell))$	$(0, L)$	$(7 + k + \ell, 4 + 2k)$	$5 + k$

Table 2. Concrete signature size for small messages with optimal setting of $k = \ell = \sqrt{L}$.

Scheme	$\lvert\sigma\rvert$					
	$L = k\ell$	1	4	9	25	100
FSP1	$(7, 7 + 3L)$	$(7, 10)$	$(7, 19)$	$(7, 34)$	$(7, 82)$	$(7, 307)$
FSP2	$(7 + k + \ell, 4 + 2k)$	$(9, 6)$	$(11, 8)$	$(13, 10)$	$(17, 14)$	$(27, 24)$

Common Parameter $(G, \tilde{G}, F_1, F_2, \tilde{F}_1, \tilde{F}_2, U_1, \tilde{U}_1)$

Public-key $(\tilde{V}_1, \tilde{V}_2, \tilde{V}_3, \tilde{V}_4, \tilde{V}_5, \tilde{V}_6, V_7, \tilde{V}_8, \{\tilde{X}_i, \tilde{X}_{i2}, \tilde{X}_{i3}\}_{i=1}^{\ell_{\mathsf{mtc}}})$

Secret-key (K_1, K_2, K_3, K_4)

Message $(\tilde{M}_1, \ldots, \tilde{M}_L)$

Signature $(\tilde{S}_0, S_1, \ldots, S_5, \tilde{G}_u, \tilde{G}_{u2}, \tilde{G}_{u3}, R, G_z, G_1, \ldots, G_{\ell_{\mathsf{pos}}}, \{A_i, \tilde{Z}_i, \tilde{R}_i\}_{i=1}^{k})$

Verification PPEs Let $(N_1, \ldots, N_{\ell_{\mathsf{mtc}}}) := (G_z, G_1, \ldots, G_{\ell_{\mathsf{pos}}}, A_1, \ldots, A_k)$.
For $j = 1, \ldots, k$:

$$e(A_j, \tilde{G}) = e(G_z, \tilde{Z}_j)\, e(G, \tilde{R}_j) \prod_{i=1}^{\ell_{\mathsf{pos}}} e(G_i, \tilde{M}_{(j-1)\ell_{\mathsf{pos}}+i}),$$

$$e(G, \tilde{G}_u) = e(R, \tilde{G}) \prod_{i=1}^{\ell_{\mathsf{mtc}}} e(N_i, \tilde{X}_i)$$

$$e(S_5, \tilde{V}_6\, \tilde{G}_{u3}) = e(G, \tilde{S}_0),$$

$$e(S_1, \tilde{V}_1)\, e(S_2, \tilde{V}_3)\, e(S_3, \tilde{V}_2) = e(S_4, \tilde{V}_4)\, e(S_5, \tilde{V}_5)\, e(V_7, \tilde{V}_8),$$

$$e(F_1, \tilde{G}_{u3}) = e(U_1, \tilde{G}_u), \quad e(F_2, \tilde{G}_{u3}) = e(U_1, \tilde{G}_{u2}).$$

4.3. *Efficiency*

In this section, we assess the efficiency of FSP1 and FSP2 instantiated as described in Sects. 4.1 and 4.2. Note that FSP1 uses a one-time signature scheme, OTS, and we evaluate the efficiency where OTS is instantiated by POS in Sect. 2.5 since the POS is the best known structure-preserving OTS under a standard assumption.

Signature Size and Number of PPEs In Table 1 we assess the sizes of a key and a signature for unilateral messages consisting of ℓ group elements. By $\lvert vk\rvert$, we denote the number of group elements in vk except for those in gk. Similarly, by $\lvert sk\rvert$, we denote the number of group elements in sk except for those in vk. By the term $\#\,\mathsf{PPE}_{\mathsf{A}}$ we denote the number of pairing product equations in the corresponding building block A. Table 2 summarizes the comparison with signature length for some concrete message lengths. In the following, we denote the size of an element by (a, b) when the element consists of a and b group elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively.

- FSP1. According to the descriptions in Sects. 2.5 and 2.6, we have the following parameters for the building blocks.
  - OTS: $|vk_{\mathsf{ots}}| = |vk_{\mathsf{pos}}| + |ovk_{\mathsf{pos}}| = (0, L+2)$, $|\sigma_{\mathsf{ots}}| = (2,0)$, and $\#\,\mathsf{PPE}_{\mathsf{ots}} = 1$.
  - xSIG: $|sk_{\mathsf{xsig}}| = (4,0)$, $|vk_{\mathsf{xsig}}| = (1,7)$, and $\#\,\mathsf{PPE}_{\mathsf{xsig}} = 2 + 2\,|vk_{\mathsf{ots}}|$.

  The common setup function for these building blocks generates bases $(G, \tilde{G}, F_1, F_2, \tilde{F}_1, \tilde{F}_2, \{U_i, \tilde{U}_i\}_{i=1}^{\ell})$ for $\ell_{\mathsf{xsig}} = |vk_{\mathsf{ots}}|$ to allow xSIG to sign vk_{ots}. (Note that vk_{ots} consists only of group elements from $\mathbb{G}$, which xSIG can sign.) Taking the message expansion factor $\mu = 3$ into account, we obtain the following for FSP1:

$$
\begin{aligned}
|gk| &= (3 + |vk_{\mathsf{ots}}|, 3 + |vk_{\mathsf{ots}}|) = (5+L, 5+L)\\
|sk| &= |sk_{\mathsf{xsig}}| = (4,0)\\
|vk| &= |vk_{\mathsf{xsig}}| = (1,7)\\
|\sigma| &= |\sigma_{\mathsf{xsig}}| + |\sigma_{\mathsf{ots}}| + \mu\,|vk_{\mathsf{ots}}| = (5,1) + (2,0) + (0, 6+3L)\\
&= (7, 7+3L)\\
\#\,\mathsf{PPE} &= \#\,\mathsf{PPE}_{\mathsf{xsig}} + \#\,\mathsf{PPE}_{\mathsf{ots}} = 7 + 2L
\end{aligned}
$$

- FSP2. The underlying components are xSIG, MTC and POS. Since POS is repeatedly used in FSP2, its message size ℓ_{pos} can be set independently from the input message size ℓ. The parameters for these underlying components are:
  - POS: $|vk_{\mathsf{pos}}| = (\ell_{\mathsf{pos}} + 1, 0)$, $|ovk_{\mathsf{pos}}| = (1,0)$, $|\sigma_{\mathsf{pos}}| = (0,2)$, and $\#\,\mathsf{PPE}_{\mathsf{pos}} = 1$.
  - MTC: $|ck_{\mathsf{mtc}}| = |vk_{\mathsf{pos}}| + |L/\ell_{\mathsf{pos}}| \cdot |ovk_{\mathsf{pos}}| = (0, 1+k+\ell_{\mathsf{pos}})$, $|com_{\mathsf{mtc}}| = (0,1)$, and $|open_{\mathsf{mtc}}| = (1,0)$.
  - xSIG: $|sk_{\mathsf{xsig}}| = (4,0)$, $|vk_{\mathsf{xsig}}| = (1,7)$, and $\#\,\mathsf{PPE}_{\mathsf{xsig}} = 2 + 2\,|com_{\mathsf{mtc}}|$.

  As in the previous case, the common setup function outputs gk including bases $(G, \tilde{G}, F_1, F_2, \tilde{F}_1, \tilde{F}_2, \{U_i, \tilde{U}_i\}_{i=1}^{\ell_{\mathsf{xsig}}})$ for $\ell_{\mathsf{xsig}} = |com_{\mathsf{mtc}}|$ to allow xSIG to sign com_{mtc}. Based on these parameters, the following evaluation is obtained for FSP2:

$$
\begin{aligned}
|sk| &= |sk_{\mathsf{xsig}}| = (4,0)\\
|gk| &= (4,4)\\
|vk| &= |gk| + |vk_{\mathsf{xsig}}| + |ck_{\mathsf{mtc}}| = (1,7) + (0, 3 + 3(k+\ell))\\
&= (1, 10 + 3k + 3\ell)\\
|\sigma| &= |\sigma_{\mathsf{xsig}}| + |open_{\mathsf{mtc}}| + |\sigma_{\mathsf{pos}}| + \mu|com_{\mathsf{mtc}}| + |vk_{\mathsf{pos}}| + |L/\ell_{\mathsf{pos}}| \cdot |ovk_{\mathsf{pos}}|\\
&= (5,1) + (k,0) + (0,2k) + (0,3) + (\ell+1, 0) + (1,0)\\
&= (7 + k + \ell, 4 + 2k)\\
\#\,\mathsf{PPE} &= \#\,\mathsf{PPE}_{\mathsf{xsig}} + \#\,\mathsf{PPE}_{\mathsf{mtc}} + |L/\ell_{\mathsf{pos}}| \cdot \#\,\mathsf{PPE}_{\mathsf{pos}}\\
&= 5 + |L/\ell_{\mathsf{pos}}| = 5 + k
\end{aligned}
$$

The last equality in each evaluation is obtained at the optimal setting; $\ell_{\mathsf{pos}} = |L/\ell_{\mathsf{pos}}| = k$.

Table 3. Size of a Groth–Sahai zero-knowledge proof of knowledge for a secret key or a signature for unilateral messages of size L with the optimal parameter setting. (x, y, z) denotes x and y elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively, and z elements in $\mathbb{Z}_p$.

Scheme	Proof of sk	Proof of σ
FSP1	$(10, 10, 2)$	$(32 + 4L, 24 + 6L, 2)$
FSP2	$(10, 10, 2)$	$(28 + 6k + 2\ell, 18 + 8k, 2)$

Proof Size Next we assess the cost for proving one's knowledge of a secret key or a signature for FSP1 and FSP2 with the Groth–Sahai proof as a non-interactive zero-knowledge proof. Results are summarized in Table 3.

Proof of Knowing a Secret Key Recall that, in either scheme, a secret key (K_1, K_2, K_3, K_4) is correct if it satisfies relations in (9). To allow zero-knowledge simulation, the relations are transformed into the following form:

$$\begin{aligned} &e(\underline{K_2}, \tilde{G}) = e(\underline{G}, \tilde{V}_1), \quad e(\underline{G}, \tilde{V}_3) = e(\underline{K_2}, \tilde{V}_2), \\ &e(\underline{K_1}, \tilde{V}_1) = e(\underline{W}, \tilde{V}_8), \quad \underline{W} = V_7, \\ &e(\underline{K_2}, \tilde{V}_4) = e(\underline{G}, \tilde{V}_5), \quad e(\underline{K_3}, \tilde{G})e(\underline{K_4}, \tilde{V}_2) = e(\underline{G}, \tilde{V}_4). \end{aligned} \tag{17}$$

Underlined variables are the witnesses the prover commits to. Observe that (17) consists of five linear PPEs and a linear multiscalar multiplication equation. According to [38], committing to a group element in $\mathbb{G}$ requires 2 elements in $\mathbb{G}$, and proving a linear PPE and a multiscalar multiplication equation yield a proof consisting of 2 group elements in $\tilde{\mathbb{G}}$, and $2 \times 1 = 2$ scalar values in $\mathbb{Z}_p$, respectively. Committing to G can be done for free by using a prescribed default commitment as suggested in [29]. Thus, with five witnesses, five linear PPEs, and one linear multiscalar multiplication equation, the resulting proof (i.e., commitments and proofs for all relations) consists of 10 elements in $\mathbb{G}$, 10 elements in $\tilde{\mathbb{G}}$, and 2 elements in $\mathbb{Z}_p$.

Proof of Knowing a Valid Signature We first consider FSP1. According to the descriptions in Sect. 4.1, a valid signature satisfies the following relations.

$$\begin{aligned} &e(G, \underline{\tilde{A}}) = e(\underline{Z}, \underline{\tilde{G}_z})\, e(\underline{R}, \tilde{G}) \prod_{i=1}^{L} e(M_i, \underline{\tilde{G}_i}), \quad e(\underline{S_5}, \tilde{V}_6\, \underline{\tilde{A}_3}\, \underline{\tilde{G}_{z3}} \prod_{i=1}^{L} \underline{\tilde{G}_{i3}}) = e(G, \underline{\tilde{S}_0}), \\ &e(\underline{S_1}, \tilde{V}_1)\, e(\underline{S_2}, \tilde{V}_3)\, e(\underline{S_3}, \tilde{V}_2) = e(\underline{S_4}, \tilde{V}_4)\, e(\underline{S_5}, \tilde{V}_5)\, e(\underline{W}, \tilde{V}_8), \quad \underline{W} = V_7, \\ &e(F_1, \underline{\tilde{A}_3}) = e(U_{\ell+2}, \underline{\tilde{A}}), \quad e(F_2, \underline{\tilde{A}_3}) = e(U_{\ell+2}, \underline{\tilde{A}_2}), \quad e(F_1, \underline{\tilde{G}_{z3}}) = e(U_{\ell+1}, \underline{\tilde{G}_z}), \\ &e(F_2, \underline{\tilde{G}_{z3}}) = e(U_{\ell+1}, \underline{\tilde{G}_{z2}}), \quad e(F_1, \underline{\tilde{G}_{i3}}) = e(U_i, \underline{\tilde{G}_i}), \quad e(F_2, \underline{\tilde{G}_{i3}}) = e(U_i, \underline{\tilde{G}_{i2}}) \end{aligned}$$

for $i = 1, \ldots, L$ for the last two relations. There are 8 underlined witnesses in $\mathbb{G}$ and $7 + 3L$ in $\tilde{\mathbb{G}}$. Committing to these witnesses requires 16 elements in $\mathbb{G}$ and $14 + 6L$ elements in $\tilde{\mathbb{G}}$. The first two relations involve witnesses in both groups whose proofs require 2×4 elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$. The third relation has witnesses only in $\mathbb{G}$. Its proof consists of 2 elements in $\tilde{\mathbb{G}}$. The fourth relation is a linear multiscalar multiplication

equation whose proof consists of 2 elements in $\mathbb{Z}_p$. The remaining $4+2L$ relations have witnesses only in $\tilde{\mathbb{G}}$, and each of their proof costs 2 elements in $\mathbb{G}$. In total the proofs and commitments consist of $16+4\times 2+2\times(4+2L)=32+4L$ elements in $\mathbb{G}$ and $14+6L+4\times 2+2=24+6L$ elements in $\tilde{\mathbb{G}}$, and 2 elements in $\mathbb{Z}_p$.

Next consider FSP2. As described in Sect. 4.2, a valid signature satisfies the following relations:

$$
\begin{aligned}
&e(\underline{A_j}, \tilde{G}) = e(\underline{G_z}, \underline{\tilde{Z}_j})\, e(G, \underline{\tilde{R}_j}) \prod_{i=1}^{\ell_{\mathsf{pos}}} e(\underline{G_i}, \tilde{M}_{(j-1)\ell_{\mathsf{pos}}+i}) \text{ (for } j=1,\ldots,k),\\
&e(G, \underline{\tilde{G}_u}) = e(\underline{R}, \tilde{G}) \prod_{i=1}^{\ell_{\mathsf{mtc}}} e(\underline{N_i}, \tilde{X}_i), \quad e(\underline{S_5}, \tilde{V}_6\, \underline{\tilde{G}_{u3}}) = e(G, \underline{\tilde{S}_0}),\\
&e(\underline{S_1}, \tilde{V}_1)\, e(\underline{S_2}, \tilde{V}_3)\, e(\underline{S_3}, \tilde{V}_2) = e(\underline{S_4}, \tilde{V}_4)\, e(\underline{S_5}, \tilde{V}_5)\, e(\underline{W}, \tilde{V}_8), \quad \underline{W} = V_7,\\
&e(F_1, \underline{\tilde{G}_{u3}}) = e(U_1, \underline{\tilde{G}_u}), \quad e(F_2, \underline{\tilde{G}_{u3}}) = e(U_1, \underline{\tilde{G}_{u2}})
\end{aligned}
$$

where $(N_1,\ldots,N_{\ell_{\mathsf{mtc}}})$ is actually $(G_z, G_1,\ldots,G_{\ell_{\mathsf{pos}}}, A_1,\ldots,A_k)$ that are also witnesses. Thus we do not need to count the cost for committing to N_i. We consider $\ell_{\mathsf{mtc}}=k=\ell$. A signature consists of $7+k+\ell$ elements in $\mathbb{G}$ and $4+2k$ elements in $\tilde{\mathbb{G}}$. Thus committing to the signature costs $2(7+k+\ell)$ and $2(4+2k)$ elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively. To achieve zero-knowledge, the prover also commits to V_7 with W, which costs 2 elements in $\mathbb{G}$. The first three relations (indeed $k+2$ relations) that came from POS and MTC involve witnesses in both groups. Hence proofs for them cost $(k+2)(4,4)$ elements in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively. The multiscalar multiplication equation for V_7 costs two elements in $\mathbb{Z}_p$. The remaining three relations that came from xSIG involves witnesses for either of $\mathbb{G}$ or $\tilde{\mathbb{G}}$. Proofs for those relations costs 2 group elements in $\tilde{\mathbb{G}}$ and 2×2 group elements in $\mathbb{G}$. In total the proofs and commitments consist of $2(7+k+\ell)+2+4(k+2)+4=28+6k+2\ell$ and $2(4+2k)+4(k+2)+2=18+8k$ in $\mathbb{G}$ and $\tilde{\mathbb{G}}$, respectively, and 2 elements in $\mathbb{Z}_p$. Accordingly, for any setting of k and ℓ satisfying $L=k\ell$, FSP2 retains better efficiency over FSP1.

5. Efficient Fully Structure-Preserving Combined Signatures

We will now construct a fully structure-preserving combined signature scheme SP1 that can be used to sign messages consisting of $L=\ell k$ group elements in $\tilde{\mathbb{G}}$. We strive for high efficiency and to optimize performance we settle for a proof of security in the generic asymmetric bilinear group model. We proceed in two steps, first we construct a (not fully) structure-preserving signature scheme and then later modify it to a fully structure-preserving signature scheme.

5.1. *Starting Point: A Structure-Preserving Combined Signature Scheme*

In this section, we construct a structure-preserving combined signature scheme SP1 that can be used to sign messages consisting of $L=\ell k$ group elements in $\tilde{\mathbb{G}}$. The signature

and verification algorithms for randomizable and strongly unforgeable signatures, respectively, are quite similar. We therefore describe them at the same time indicating the choice by $b = 0$ for randomizable signatures and $b = 1$ for strongly unforgeable ones.

In order to explain some of the design principles underlying the construction, let us first consider the special case where the message space is $\tilde{\mathbb{G}}$, i.e., we are signing a single group element and $L = \ell = k = 1$. The setup includes a random group element $\tilde{Y} = \tilde{G}^y \in \tilde{\mathbb{G}}$, the verification key consists of a single group element $V = G^v \in \mathbb{G}$, and both randomizable and strongly unforgeable signatures are of the form $\sigma = (R, \tilde{S}, \tilde{T}) \in \mathbb{G} \times \tilde{\mathbb{G}}^2$.

For a randomizable signature, there will be two verification equations:

$$e(R, \tilde{S}) = e(G, \tilde{Y})e(V, \tilde{G}) \qquad e(R, \tilde{T}) = e(G, \tilde{M})e(V, \tilde{Y}).$$

It is easy to see that we can randomize the factors in $e(R, \tilde{S})$ and $e(R, \tilde{T})$ into $e(R^{\frac{1}{\beta}}, \tilde{S}^{\beta})$ and $e(R^{\frac{1}{\beta}}, \tilde{T}^{\beta})$ without changing the products themselves, which gives us randomizability of the signatures.

The first verification equation is designed to prevent the adversary from creating a forged signature from scratch after seeing the verification key only. An adversary *using only generic group operations* can do no better than computing $R = G^{\rho} V^{\rho_v}$ and $\tilde{S} = \tilde{G}^{\sigma} \tilde{Y}^{\sigma_y}$ using known scalars $\rho, \rho_v, \sigma, \sigma_y \in \mathbb{Z}_p$. Looking at the underlying discrete logarithms, the first verification equation then corresponds to the polynomial equation

$$(\rho + \rho_v v)(\sigma + \sigma_y y) = y + v$$

in the unknown discrete logarithms v and y. Let us first argue that this equation is not solvable when viewing it as a formal polynomial equation in v, y. Looking at the coefficients of the term v, we get $\rho_v \sigma = 1$, which means $\sigma \neq 0$. Looking at coefficients of the term y we get $\rho\sigma_y = 1$ we get $\rho \neq 0$. But this leaves us with a constant term $\rho\sigma \neq 0$ and therefore we cannot solve the equation formally. On the other hand, in the generic group model the random encoding of group elements mean that the adversary has no further information about the actual values of v, y that are chosen at random, so the Schwartz–Zippel lemma implies it has a negligible probability $\frac{2}{p}$ of guessing $\rho, \rho_v, \sigma, \sigma_y$ such that the equation holds for the concrete discrete logarithms v, y.

What if the adversary instead of creating a signature from scratch tries to modify an existing signature or combine many existing signatures? Due to the randomness in the choice of $z \leftarrow \mathbb{Z}_p^*$ in the signing protocol each signature query will return a signature with a random R_i. As it turns out the randomization used in each signature makes it hard for the adversary to combine multiple signatures, or even modify one signature, in a meaningful way with generic group operations. Intuitively this is because generic group operations allow the adversary to compute linear combinations of elements it has seen; however, the verification equations are quadratic.

Let us now turn to the other option, to make strongly existentially unforgeable signature. In order to prevent randomization when strong unforgeability is desired, the combined signature scheme modifies the latter verification equation by including also

$e(V, \tilde{S})$. This gives us the following verification equations for strongly unforgeable signatures

$$e(R, \tilde{S}) = e(G, \tilde{Y})e(V, \tilde{G}) \qquad e(R, \tilde{T}) = e(G, \tilde{M})e(V, \tilde{Y})e(V, \tilde{S})$$

Now the randomization technique fails because a randomization of $\tilde{S}$ means we must change $\tilde{T}$ in a way that counteracts this change in the second verification equation. However, $\tilde{T}$ is paired with R that also changes when $\tilde{S}$ changes. The adversary is therefore faced with a nonlinear modification of the signatures and gets stuck because generic group operations only enable it to do linear modifications of signature elements.

We can extend the one-element signature scheme to sign a vector $\tilde{\boldsymbol{M}}_{[1]} = (\tilde{M}_{(1,1)}, \ldots, \tilde{M}_{(\ell,1)})$ with ℓ group elements in $\tilde{\mathbb{G}}$ by extending the verification key by $\ell - 1$ random group elements $\boldsymbol{U} = (U_1, \ldots U_{\ell-1})$. Now the verification equations become

$$e(R, \tilde{S}) = e(G, \tilde{Y})e(V, \tilde{G}) \qquad e(R, \tilde{T}) = \prod_{i=1}^{\ell-1} e(U_i, \tilde{M}_{(i,1)}) \cdot e(G, M_{(\ell,1)}) \cdot e(V, \tilde{Y}) \cdot e(V, \tilde{S})^b$$

where $b = 0$ for a randomizable signature and $b = 1$ for a strong signature. The idea is that the discrete logarithms of the elements in $\boldsymbol{U}$ are unknown to the adversary making it hard to change any group elements in a previously signed message to get a new message that will verify under the same signature.

Finally, to sign $L = \ell k$ group elements in $\tilde{\mathbb{G}}$ instead of ℓ group elements we keep the first verification equation, which does not involve the message, but add $k - 1$ extra verification equations similar to the second verification equation for a vector of group elements described above. This allows us to sign k vectors in parallel. In order to avoid linear combinations of message vectors and signature components being useful in other verification equations, we give each verification equation a separate $e(V, \tilde{Y}_j)$ factor, where $j = 1, \ldots, k$ is the index of the verification equation. The resulting signature scheme is given below.

[**Combined SPS :** SP1]

Setup(1^λ): Generate $(p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G}) \leftarrow \mathcal{G}(1^\lambda)$ and $\tilde{\boldsymbol{Y}} \leftarrow \tilde{\mathbb{G}}^k$. Return $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G}, \tilde{\boldsymbol{Y}})$. Message space $\mathcal{M}$ is set to $\tilde{\mathbb{G}}^{\ell \times k}$ for prescribed ℓ and k, and we represent a message in $\mathcal{M}$ as a matrix of size $\ell \times k$ over $\tilde{\mathbb{G}}$ hereafter.

Key(gk): Pick $\boldsymbol{u} \leftarrow \mathbb{Z}_p^{\ell-1}$ and $v \leftarrow \mathbb{Z}_p$. Output (vk, sk) where $vk := (\boldsymbol{U}, V) = (G^{\boldsymbol{u}}, G^v)$ and $sk := (\boldsymbol{u}, v)$.

Sign$_b(sk, \tilde{\boldsymbol{M}})$: Pick $z \leftarrow \mathbb{Z}_p^*$ and set $r = \frac{1}{z}$. Compute $\tilde{S} := (\tilde{Y}_1 \tilde{G}^v)^z$, $R := G^r$. For $j = 1 \cdots k$, compute $\tilde{T}_j := \left(\prod_{i=1}^{\ell-1} \tilde{M}_{(i,j)}^{u_i} \cdot \tilde{M}_{(\ell,j)} \cdot \tilde{Y}_j^v \cdot \tilde{S}^{vb}\right)^z$ and set $\tilde{\boldsymbol{T}} := (\tilde{T}_1, \cdots, \tilde{T}_k)$. Output $\sigma := (R, \tilde{S}, \tilde{\boldsymbol{T}})$.

$\mathsf{Vrf}_b(vk, \tilde{\boldsymbol{M}}, \sigma)$: Parse $\sigma = (R, \tilde{S}, \tilde{\boldsymbol{T}})$. Return 1 if and only if $\tilde{\boldsymbol{M}} \in \tilde{\mathbb{G}}^{\ell \times k}$, $R \in \mathbb{G}$, $\tilde{S} \in \tilde{\mathbb{G}}$, $\tilde{\boldsymbol{T}} \in \tilde{\mathbb{G}}^k$, and

$$e(R, \tilde{S}) = e(G, \tilde{Y}_1)\, e(V, \tilde{G}), \text{ and} \tag{18}$$

$$e(R, \tilde{T}_j) = \prod_{i=1}^{\ell-1} e(U_i, \tilde{M}_{(i,j)})\, e(G, \tilde{M}_{(\ell,j)})\, e(V, \tilde{Y}_j)\, e(V, \tilde{S})^b \text{ for all } j = 1, \ldots, k. \tag{19}$$

$\mathsf{Rand}(vk, \tilde{\boldsymbol{M}}, \sigma)$: Parse σ into $(R, \tilde{S}, \tilde{\boldsymbol{T}})$. Pick $\beta \leftarrow \mathbb{Z}_p^*$, calculate $R' := R^{\frac{1}{\beta}}$, $\tilde{S}' := \tilde{S}^{\beta}$, and $\tilde{\boldsymbol{T}}' := \tilde{\boldsymbol{T}}^{\beta}$. Return $\sigma' := (R', \tilde{S}', \tilde{\boldsymbol{T}}')$.

Theorem 9. SP1 *is structure-preserving combined signature scheme that is combined existentially unforgeable under chosen-message attack (*C-EUF-CMA *secure) in the generic group model.*

Proof. Perfect correctness, perfect randomizability and structure preservation follow by inspection. What remains is to prove that the signature scheme is C-EUF-CMA secure in the generic bilinear group model. In the (Type-III) generic bilinear group model, the adversary may compute new group elements in either source group by taking arbitrary linear combinations of previously seen group elements in the same source group. We shall see that no such linear combination of group elements, viewed as formal Laurent polynomials in the variables picked by the key generator and the signing oracle, yields an existential forgery. It follows along the lines of the Uber assumption of Boneh, Boyen and Goh [19] from the inability to produce forgeries when working with formal Laurent polynomials that the signature scheme is C-EUF-CMA secure in the generic bilinear group model.

Let $\tilde{\boldsymbol{M}}_i = \tilde{G}^{\mathbf{W}_i} \in \tilde{\mathbb{G}}^{\ell \times k}$ for $\mathbf{W}_i \in \mathbb{Z}_p^{\ell \times k}$ be the i-th ($0 \le i \le q$) signing query made by the adversary. The group elements in the message may be constructed by combining previously seen group elements, so $\mathbf{W}_i$ may depend linearly on the discrete logarithms of public key elements in $\tilde{\mathbb{G}}$ and all previously seen signature elements in $\tilde{S}_j, \tilde{\boldsymbol{T}}_j$ for $j < i$. The adversary obtains signatures $(R_i, \tilde{S}_i, \tilde{\boldsymbol{T}}_i)$ that

$$R_i = G^{\frac{1}{z_i}} \qquad \tilde{S}_i = (\tilde{Y}_1 \tilde{G}^v)^{z_i} \qquad \tilde{\boldsymbol{T}}_i = \tilde{G}^{z_i((\boldsymbol{u},1)\mathbf{W}_i + v\boldsymbol{y} + b_i z_i v(y_1+v))\,\mathbf{1})}$$

where $b_i = 0$ if query i is for a randomizable signature and $b_i = 1$ if query i is for a strong signature.

Viewed as Laurent polynomials, we have that the discrete logarithm of a signature $(R, \tilde{S}, \tilde{\boldsymbol{T}})$ generated by the adversary on a message $\tilde{\boldsymbol{M}}^{\ell \times k}$ defined by $\mathbf{W} \in \mathbb{Z}_p^{\ell \times k}$ is of the form

$$r = \rho + v\rho_v + \boldsymbol{u}\boldsymbol{\rho}_u^\top + \sum_i \frac{1}{z_i}\rho_{r_i}$$

$$s = \sigma + \boldsymbol{\sigma}_y \boldsymbol{y}^\top + \sum_j \sigma_{s_j} z_j (y_1 + v) + \sum_j \boldsymbol{\sigma}_{t_j} z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j z_j v(y_1 + v)\mathbf{1}\right)$$

$$\boldsymbol{t} = \boldsymbol{\tau} + \boldsymbol{y} T_y + \sum_j z_j (y_1 + v)\boldsymbol{\tau}_{s_j} + \sum_j z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j z_j v(y_1 + v)\mathbf{1}\right) T_{t_j}$$

Similarly, all ℓk entries in $\mathbf{W}$ can be written in a form similar to s, and all entries in queried messages with discrete logarithms $\mathbf{W}_i$ can be written in a form similar to s where the sums are bounded by $j < i$.

For the first verification equation to be satisfied, we must have $rs = y_1 + v$, i.e.,

$$\begin{pmatrix} \rho + \boldsymbol{u}\boldsymbol{\rho}_u^\top \\ +v\rho_v + \sum_i \frac{1}{z_i}\rho_{r_i} \end{pmatrix} \begin{pmatrix} \sigma + \boldsymbol{\sigma}_y \boldsymbol{y}^\top + \sum_j \sigma_{s_j} z_j (y_1 + v) \\ +\sum_j \boldsymbol{\sigma}_{t_j} z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + v)\mathbf{1}\right)^\top \end{pmatrix} = y_1 + v$$

We start by noting that $r \neq 0$ since otherwise the left hand side multivariate polynomial rs cannot have the term y_1 that appears on the right hand side. Please observe that it is only in $\mathbb{G}$ that we have terms including indeterminates with negative power, i.e., $\frac{1}{z_i}$. In $\tilde{\mathbb{G}}$ all indeterminates have positive power, i.e., so $s_j, \boldsymbol{t}_j, \mathbf{W}_j$ only contain proper multivariate polynomials. Now suppose for a moment that $\rho_{r_i} = 0$ for all i. Then in order not to have a terms involving z_j's in rs we must have $\sum_j \sigma_{s_j} z_j (y_1 + v) + \sum_j \boldsymbol{\sigma}_{t_j} z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + v)\mathbf{1}\right)^\top = 0$. The term y_1 now gives us $\rho\sigma_{y,1} = 1$ and the term v gives us $\rho_v \sigma = 1$. This means $\rho \neq 0$ and $\sigma \neq 0$, and therefore, we reach a contradiction since the constant term should be $\rho\sigma = 0$. We conclude that there must exist some J for which $\rho_{r_J} \neq 0$.

Now we have the term $\rho_{r_J} \sigma \frac{1}{z_J} = 0$, which shows us $\sigma = 0$. The terms $\rho_{r_J} \sigma_{y,h} \frac{y_h}{z_J} = 0$ for $h = 1, \ldots, k$ give us $\boldsymbol{\sigma}_y = \mathbf{0}$.

The polynomials corresponding to s_j and $\boldsymbol{t}_j$ contain the indeterminate z_j in all terms, so no linear combination of them can give us a term where the indeterminate component is vy_h for some $h \in \{1, \ldots, k\}$. Since M_j is constructed as a linear combination of elements in the verification key and components in $\tilde{\mathbb{G}}$ from previously seen signatures, it too cannot contain a term where the indeterminate component is vy_h. The coefficient of $\frac{z_j}{z_J} v y_h$ is therefore $\rho_{r_J} \sigma_{t_j,h} = 0$ and therefore $\sigma_{t_j,h} = 0$ for every $j \neq J$ and $h \in \{1, \ldots, k\}$. This shows $\boldsymbol{\sigma}_{t_j} = \mathbf{0}$ for all $j \neq J$. Looking at the coefficients for vy_h for $h = 1, \ldots, k$ we see that $\boldsymbol{\sigma}_{t_J} = \mathbf{0}$ too.

The terms $\rho_{r_J} \sigma_{s_j} \frac{z_j}{z_l} v$ give us $\sigma_{s_j} = 0$ for all $j \neq J$. In order to get a coefficient of 1 for the term y_1 we see that $\sigma_{s_J} = \frac{1}{\rho_{r_J}}$, which is nonzero. Our analysis has now shown that

$$s = \frac{1}{\rho_{r_J}} z_J (y_1 + v).$$

Let us now analyze the structure of r. The term $\rho_v \sigma_J v^2 z_J = 0$ gives us $\rho_v = 0$. We know from our previous analysis that if there was a second $i \neq J$ for which $\rho_{r_i} \neq 0$

then also $\sigma_{\rho_J} = 0$, which it is not. Therefore for all $i \neq J$ we have $\rho_{r_i} = 0$. The term $\rho\sigma_{s_J} z_J y_1$ gives $\rho = 0$. The terms in $\sigma_{s_J} \boldsymbol{u} z_J v \boldsymbol{\rho}_u^\top$ give us $\boldsymbol{\rho}_u = \mathbf{0}$. Our analysis therefore shows

$$r = \rho_{r_J} \frac{1}{z_J}.$$

We now turn to the second verification equation, which is $rt_1 = (\boldsymbol{u}, 1)\mathbf{w}^\top + vy_1 + bvs$, where $\mathbf{w}^\top$ is the first column vector of $\mathbf{W}$. The message vector is of the form

$$\begin{aligned}\mathbf{w} = \boldsymbol{\mu} + \boldsymbol{y}\mathbf{W}_y + \sum_j \boldsymbol{\mu}_{s_j} z_j (y_1 + v) \\ + \sum_j z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + v)\mathbf{1}\right) \mathbf{W}_{t_j}\end{aligned}$$

where $\boldsymbol{\mu}, \mathbf{W}_y \boldsymbol{\mu}_{s_j}$ and $\mathbf{W}_{t_j}$ are vectors and matrices of corresponding size with entries in $\mathbb{Z}_p$ chosen by the adversary. Similarly, we can write out $t_1 = \tau + \boldsymbol{\tau}_y \boldsymbol{y}^\top + \sum_j \tau_{s_j} z_j (y_1 + v) + \sum_j \boldsymbol{\tau}_{t_j} z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + v)\mathbf{1}\right)$ for elements and vectors of corresponding size $\tau, \boldsymbol{\tau}_y, \tau_{s_j}, \boldsymbol{\tau}_{t_j}$ with entries in $\mathbb{Z}_p$ chosen by the adversary.

Writing out the second verification equation, we have

$$\begin{aligned}&\rho_{r_J} \frac{1}{z_J} \begin{pmatrix} \tau + \boldsymbol{\tau}_y \boldsymbol{y}^\top + \sum_j \tau_{s_j} z_j (y_1 + v) \\ + \sum_j \boldsymbol{\tau}_{t_j} z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + v)\mathbf{1}\right) \end{pmatrix} \\ &= vy_1 + bv \left(\frac{1}{\rho_{r_J}} z_J (y_1 + v)\right) \\ &\quad + (\boldsymbol{u}, 1) \begin{pmatrix} \boldsymbol{\mu} + \boldsymbol{y}\mathbf{W}_y + \sum_j \boldsymbol{\mu}_{s_j} z_j (y_1 + v) \\ + \sum_j z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + v)\mathbf{1}\right) \mathbf{W}_{t_j} \end{pmatrix}^\top.\end{aligned}$$

Looking at the coefficients of terms involving $\frac{1}{z_J}$ and $\frac{y_h}{z_J}$, we get $\tau = 0$ and $\boldsymbol{\tau}_y = \mathbf{0}$. Looking at the terms in $\rho_{r_J} \boldsymbol{\tau}_{t_j} \frac{z_j}{z_J} v\boldsymbol{y}$, we get $\boldsymbol{\tau}_{t_j} = \mathbf{0}$ for all $j \neq J$. Similarly, the terms $\rho_{r_J} \tau_{s_j} \frac{z_j}{z_J} v$ give us $\tau_{s_j} = 0$ for all $j \neq J$. We are now left with

$$\begin{aligned}&\rho_{r_J} \left(\tau_{s_J} (y_1 + v) + \boldsymbol{\tau}_{t_J} \left((\boldsymbol{u}, 1)\mathbf{W}_J + v\boldsymbol{y} + b_J v z_J (y_1 + v)\mathbf{1}\right)\right) \\ &= vy_1 + bv \frac{1}{\rho_{r_J}} z_J (y_1 + v) \\ &\quad + (\boldsymbol{u}, 1) \begin{pmatrix} \boldsymbol{\mu} + \boldsymbol{y}\mathbf{W}_y + \sum_j \boldsymbol{\mu}_{s_j} z_j (y_1 + v) \\ + \sum_j z_j \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + v)\mathbf{1}\right) \mathbf{W}_{t_j} \end{pmatrix}^\top.\end{aligned}$$

Terms involving z_j and z_j^2 must cancel out, so we can assume $\boldsymbol{\mu}_{s_j} = \mathbf{0}$ and $\mathbf{W}_{t_j} = 0$ for $j > J$. Since $\mathbf{W}_J$ does not involve z_J in any of its terms, we get from the terms in $(\boldsymbol{u}, 1) z_J v \boldsymbol{\mu}_{s_J}^\top$ that $\boldsymbol{\mu}_{s_J} = 0$. Since there can be no terms involving z_J^2 we get $b_J \mathbf{1} \mathbf{W}_{t_J}^\top = \mathbf{0}$. Looking at the coefficients for v we get $\tau_{s_J} = 0$. This leaves us with

$$\begin{aligned}
&\rho_{r_J}\boldsymbol{\tau}_{t_J}\left((\boldsymbol{u},1)\mathbf{W}_J+v\boldsymbol{y}+b_J v z_J(y_1+v)\mathbf{1}\right)^\top\\
&=vy_1+bv\frac{1}{\rho_{r_J}}z_J(y_1+v)+(\boldsymbol{u},1)z_J\left((\boldsymbol{u},1)\mathbf{W}_J+v\boldsymbol{y}\right)\mathbf{W}_{t_J}\Big)^\top\\
&\quad+(\boldsymbol{u},1)\begin{pmatrix}\boldsymbol{\mu}+\boldsymbol{y}\mathbf{W}_y+\sum_{j<J}\boldsymbol{\mu}_{s_j}z_j(y_1+v)\\ +\sum_{j<J}z_j\left((\boldsymbol{u},1)\mathbf{W}_j+v\boldsymbol{y}+b_j v z_j(y_1+v)\mathbf{1}\right)\mathbf{W}_{t_j}\end{pmatrix}^\top.
\end{aligned}$$

Looking at the terms involving $z_J v^2$ we see $\rho_{r_J}\boldsymbol{\tau}_{t_J}b_J\mathbf{1}^\top=b\frac{1}{\rho_{r_J}}$. This cancels out the first two parts involving z_J. The only remaining terms involving z_J now give us $\mathbf{W}_{t_J}=0$. This gives us

$$\begin{aligned}
&\rho_{r_J}\boldsymbol{\tau}_{t_J}\left((\boldsymbol{u},1)\mathbf{W}_J+v\boldsymbol{y}\right)^\top-\boldsymbol{y}_1\\
&=(\boldsymbol{u},1)\begin{pmatrix}\boldsymbol{\mu}+\boldsymbol{y}\mathbf{W}_y+\sum_{j<J}\boldsymbol{\mu}_{s_j}^{(J)}z_j(y_1+v)\\ +\sum_{j<J}z_j\left((\boldsymbol{u},1)\mathbf{W}_j+v\boldsymbol{y}+b_j v z_j(y_1+v)\mathbf{1}\right)\mathbf{W}_{t_j}\end{pmatrix}^\top
\end{aligned}$$

Looking at the terms in $v\boldsymbol{y}$ we now get $\rho_{r_J}\boldsymbol{\tau}_{t_J}=(1,0,\ldots,0)$. Let the first column vector of $\mathbf{W}_J$ be $\mathbf{w}_J^\top$ then we now have

$$(\boldsymbol{u},1)\mathbf{w}_J^\top=(\boldsymbol{u},1)\mathbf{w}^\top.$$

Writing

$$\begin{aligned}
\mathbf{w}'=\mathbf{w}_J-\mathbf{w}=\boldsymbol{\mu}'+\boldsymbol{y}\mathbf{W}'_y+\sum_{j<J}\boldsymbol{\mu}'_{s_j}z_j(y_1+v)\\
+\sum_{j<J}z_j\left((\boldsymbol{u},1)\mathbf{W}_j+v\boldsymbol{y}+b_j v z_j(y_1+v)\mathbf{1}\right)\mathbf{W}'_{t_j}
\end{aligned}$$

we now have

$$(\boldsymbol{u},1)\begin{pmatrix}\boldsymbol{\mu}'+\boldsymbol{y}\mathbf{W}'_y+\sum_{j<J}\boldsymbol{\mu}'_{s_j}z_j(y_1+v)\\ +\sum_{j<J}z_j\left((\boldsymbol{u},1)\mathbf{W}_j+v\boldsymbol{y}+b_j v z_j(y_1+v)\mathbf{1}\right)\mathbf{W}'_{t_j}\end{pmatrix}^\top=0.$$

The terms in $(\boldsymbol{u},1)\boldsymbol{\mu}'^\top$ tell us $\boldsymbol{\mu}'=\mathbf{0}$. Looking at terms involving $u_i y_h$ or y_h gives us $\mathbf{W}'_y=0$. Terms with z_j^2 tell us $b_j\mathbf{1}\mathbf{W}'_{t_j}=\mathbf{0}$ for all j. Terms in $(\boldsymbol{u},1)z_j v\mu'_{s_j}$ tell us $\mu'_{s_j}=0$ for all j. Finally, terms in $(\boldsymbol{u},1)(v\boldsymbol{y}\mathbf{W}'_{t_j})$ give us $\mathbf{W}'_{t_j}=0$.

We have now deduced that $\mathbf{w}'=\mathbf{0}$ and therefore $\mathbf{w}_J=\mathbf{w}$. This means the first column in $\mathbf{W}$ for which the adversary has produced a signature is a copy of the first column in the queried message $\mathbf{W}_J$. Using the same analysis on the last $k-1$ verification equations gives us that the other $k-1$ columns also match. This means a generic adversary can only produce valid signatures for previously queried messages, so we have EUF-CMA security.

Finally, let us consider the case where $b=1$, i.e., we are doing a strong signature verification. We saw earlier that $\rho_{r_J}\boldsymbol{\tau}_{t_J}b_J\mathbf{1}^\top=b_J=b\frac{1}{\rho_{r_J}}$ which can only be satisfied

if $b_J = 1$ and $\rho_{r_J} = 1$. This means $s = s_J$ and $r = r_J$ and $\mathbf{W} = \mathbf{W}_J$ and therefore $\boldsymbol{t} = \boldsymbol{t}_J$. So the generic adversary can only satisfy the strong verification equation with $b = 1$ by copying both the message and signature from a previous query with $b_J = 1$.

On the other hand, if $b = 0$, i.e., we are verifying a randomizable signature, we see from $\rho_{r_J} \boldsymbol{\tau}_{t_J} b_l \mathbf{1}^\top = b_J = b\frac{1}{\rho_{r_J}}$ that $b_J = 0$. So the adversary has randomized a signature intended for randomization. □

5.2. *Combined FSPS*

The structure-preserving signature scheme we just gave uses knowledge of the discrete logarithms of $\boldsymbol{U}$ in a fundamental way since $\tilde{\boldsymbol{T}}$ contains linear combinations of group elements in $\tilde{M}$, which yield a vector of group elements $\tilde{G}^{z(\boldsymbol{u},1)\mathbf{W}}$ that could not be computed without knowing $\boldsymbol{u}$. This situation is common for all structure-preserving signature schemes for messages that are vectors of group elements. The need to specify such discrete logarithms in the signing key therefore prevents them from being fully structure preserving.

To get full structure preservation, we circumvent this problem by only pairing message group elements with signature group elements where the signer does actually know the discrete logarithms. In our case, we will modify the structure-preserving signature scheme by letting the signer pick $\boldsymbol{U}$ herself and include it in the signature.

To make this idea work, we first make a minor modification to our signature scheme from before. We include a vector of $\ell - 1$ group elements $\tilde{\boldsymbol{X}}$ in the setup, and we modify $\tilde{S}$ to have the form $\tilde{S} = \left(\tilde{Y}_1 \tilde{\boldsymbol{X}}^{\boldsymbol{u}} \tilde{G}^v\right)^z$. The first verification equation then becomes

$$e(R, \tilde{S}) = e(G, \tilde{Y}_1) \prod_{i=1}^{\ell-1} e(U_i, \tilde{X}_i) e(V, \tilde{G})$$

If this was the only modification, we made it is not hard to see that the same security proof we gave earlier will work again, we are only modifying the verification equation by a random constant $\prod_{i=1}^{\ell-1} e(U_i, \tilde{X}_i)$. The surprising thing though is that the signature scheme remains secure if we let the signer pick the $\boldsymbol{U}$ part of the verification key herself and include it in the signature.

Letting the signer pick $\boldsymbol{U}$ as part of the verification key means that she can know their discrete logarithms. Since she also picks $z \leftarrow \mathbb{Z}_p^*$ herself, she can now use linear operations on the group elements in the message matrix to compute the group elements in the vector $\tilde{G}^{z(\boldsymbol{u},1)\mathbf{W}}$ part of $\tilde{\boldsymbol{T}}$. Furthermore, we have designed the scheme such that the rest can be computed with linear operations as well. To make randomizable signatures, the signer just needs to know $\tilde{G}^v$ and $\tilde{\boldsymbol{Y}}^v$ To make strong signatures she additionally needs to know $\tilde{\boldsymbol{X}}^v$ and $\tilde{G}^{v^2}$.

The resulting fully structure-preserving signature scheme is described below and can be used to sign messages consisting of $L = \ell k$ group elements in $\tilde{\mathbb{G}}$. It has a verification key size of 1 group element, a signature size of $\ell + k + 1$ group elements, and verification involves evaluating $k + 1$ pairing product equations. Since they are quite similar, we described the randomizable signature and the strongly unforgeable signature algorithms

at the same time. Setting $b = 0$ gives the algorithms for randomizable signatures and setting $b = 1$ gives the algorithms for strongly unforgeable signatures.

[**Efficient FSPS :** EFSP1]

$\mathsf{Setup}(1^\lambda, \ell, k)$: Run $(p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G}) \leftarrow \mathcal{G}(1^\lambda)$. Pick $\boldsymbol{x} \in \mathbb{Z}_p^{\ell-1}$, $\boldsymbol{y} \in \mathbb{Z}_p^k$, and compute $\tilde{\boldsymbol{X}} \leftarrow \tilde{\mathbb{G}}^{\ell-1}$, $\tilde{\boldsymbol{Y}} \leftarrow \tilde{\mathbb{G}}^k$. Return $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, e, G, \tilde{G}, \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}})$. Message space $\mathcal{M}$ is set to $\tilde{\mathbb{G}}^{\ell\times k}$ for prescribed ℓ and k.

$\mathsf{Key}(gk)$: Pick $v \leftarrow \mathbb{Z}_p$ calculate $V := G^v$. Output (vk, sk) where $vk = V$ and $sk = (\tilde{G}^v, \tilde{\boldsymbol{X}}^v, \tilde{\boldsymbol{Y}}^v, \tilde{G}^{v^2})$.

$\mathsf{Sign}_b(sk, \tilde{\boldsymbol{M}})$: Pick $\boldsymbol{u} \leftarrow \mathbb{Z}_p^{\ell-1}$, $z \leftarrow \mathbb{Z}_p^*$, and compute $r := \frac{1}{z}$, $\boldsymbol{U} := G^{\boldsymbol{u}}$, $R := G^r$, and $\tilde{S} := \left(\tilde{Y}_1 \tilde{\boldsymbol{X}}^{\boldsymbol{u}} \tilde{G}^v\right)^z$. For $j = 1, \ldots, k$,

$$\tilde{T}_j = \left(\prod_{i=1}^{\ell-1} \tilde{M}_{(i,j)}^{u_i} \cdot \tilde{M}_{(\ell,j)} \tilde{Y}_j^v \left(\tilde{Y}_1^v \prod_{i=1}^{\ell-1} (\tilde{X}_i^v)^{u_i} \tilde{G}^{v^2}\right)^{bz}\right)^z.$$

Set $\tilde{\boldsymbol{T}} := (\tilde{T}_1, \ldots, \tilde{T}_k)$. Output $\sigma := (\boldsymbol{U}, R, \tilde{S}, \tilde{\boldsymbol{T}})$.

$\mathsf{Vrf}_b(vk, \tilde{\boldsymbol{M}}, \sigma)$: Parse σ into $(\boldsymbol{U}, R, \tilde{S}, \tilde{\boldsymbol{T}})$. Return 1 if and only if $\tilde{M}_{(i,j)} \in \tilde{\mathbb{G}}^{\ell\times k}$, $R \in \mathbb{G}$, $\boldsymbol{U} \in \mathbb{G}^{\ell-1}$, $\tilde{S} \in \tilde{\mathbb{G}}$, $\tilde{\boldsymbol{T}} \in \tilde{\mathbb{G}}^k$, and

$$e(R, \tilde{S}) = e(G, \tilde{Y}_1) \prod_{i=1}^{\ell-1} e(U_i, \tilde{X}_i) e(V, \tilde{G}), \text{ and} \tag{20}$$

$$e(R, \tilde{T}_j) = \prod_{i=1}^{\ell-1} e(U_i, \tilde{M}_{(i,j)}) e(G, \tilde{M}_{(\ell,j)})\, e(V, \tilde{Y}_j) e(V, \tilde{S})^b \text{ for all } j = 1, \ldots, k. \tag{21}$$

$\mathsf{Rand}(vk, \tilde{\boldsymbol{M}}, \sigma)$: Parse σ into $(\boldsymbol{U}, R, \tilde{S}, \tilde{\boldsymbol{T}})$. Pick $\boldsymbol{\alpha} \leftarrow \mathbb{Z}_p^{\ell-1}$ and $\beta \leftarrow \mathbb{Z}_p^*$, calculate $\boldsymbol{U}' := \boldsymbol{U} R^{\boldsymbol{\alpha}}$, $R' := R^{\frac{1}{\beta}}$, $\tilde{S}' := \left(\tilde{S} \prod_{i=1}^{\ell-1} \tilde{X}_i^{\alpha_i}\right)^\beta$ and $\tilde{T}_j' := \left(\tilde{T}_j \prod_{i=1}^{\ell-1} M_{(i,j)}^{\alpha_i}\right)^\beta$. Return $\sigma' := (\boldsymbol{U}', R', \tilde{S}', \tilde{\boldsymbol{T}}')$.

Theorem 10. EFSP1 *gives a fully structure-preserving combined signature scheme that is* C-EUF-CMA *secure in the generic group model.*

Proof. Perfect correctness, perfect randomizability and structure preservation follow by inspection. The secret key $sk = (\mathcal{A}, \tilde{\boldsymbol{X}}^v, \tilde{\boldsymbol{Y}}^v, \tilde{G}^{v^2})$ consists of $\ell + k + 1$ group elements, and we can verify that it matches the verification key $vk = V$ by checking the pairing product equations

$$e(V,\tilde{G}) = e(G,\tilde{G}^{v}), \quad e(V,\tilde{\boldsymbol{X}}) = e(G,\tilde{\boldsymbol{X}}^{v}),$$
$$e(V,\tilde{\boldsymbol{Y}}) = e(G,\tilde{\boldsymbol{Y}}^{v}), \quad e(V,\tilde{G}^{v}) = e(G,\tilde{G}^{v^2})$$

so the signature scheme is fully structure preserving.

What remains now is to prove that the signature scheme is C-EUF-CMA secure in the generic group model. In the (Type-III) generic bilinear group model, the adversary may compute new group elements in either source group by taking arbitrary linear combinations of previously seen group elements in the same source group. We shall see that no such linear combination of group elements, viewed as formal Laurent polynomials in the variables picked by the key generator and the signing oracle, yields an existential forgery. It follows along the lines of the Uber assumption in [19] this that the signature scheme is C-EUF-CMA secure in the generic bilinear group model.

Let $\tilde{\boldsymbol{M}}_i = \tilde{G}^{\mathbf{W}_i} \in \tilde{\mathbb{G}}^{\ell\times k}$ for $\mathbf{W}_i \in \mathbb{Z}_p^{\ell\times k}$ be the i-th ($0 \le i \le q$) signing query made by the adversary. Since the adversary can use generic group operations to construct the message group elements, $\mathbf{W}_i$ may depend linearly on the discrete logarithms of public key elements in $\tilde{\mathbb{G}}$ and all previously seen signature elements in $\tilde{S}_j, \tilde{\boldsymbol{T}}_j$ for $j < i$. The adversary obtains signatures $(\boldsymbol{U}_i, R_i, \tilde{S}_i, \tilde{\boldsymbol{T}}_i)$ that

$$\boldsymbol{U}_i, \qquad R_i = G^{\frac{1}{z_i}}, \qquad \tilde{S}_i = \left(\tilde{Y}_1^{z_i}\sum_{\kappa=1}^{m-1}\tilde{X}_\kappa^{u_\kappa}\tilde{G}^{v}\right)^{z_i},$$
$$\tilde{\boldsymbol{T}}_i = \tilde{G}^{z_i((\boldsymbol{u}_i,1)\mathbf{W}_i + v\boldsymbol{y} + b_i z_i v(y_1 + \boldsymbol{u}_i\cdot\boldsymbol{x} + v))}$$

where $b_i = 0$ if query i is for a randomizable signature and $b_i = 1$ if query i is for a strong signature.

Viewed as Laurent polynomials we have that the discrete logarithms of a signature $(\boldsymbol{U}, R, \tilde{S}, \tilde{\boldsymbol{T}})$ generated by the adversary on $\tilde{\boldsymbol{M}} \in \tilde{\mathbb{G}}^{\ell\times k}$ are of the forms

$$\boldsymbol{u} = \boldsymbol{\alpha} + v\boldsymbol{\alpha}_v + \sum_i \boldsymbol{u}_i A_i + \sum_i \frac{1}{z_i}\boldsymbol{\alpha}_{r_i}$$
$$r = \rho + v\rho_v + \sum_i \boldsymbol{u}_i \boldsymbol{\rho}_{u_i}^{\top} + \sum_i \frac{1}{z_i}\rho_{r_i}$$
$$s = \sigma + \boldsymbol{\sigma}_x\boldsymbol{x}^{\top} + \boldsymbol{\sigma}_y\boldsymbol{y}^{\top} + \sum_j \sigma_{s_j} z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^{\top} + v)$$
$$+ \sum_j \boldsymbol{\sigma}_{t_j} z_j \left((\boldsymbol{u}_j,1)\mathbf{W}_j + v\boldsymbol{y} + b_j z_j v(y_1 + \boldsymbol{u}\boldsymbol{x}^{\top} + v)\mathbf{1}\right)$$
$$\boldsymbol{t} = \boldsymbol{\tau} + \boldsymbol{x}T_x + \boldsymbol{y}T_y + \sum_j z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^{\top} + v)\boldsymbol{\tau}_{s_j}$$
$$+ \sum_j z_j \left((\boldsymbol{u}_j,1)\mathbf{W}_j + v\boldsymbol{y} + b_j z_j v(y_1 + \boldsymbol{u}\boldsymbol{x}^{\top} + v)\mathbf{1}\right) T_{t_j}$$

Similarly, all ℓk discrete logarithms $\mathbf{W}$ of $\tilde{M}$ can be written in a form similar to s, and all discrete logarithms of queried message matrices $\mathbf{W}_i$ can be written in a form similar to s where the sums are bounded by $j < i$.

For the first verification equation to be satisfied, we must have $rs = y_1 + \boldsymbol{u}\boldsymbol{x}^\top + v$, i.e.,

$$\begin{pmatrix} \rho + \sum_i \boldsymbol{u}_i \boldsymbol{\rho}_{u_i}^\top \\ +v\rho_v + \sum_i \frac{1}{z_i}\rho_{r_i} \end{pmatrix} \cdot \begin{pmatrix} \sigma + \boldsymbol{\sigma}_x \boldsymbol{x}^\top + \boldsymbol{\sigma}_y \boldsymbol{y}^\top + \sum_j \sigma_{s_j} z_j (y_1 + \boldsymbol{u}_j \boldsymbol{x}^\top + v) \\ + \sum_j \boldsymbol{\sigma}_{t_j} z_j \big((\boldsymbol{u}_j, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + \boldsymbol{u}_j \boldsymbol{x}^\top + v)\mathbf{1}\big)^\top \end{pmatrix}$$
$$= y_1 + \left(\boldsymbol{\alpha} + v\boldsymbol{\alpha}_v + \sum_i \boldsymbol{u}_i A_i + \sum_i \frac{1}{z_i}\boldsymbol{\alpha}_{r_i}\right)\boldsymbol{x}^\top + v$$

We start by noting that $r \neq 0$ since otherwise rs cannot have the term y_1. Please observe that it is only in $\mathbb{G}$ that we have terms including indeterminates with negative power, i.e., $\frac{1}{z_i}$. In $\tilde{\mathbb{G}}$, all indeterminates have positive power, i.e., so $s_j, \boldsymbol{t}_j, \mathbf{W}_j$ only contain proper multivariate polynomials. Now suppose for a moment that $\rho_{r_i} = 0$ for all i. Then in order not to have a terms involving z_j's in rs we must have

$$\sum_j \sigma_{s_j} z_j (y_1 + \boldsymbol{u}_j \boldsymbol{x}^\top + v) + \sum_j \boldsymbol{\sigma}_{t_j} z_j \left((\boldsymbol{u}_j, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + \boldsymbol{u}_j \boldsymbol{x}^\top + v)\mathbf{1}\right)^\top = 0.$$

The term y_1 now gives us $\rho\sigma_{y,1} = 1$ and the term v gives us $\rho_v\sigma = 1$. This means $\rho \neq 0$ and $\sigma \neq 0$ and therefore we reach a contradiction since the constant term should be $\rho\sigma = 0$. We conclude that there must exist some J for which $\rho_{r_J} \neq 0$.

Now we have the term $\rho_{r_J}\sigma\frac{1}{z_J} = 0$, which shows us $\sigma = 0$. The terms $\rho_{r_J}\sigma_{y,h}\frac{y_h}{z_J} = 0$ for $h = 1, \ldots, k$ give us $\boldsymbol{\sigma}_y = \mathbf{0}$.

The polynomials corresponding to s_j and $\boldsymbol{t}_j$ contain the indeterminate z_j in all terms, so no linear combination of them can give us a term where the indeterminate component is vy_h for some $h \in \{1, \ldots, k\}$. Since $\mathbf{W}_j$ is constructed as a linear combination of elements in the verification key and components in $\tilde{\mathbb{G}}$ from previously seen signatures, it too cannot contain a term where the indeterminate component is vy_h. The coefficient of $\frac{z_j}{z_J}vy_h$ is therefore $\rho_{r_J}\sigma_{t_j,h} = 0$ and therefore $\sigma_{t_j,h} = 0$ for every $j \neq J$ and $h \in \{1, \ldots, k\}$. This shows $\boldsymbol{\sigma}_{t_j} = \mathbf{0}$ for all $j \neq J$. Looking at the coefficients for vy_h for $h = 1, \ldots, k$ we see that $\boldsymbol{\sigma}_{t_J} = \mathbf{0}$ too.

The terms $\rho_{r_J}\sigma_{s_j}\frac{z_j}{z_l}v$ give us $\sigma_{s_j} = 0$ for all $j \neq J$. In order to get a coefficient of 1 for the term y_1 we see that $\sigma_{s_J} = \frac{1}{\rho_{r_J}}$, which is nonzero. Our analysis has now shown that

$$s = \boldsymbol{\sigma}_x \boldsymbol{x}^\top + \frac{1}{\rho_{r_J}} z_J (y_1 + \boldsymbol{u}_J \boldsymbol{x}^\top + v).$$

Let us now analyze the structure of r. The term $\rho_v\sigma_J v^2 z_J = 0$ gives us $\rho_v = 0$. We know from our previous analysis that if there was a second $i \neq J$ for which $\rho_{r_i} \neq 0$

then also $\sigma_{\rho_J} = 0$, which it is not. Therefore, for all $i \neq J$ we have $\rho_{r_i} = 0$. The term $\rho\sigma_{s_J} z_J y_1$ gives $\rho = 0$. The terms in $\boldsymbol{\rho}_{u_i}\sigma_{s_J}\boldsymbol{u}_i z_J v$ give us $\boldsymbol{\rho}_{u_i} = \mathbf{0}$ for all i. Our analysis therefore shows

$$r = \rho_{r_J}\frac{1}{z_J}.$$

Finally, having simplified r and s analyzing the terms in $\boldsymbol{u}$ gives us

$$\boldsymbol{u} = \boldsymbol{u}_J + \rho_{r_J}\boldsymbol{\sigma}_x\frac{1}{z_J}.$$

We now turn to the second verification equation, which is $rt_1 = (\boldsymbol{u}, 1)\mathbf{w}^\top + vy_1 + bvs$, where $\mathbf{w}^\top$ is the first column vector of $\mathbf{W}$. The message vector is of the form

$$\mathbf{w} = \begin{array}{l}\boldsymbol{\mu} + \boldsymbol{x}\mathbf{W}_x + \boldsymbol{y}\mathbf{W}_y + \sum_j \boldsymbol{\mu}_{s_j} z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v) \\ + \sum_j z_j \left((\boldsymbol{u}_j, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v)\mathbf{1}\right)\mathbf{W}_{t_j}\end{array},$$

where $\boldsymbol{\mu}, \mathbf{W}_x, \mathbf{W}_y\boldsymbol{\mu}_{s_j}$ and $\mathbf{W}_{t_j}$ are vectors and matrices of corresponding size with entries in $\mathbb{Z}_p$ chosen by the adversary. Similarly, we can write out

$$\begin{aligned} t_1 = \tau + \boldsymbol{\tau}_x\boldsymbol{x}^\top + \boldsymbol{\tau}_y\boldsymbol{y}^\top + \sum_j \tau_{s_j} z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v) + \sum_j \boldsymbol{\tau}_{t_j} z_j \\ \left((\boldsymbol{u}, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v)\mathbf{1}\right) \end{aligned}$$

for elements and vectors of corresponding size $\tau, \boldsymbol{\tau}_x, \boldsymbol{\tau}_y, \tau_{s_j}, \boldsymbol{\tau}_{t_j}$ with entries in $\mathbb{Z}_p$ chosen by the adversary.

Writing out the second verification equation, we have

$$\begin{aligned} &\rho_{r_J}\frac{1}{z_J}\begin{pmatrix} \tau + \boldsymbol{\tau}_x\boldsymbol{x}^\top + \boldsymbol{\tau}_y\boldsymbol{y}^\top + \sum_j \tau_{s_j} z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v) \\ + \sum_j \boldsymbol{\tau}_{t_j} z_j \left((\boldsymbol{u}_j, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v)\mathbf{1}\right)^\top \end{pmatrix} \\ &= vy_1 + bv\left(\boldsymbol{\sigma}_x\boldsymbol{x}^\top + \frac{1}{\rho_{r_J}} z_J (y_1 + \boldsymbol{u}_J\boldsymbol{x}^\top + v)\right) \\ &\quad + \left(\boldsymbol{u}_J + \rho_{r_J}\boldsymbol{\sigma}_x\frac{1}{z_J}, 1\right) \\ &\quad \times \begin{pmatrix} \boldsymbol{\mu} + \boldsymbol{x}\mathbf{W}_x + \boldsymbol{y}\mathbf{W}_y + \sum_j \boldsymbol{\mu}_{s_j} z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v) \\ + \sum_j z_j \left((\boldsymbol{u}_j, 1)\mathbf{W}_j + v\boldsymbol{y} + b_j v z_j (y_1 + \boldsymbol{u}_j\boldsymbol{x}^\top + v)\mathbf{1}\right)\mathbf{W}_{t_j} \end{pmatrix}^\top. \end{aligned}$$

Looking at the coefficients of terms involving $\frac{1}{z_J}$, we get the following equalities for all $j \neq J$: $\tau = \boldsymbol{\sigma}_x\mu^\top$ $(\frac{1}{z_J})$, $\boldsymbol{\tau}_x = \boldsymbol{\sigma}_x\mathbf{W}_x^\top$ $(\frac{x_h}{z_J})$, $\boldsymbol{\tau}_y = \boldsymbol{\sigma}_x\mathbf{W}_y^\top$ $(\frac{y_h}{z_J})$, $\tau_{s_j} = \boldsymbol{\sigma}_x\boldsymbol{\mu}_{s_j}^\top$ $(\frac{vz_j}{z_J})$, $\boldsymbol{\tau}_{t_j} = \boldsymbol{\sigma}_x T_{t_j}^\top$ $(\frac{vy_k z_j}{z_J})$. Canceling out these terms, we are left with

$$
\begin{aligned}
&\rho_{r_J}\left(\tau_{s_J}(y_1+\boldsymbol{u}_J\boldsymbol{x}^\top+v)+\boldsymbol{\tau}_{t_J}\left((\boldsymbol{u}_J,1)\mathbf{W}_J+v\boldsymbol{y}+b_Jvz_J(y_1+\boldsymbol{u}_J\boldsymbol{x}^\top+v)\mathbf{1}\right)^\top\right)\\
&\quad=vy_1+bv\left(\boldsymbol{\sigma}_x\boldsymbol{x}^\top+\frac{1}{\rho_{r_J}}z_J(y_1+\boldsymbol{u}_J\boldsymbol{x}^\top+v)\right)\\
&\qquad+\rho_{r_J}\boldsymbol{\sigma}_x\left(\boldsymbol{\mu}_{s_J}(y_1+\boldsymbol{u}_J\boldsymbol{x}^\top+v)\right.\\
&\qquad\left.+\left((\boldsymbol{u}_J,1)\mathbf{W}_J+v\boldsymbol{y}+b_Jvz_J(y_1+\boldsymbol{u}_J\boldsymbol{x}^\top+v)\mathbf{1}\right)\mathbf{W}_{t_J}\right)^\top\\
&\qquad+(\boldsymbol{u}_J,1)\begin{pmatrix}\boldsymbol{\mu}+\boldsymbol{x}\mathbf{W}_x+\boldsymbol{y}\mathbf{W}_y+\sum_j\boldsymbol{\mu}_{s_j}z_j(y_1+\boldsymbol{u}_j\boldsymbol{x}^\top+v)\\+\sum_j z_j\left((\boldsymbol{u}_j,1)\mathbf{W}_j+v\boldsymbol{y}+b_jvz_j(y_1+\boldsymbol{u}_j\boldsymbol{x}^\top+v)\mathbf{1}\right)\mathbf{W}_{t_j}\end{pmatrix}^\top.
\end{aligned}
$$

Terms involving z_j and z_j^2 must cancel out, so we can assume $\mu_{s_j}=\mathbf{0}$ and $\mathbf{W}_{t_j}=0$ for $j>J$. Since $\mathbf{W}_J$ does not involve z_J in any of its terms, we get from the terms in $(\boldsymbol{u}_J,1)z_Jv\mu_{s_J}^\top$ that $\boldsymbol{\mu}_{s_J}=0$. Since there can be no terms involving z_J^2 we get $b_J\mathbf{1}\mathbf{W}_{t_J}^\top=\mathbf{0}$. Looking at the coefficients for v we get $\tau_{s_J}=\boldsymbol{\sigma}_x\boldsymbol{\mu}_{s_J}$. This leaves us with

$$
\begin{aligned}
&\rho_{r_J}\boldsymbol{\tau}_{t_J}\left((\boldsymbol{u}_J,1)\mathbf{W}_J+v\boldsymbol{y}+b_Jvz_J(y_1+\boldsymbol{u}_J\boldsymbol{x}^\top+v)\mathbf{1}\right)^\top\\
&\quad=vy_1+bv\left(\boldsymbol{\sigma}_x\boldsymbol{x}^\top+\frac{1}{\rho_{r_J}}z_J(y_1+\boldsymbol{u}_J\boldsymbol{x}^\top+v)\right)\\
&\qquad+\rho_{r_J}\boldsymbol{\sigma}_x\left(((\boldsymbol{u}_J,1)\mathbf{W}_J+v\boldsymbol{y})\,\mathbf{W}_{t_J}\right)^\top\\
&\qquad+(\boldsymbol{u}_J,1)\begin{pmatrix}\boldsymbol{\mu}+\boldsymbol{x}\mathbf{W}_x+\boldsymbol{y}\mathbf{W}_y+\sum_{j<J}\boldsymbol{\mu}_{s_j}z_j(y_1+\boldsymbol{u}_j\boldsymbol{x}^\top+v)\\+\sum_{j<J}z_j\left((\boldsymbol{u}_j,1)\mathbf{W}_j+v\boldsymbol{y}+b_jvz_j(y_1+\boldsymbol{u}_j\boldsymbol{x}^\top+v)\mathbf{1}\right)\mathbf{W}_{t_j}\end{pmatrix}^\top\\
&\qquad+(\boldsymbol{u}_J,1)z_J\left((\boldsymbol{u}_J,1)\mathbf{W}_J+v\boldsymbol{y})\mathbf{W}_{t_J}\right)^\top.
\end{aligned}
$$

Looking at the terms involving z_Jv^2 we see $\rho_{r_J}\boldsymbol{\tau}_{t_J}b_J\mathbf{1}^\top=b\frac{1}{\rho_{r_J}}$. The only remaining terms involving z_J now give us $\mathbf{W}_{t_J}=0$. This gives us

$$
\begin{aligned}
&\rho_{r_J}\boldsymbol{\tau}_{t_J}\left((\boldsymbol{u}_J,1)\mathbf{W}_J+v\boldsymbol{y}\right)^\top\\
&\quad=vy_1+bv\boldsymbol{\sigma}_x\boldsymbol{x}^\top\\
&\qquad+(\boldsymbol{u}_J,1)\begin{pmatrix}\boldsymbol{\mu}+\boldsymbol{x}\mathbf{W}_x+\boldsymbol{y}\mathbf{W}_y+\sum_{j<J}\boldsymbol{\mu}_{s_j}z_j(y_1+\boldsymbol{u}_j\boldsymbol{x}^\top+v)\\+\sum_{j<J}z_j\left((\boldsymbol{u}_j,1)\mathbf{W}_j+v\boldsymbol{y}+b_jvz_j(y_1+\boldsymbol{u}_j\boldsymbol{x}^\top+v)\mathbf{1}\right)\mathbf{W}_{t_j}\end{pmatrix}^\top
\end{aligned}
$$

Looking at the terms in $v\boldsymbol{y}$ we now get $\rho_{r_J}\boldsymbol{\tau}_{t_J}=(1,0,\ldots,0)$. This means $(\boldsymbol{u}_J,1)\mathbf{W}_J^\top=b\boldsymbol{\sigma}_x\boldsymbol{x}^\top+(\boldsymbol{u}_J,1)\mathbf{W}^\top$, where $\mathbf{W}_J^\top$ is the first column of $\mathbf{W}_J$. Looking at the coefficients of vx_h, we see that if $b\boldsymbol{\sigma}_x=\mathbf{0}$. Since $\mathbf{W}_J$ and $\mathbf{W}$ are independent of $\boldsymbol{u}_J$ this means $\mathbf{W}=\mathbf{W}_J$.

A similar argument can applied to the remaining $k-1$ verification equations showing us that in all columns $\mathbf{W}$ and ϖ_J match. This means $\tilde{\boldsymbol{M}}=\tilde{\boldsymbol{M}}_J$, so the signature scheme is existentially unforgeable both for randomizable signatures and strong signatures.

Finally, let us consider the case where $b=1$, i.e., we are doing a strong signature verification. We have already seen that $b\boldsymbol{\sigma}_x=\mathbf{0}$ so when $b=1$ this means $\boldsymbol{\sigma}_x=\mathbf{0}$.

Table 4. Size of objects and number of verification equations in fully structure-preserving signature schemes for messages consisting of $L = \ell k$ elements in $\tilde{\mathbb{G}}$.

Scheme	$\lvert sk\rvert$	$\lvert gk\rvert$	$\lvert vk\rvert$	$\lvert\sigma\rvert$	PPE
FSP2	$(4, 0)$	$(4, 4)$	$(1, 10 + 3\ell + 3k)$	$(7 + \ell + k, 4 + 2k)$	$5 + k$
EFSP1	$(0, k + \ell + 1)$	$(1, k + \ell)$	$(1, 0)$	$(\ell, 1 + k)$	$1 + k$

Since $\rho_{r_J} \boldsymbol{\tau}_{t_J} b_J \mathbf{1}^\top = b_J = b\frac{1}{\rho_{r_J}}$ we see that $b_J = 1$ and $\rho_{r_J} = 1$. This means $s = s_J$ and $r = r_J$ and $\boldsymbol{u} = \boldsymbol{u}_J$ and $\mathbf{W} = \mathbf{W}_J$, it means $\tilde{\boldsymbol{M}} = \tilde{\boldsymbol{M}}_J$, and therefore $\boldsymbol{t} = \boldsymbol{t}_J$. So the generic adversary can only satisfy the strong verification equation with $b = 1$ by copying both the message and signature from a previous query with $b_J = 1$.

On the other hand, if we have $b = 0$, i.e., we are verifying a randomizable signature, we see from $\rho_{r_J} \boldsymbol{\tau}_{t_J} b_l \mathbf{1}^\top = b_J = b\frac{1}{\rho_{r_J}}$ that $b_J = 0$. So the adversary has randomized a signature intended for randomization. □

5.3. *Efficiency*

We give a detailed performance comparison in Table 4 between FSP2 based on standard assumptions and our most efficient scheme EFSP1. Unsurprisingly we get significantly smaller signature size and a modest reduction in the number of verification equations. We also observe the verification key in EFSP1 is just a single group element, which is optimal and makes it cheap to certify the verification key in digital credential systems or by a certification authority.

6. Lower Bound on Signature Size and Verification Key Size

The signatures of our concrete FSPSs consist of $\Omega(\sqrt{L})$ group elements when signing L-element messages. This may seem disappointing given previous constant-size constructions of SPS, but we argue that the $\sqrt{L}$ factor is unavoidable. It is a consequence of the following new trade-off between signature and verification key size for arbitrary (even one-time) SPS schemes.

Theorem 11. *Consider a (one-time) SPS scheme on messages in $\tilde{\mathbb{G}}^L$ in the asymmetric (Type-III) bilinear group setting. Let κ be the number of group elements in vk (and gk) used in evaluating the PPEs in verification. Let σ the number of group elements in signatures. If the scheme is existentially unforgeable in a model in which the adversary has access to a valid signature on a known message and the scheme has a generic signing algorithm, we have $\kappa + \sigma \geq \sqrt{L}$.*

Proof. Denote by $(M_1, \ldots, M_L) \in \tilde{\mathbb{G}}^L$ the message vector, by $(U_1, \ldots, U_{\kappa_1}, V_1, \ldots, V_{\kappa_2}) \in \mathbb{G}^{\kappa_1} \times \tilde{\mathbb{G}}^{\kappa_2}$ $(\kappa_1 + \kappa_2 = \kappa)$ the verification key elements, and by $(R_1, \ldots, R_{\sigma_1}, S_1, \ldots, S_{\sigma_2}) \in \mathbb{G}^{\sigma_1} \times \tilde{\mathbb{G}}^{\sigma_2}$ $(\sigma_1 + \sigma_2 = \sigma)$ the signature elements. The corresponding discrete logarithms are written in lowercase letters.

Each verification equation of the scheme can be expressed as a bilinear relation between the discrete logarithms of the group elements in $\mathbb{G}$ (namely the U_i's and R_i's) on the one hand, and those of the elements in $\tilde{\mathbb{G}}$ (namely the M_i's, V_i's and S_i's) on the other. The i-th pairing product equation can thus be written in matrix form as:

$$X^T E_i Y = 0, \tag{22}$$

where X and Y are the column vectors given by

$$X = (r_1, \ldots, r_{\sigma_1}, u_1, \ldots, u_{\kappa_1}, 1)^T, \text{ and}$$
$$Y = (m_1, \ldots, m_L, s_1, \ldots, s_{\sigma_2}, v_1, \ldots, v_{\kappa_2}, 1)^T,$$

and E_i is a public $(\kappa_1 + \sigma_1 + 1) \times (L + \kappa_2 + \sigma_2 + 1)$ matrix over $\mathbb{Z}_p$.

Now fix a valid message-signature pair $(M_1, \ldots, M_L, R_1, \ldots, R_{\sigma_1}, S_1, \ldots, S_{\sigma_2})$. By linear algebra, we can efficiently compute a nonzero tuple $(m_1^*, \ldots, m_L^*) \in \mathbb{Z}_p^L$ that satisfies

$$E_i(m_1^*, \ldots, m_L^*, 0, \ldots, 0)^T = 0$$

for all i, if it exists. Then, it is clear from Eq. (22) that $(R_1, \ldots, R_{\sigma_1}, S_1, \ldots, S_{\sigma_2})$ is still a valid signature on the distinct message vector $(M_1 \tilde{G}^{m_1^*}, \ldots, M_L \tilde{G}^{m_L^*})$, which contradicts existential unforgeability.

Therefore, with n being the number of verification equations, the linear map $\mathbb{Z}_p^L \to \mathbb{Z}_p^{n(\kappa_1+\sigma_1+1)}$ mapping $(m_1, \ldots, m_L)$ to the concatenation of all vectors $E_i(m_1, \ldots, m_L, 0, \ldots, 0)^T$ must be injective. In particular, we have:

$$L \leq n \cdot (\kappa_1 + \sigma_1 + 1) \leq n \cdot (\kappa + \sigma),$$

where the second inequality comes from the fact that we must have $\sigma_2 \geq 1$; otherwise, the generic signing algorithm would output signatures that cannot depend on the message.

Finally, we must have $n \leq \sigma$ (after removing possibly redundant verification equations). Indeed, if it were not the case, the quadratic system satisfied by the discrete logarithms of the signature elements would be overdetermined, and a generic message would not admit any valid signature at all. We thus obtain $L \leq \sigma \cdot (\kappa + \sigma) \leq (\kappa + \sigma)^2$, which concludes the proof. □

The following theorem can be proven in a similar manner as above by replacing public keys, secret keys, and signatures with commitment keys, opening information, and commitments, respectively.

Theorem 12. *Consider a structure-preserving commitment scheme on messages in $\tilde{\mathbb{G}}^L$ in the asymmetric (Type-III) bilinear group setting. Assume that the commitment key consists of elements in $\tilde{\mathbb{G}}$, and let χ be the number of elements in commitments and o the number of group elements in the opening information. If the scheme is target collision resistant and has a generic commitment algorithm, we have $\chi + o \geq \sqrt{L}$.*

From Theorem 11, we immediately see that an FSPS scheme obtained from construction FSP1 must have signatures of more than $\sqrt{L}$ elements. This is because all signatures include as a subset including both the verification key and signature of a structure-preserving OTS scheme signing L-element messages.

Regarding EFSP1 in Sect. 5.2, only a constant number of group elements from vk and gk are involved in the verification. With such optimized verification keys, signatures have to have more than $\sqrt{L}$ elements according to Theorem 11.

Finally, Theorem 12 shows that an FSPS scheme obtained from construction FSP2 must also have signatures of more than $\sqrt{L}$ elements, at least when the underlying trapdoor commitment scheme has its key elements on the same side as the resulting signature, which seems necessary with our approach based on MTC.

7. Applications

We first discuss potential applications of FSPS in this section. Later, we show composable and modular anonymous credentials from [21] as a concrete example.

Public key infrastructure. On the very applied side, the question is connected with the timely problem of public key infrastructures. Few protocols have been designed with the goal of being secure against adversarial keys, and few real-world certificate authorities validate that registries provide valid public keys or prove knowledge of the corresponding secret keys. The availability of schemes with efficient non-interactive proofs-of-knowledge of secret key possession can only improve this situation. In the provable security literature, this *knowledge of secret key* solution to *rogue key attacks* appeared early on in the study of multi-signatures by Micali et al. [44, Problem 4 and Fix 4]. See Ristenpart and Yilek [46] for a comprehensive study of this problem.

Protocol design in strong security model. More generally, these obstacles to secret key extraction have hindered modular composable protocol design. Camenisch et al. [23] developed a framework for practical universally composable (UC) zero-knowledge proofs, in which they identify proofs-of-knowledge of exponents as a major bottleneck. Camenisch et al. [21] constructed unlinkable redactable signatures and anonymous credentials that are UC-secure. Their construction requires proofs-of-knowledge of the signing key of a structure-preserving signature scheme, which in turn, as studied by Chase et al. [26], is an instance of a general transformation for making signature schemes simulatable [11]. Given these examples, we conjecture that fully structure-preserving signature schemes help build UC-secure privacy preserving protocols.

Strengthening privacy in group and ring signatures. In classical group and ring signatures, e.g., [15,17,36,47], the goal of the adversary against privacy is to distinguish signatures from two *honest* members whose keys are actually generated and registered by the challenger. The attack game aborts if either of the targets is a corrupted member registered with an adversarially generated key. Instead of excluding such corrupt members from the scope of security, stronger privacy in the presence of adversarial keys can be guaranteed, if the challenger can extract the secret key to

create group or ring signatures on their behalf. Such a model is meaningful when some keys are generated incorrectly, e.g., because of multiple potentially flawed implementations, but their owners nevertheless use them with the correct signing algorithm. Note that this requires a trusted common reference string that puts mild assumptions on the trust model to retain other security properties such as unforgeability and non-frameability: the extraction trapdoor must be inaccessible for the adversary.

As a concrete example, we overview a UC-secure anonymous credential system in [21].

Anonymous Credentials Like a traditional digital certificate, an anonymous credential can be seen as a signature by an issuer on the attributes of users. To preserve privacy, the signature scheme used to certify information must be redactable. This allows users to only reveal the information that they deem adequate for a given service provider and context. To preserve anonymity, the signature scheme must be unlinkable. This prevents service providers from collecting meta-data about the behavior of users and also prevents the collation of attribute information about the same user previously revealed in different contexts or across multiple services.

Thus, in addition to the Key, Sign, and Vrf algorithms of traditional signatures, unlinkable redactable signatures provide a Derive algorithm that given a signature on a message produces an unlinkable signature on a redacted message. Anonymous credentials guarantee unlinkability even when issuers and service providers collude. To model this, we require that signatures produced by Derive are indistinguishable from fresh signatures, as long as the verification key satisfies the predicate CheckVK. For such keys, we will require that signing keys are online extractable and this is exactly where FSPS are needed.

A UC Functionality for Anonymous Credentials The following functionality models the unforgeability, unlinkability, and redactability requirements of anonymous credentials.[2]

Intuitively, for the scheme to be composably secure, the ideal functionality together with the simulator must be indistinguishable from the real functionality that on messages keygen, check_vk, issue, show, and verify simply runs the cryptographic algorithms Key, CheckVK, Sign, Derive, and Vrf of an unlinkable redactable signature scheme. Instead of running CheckVK, the simulator provides the functionality with signing keys extracted from well-formed issuer keys.

[2] Note that [21] further extend this functionality with pseudonyms. We ignore additional features and focus on the main obstacles to UC security.

Functionality $\mathcal{F}$

- On input (keygen, sid) from the issuer, ask the simulator for system parameters, and polynomial time algorithms Key, Sign, Derive, and Vrf. Generate an issuing key pair using Key. Store the parameters and keys and return the verification key to the issuer.
- On input (check_vk, sid, vk') from some party, if vk' has not been stored before, ask the simulator for a signing key and store both keys. If vk' was stored or the simulator provides a signing key return true, otherwise return false.
- On input (leak_sk, sid) from the issuer, return the signing key recorded during (keygen, sid) and set a leak flag.
- On input (issue, sid, $\boldsymbol{m}$) from a credential user, generate a signature on $\boldsymbol{m}$ using Sign, store the message in $\mathcal{Q}$ and return the signature.
- On input (show, sid, vk', I, $\boldsymbol{m}$, σ) from a credential user, check whether Derive(vk', I, $\boldsymbol{m}$, σ) succeeds and whether a signing key sk' corresponding to vk' was stored. In this case return Derive(pk', I, Zero($\boldsymbol{m}$, I), Sign(sk', Zero($\boldsymbol{m}$, I))), otherwise return the output of Derive(vk', I, $\boldsymbol{m}$, σ).
- On input (verify, sid, vk', σ, $\boldsymbol{m}_I$) from a credential verifier, compute $result \leftarrow$ Vrf(vk', $\boldsymbol{m}_I$, σ). If vk' is the public key recorded during (keygen, sid), the leak flag was not set, and $\nexists \boldsymbol{m} \in \mathcal{Q}$ such that $\boldsymbol{m}_I = I(\boldsymbol{m})$, then output 0. Otherwise output $result$.

The functionality enforces security properties regardless of the algorithms provided by the simulator:

Unforgeability is guaranteed by rejecting signatures for the honestly generated verification key that cannot be derived from signed messages, unless the signing key has been leaked.

Unlinkability and Redactability are guaranteed by generating a fresh redacted signature that covers the part of the message that is not redacted, while the rest of the initial message is set to zero. The presence of the (leak_sk, sid) message models that privacy guarantees are ensured even when the adversary learns the issuers signing key. This corresponds to the *full anonymity* property of group signatures [15]. The check_vk message models that privacy guarantees are ensured even for adversarial verification keys as long as verification keys are well formed.

Realizing Compact Unlinkable Redactable Signatures Traditional certificates sign a hash of their attributes and can thus be very compact. Using hashed data structures such as Merkle trees, it is not too hard to extend the approach to support redactability. It is, however, challenging to simultaneously achieve compactness, unforgeability, unlinkability, and redactability. To our knowledge, all existing approaches [21,33] employ structure-preserving signatures in one way or another. [33] uses a SPS to sign a *set commitment* while [21] employ a *vector commitment* to compresses sets, respectively, vectors, of attributes into a single group element to allow compact openings. Both set and vector commitments can be seen as commitments with an efficient partial opening algorithm.

An unlinkable redactable signature (URS) scheme consists of five algorithms. We recall the construction of [21] of URS from SPS and generalize it for commitments with partial opening.

[**URS:** URS]

Setup(1^λ): Take a security parameter as input. Output system parameters gk.
The construction of [21] generates a bilinear group, and the parameters for a set/vector commitment scheme, as well as a Groth-Sahai common reference string.

Key(gk): Take system parameters gk as input. Output public verification and private signing keys (vk, sk). The verification key defines the message space $\mathcal{M}$.
The construction generates an SPS key pair and extends the verification key with a Groth-Sahai proof of knowledge of the signing key.

CheckVK(vk): Take a verification key vk as input and check that it is correctly formed. Outputs 1 if vk is correct, and 0 otherwise.
The construction verifies the proof of knowledge of the signing key.

Sign($sk, \boldsymbol{m}$): Take a signing key sk and an initial message $\boldsymbol{m} \in \mathcal{M}$ as input. Produce an initial signature σ.
The construction signs a commitment to $\boldsymbol{m}$.

Derive($vk, I, \boldsymbol{m}, \sigma$): Take the verification key vk, a reduction specification I, and an initial message and signature $\boldsymbol{m}, \sigma$ as input. Produce a redacted message and signature $\sigma_I, \boldsymbol{m}_I$.
The construction of [21] partially opens the commitment to $\boldsymbol{m}$ contained in σ to $\boldsymbol{m}_I$ and proves knowledge of the SPS signature and opening.

Vrf($vk, \boldsymbol{m}, \sigma$): Take a verification key vk and a message and signature as input. Check the signature. The message and signature can be either initial or redacted.
In the construction, verification for initial messages corresponds to verifying the SPS signature, while verification of redacted signatures corresponds to verifying the Groth-Sahai proof-of-knowledge of the SPS signature and opening.

The composable unlinkability of the ideal functionality can only be realized when the unlinkable redactable signature scheme allows for the online extraction of signing keys. Informally, key extractability requires additional predicates CheckVK, CheckKeys, and trapdoor parameter generation and extraction algorithms SetupTd, ExtractKey. When SetupTd is run and CheckVK outputs 1, then ExtractKey(vk, td) must output a valid signing key sk, that is, CheckKeys(vk, sk) $= 1$. The efficient construction of such an extraction algorithm is facilitated by FSPS as we can efficiently prove knowledge of the signing key using the Groth–Sahai proof system. Note that the scheme only signs a vector commitment that compresses a large number of messages into a single group element. Consequently, the overhead of full structure preservation is small: When using FSP2 from Sect. 4.2 as the FSPS, signatures for signing a single group element consist of only 15 elements per signature and proofs of key possession consist of just 18 elements, respectively. Consequently, the initial signatures of the URS consist of only 16 elements and redacted signatures of 38 elements.

References

[1] M. Abe, J. Camenisch, R. Dowsley, M. Dubovitskaya, On the impossibility of structure-preserving deterministic primitives, in *Proceedings of Theory of Cryptography—11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24–26, 2014* (2014), pp. 713–738

[2] M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, M. Ohkubo, Constant-size structure-preserving signatures: Generic constructions and simple assumptions. *J. Cryptology* **29**(4), 833–878 (2016)

[3] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. Ohkubo, Structure-preserving signatures and commitments to group elements. *J. Cryptology* **29**(2), 363–421 (2016)

[4] M. Abe, J. Groth, K. Haralambiev, M. Ohkubo. Optimal structure-preserving signatures in asymmetric bilinear groups, in *Advances in Cryptology—CRYPTO 2011, volume 6841 of LNCS* (Springer, 2011), pp. 649–666

[5] M. Abe, J. Groth, M. Ohkubo, Separating short structure-preserving signatures from non-interactive assumptions, in *Advances in Cryptology—ASIACRYPT 2011, volume 7073 of LNCS* (Springer, 2011), pp. 628–646

[6] M. Abe, J. Groth, M. Ohkubo, M. Tibouchi, Structure-preserving signatures from type II pairings, in J. A. Garay, R. Gennaro, editors, *Advances in Cryptology—CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17–21, 2014, Proceedings, Part I, volume 8616 of Lecture Notes in Computer Science* (Springer, 2014), pp. 390–407

[7] M. Abe, J. Groth, M. Ohkubo, M. Tibouchi, Unified, minimal and selectively randomizable structure-preserving signatures, in *Theory of Cryptography—11th Theory of Cryptography Conference, volume 8349 of LNCS* (Springer, 2014), pp. 688–712

[8] M. Abe, K. Haralambiev, M. Ohkubo, Group to group commitments do not shrink, in D. Pointcheval, T. Johansson, editors, *EUROCRYPT 2012, volume 7237 of LNCS* (Springer, 2012), pp. 301–317

[9] M. Abe, M. Kohlweiss, M. Ohkubo, M. Tibouchi, Fully structure-preserving signatures and shrinking commitments, in *Advances in Cryptology—EUROCRYPT 2015—34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26—30, 2015, Proceedings, Part II* (2015), pp. 35–65

[10] M. Abe, M. Kohlweiss, M. Ohkubo, M. Tibouchi, Fully structure-preserving signatures and shrinking commitments. IACR ePrint Archive, Report 2015/076 (2015). http://eprint.iacr.org/2015/076. Accessed 2 Feb 2015

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

[12] G. Barthe, E. Fagerholm, D. Fiore, A. Scedrov, B. Schmidt, M. Tibouchi, Strongly-optimal structure preserving signatures from type II pairings: synthesis and lower bounds, in J. Katz, editor, *PKC 2015, Lecture Notes in Computer Science* (Springer, 2015) to appear

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

[14] M. Bellare, A. Palacio, The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols, in M. K. Franklin, editor, *CRYPTO, volume 3152 of LNCS* (Springer, 2004), pp. 273–289

[15] M. Bellare, H. Shi, C. Zhang, Foundations of group signatures: The case of dynamic groups, in *Topics in Cryptology—CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14–18, 2005, Proceedings* (2005), pp. 136–153

[16] M. Bellare, S. Shoup, Two-tier signatures, strongly unforgeable signatures, and Fiat-Shamir without random oracles, in *Public-Key Cryptography, volume 4450 of LNCS* (2007), pp. 201–216

[17] A. Bender, J. Katz, R. Morselli, Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology* **22**(1), 114–138 (2009)

[18] D. Boneh, X. Boyen, Short signatures without random oracles and the sdh assumption in bilinear groups. *J. Cryptology* **21**(2), 149–177 (2008)

[19] D. Boneh, X. Boyen, E. Goh, Hierarchical identity based encryption with constant size ciphertext. in *Advances in Cryptology—EUROCRYPT 2005, 24th Annual International Conference on the Theory and*

Applications of Cryptographic Techniques, Aarhus, Denmark, May 22–26, 2005, Proceedings (2005), pp. 440–456

[20] J. Camenisch, N. Chandran, V. Shoup, A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks, in A. Joux, editor, *Advances in Cryptology—EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26–30, 2009. Proceedings, volume 5479 of Lecture Notes in Computer Science* (Springer, 2009), pp. 351–368

[21] J. Camenisch, M. Dubovitskaya, K. Haralambiev, M. Kohlweiss, Composable and modular anonymous credentials: Definitions and practical constructions. in T. Iwata and J. H. Cheon, editors, *Advances in Cryptology—ASIACRYPT 2015—21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29–December 3, 2015, Proceedings, Part II, volume 9453 of Lecture Notes in Computer Science* (Springer, 2015), pp. 262–288

[22] J. Camenisch, K. Haralambiev, M. Kohlweiss, J. Lapon, V. Naessens, Structure preserving CCA secure encryption and applications. in D. H. Lee and X. Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science* (Springer, 2011), pp. 89–106

[23] J. Camenisch, S. Krenn, V. Shoup, A framework for practical universally composable zero-knowledge protocols, in *Advances in Cryptology—ASIACRYPT 2011—17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4–8, 2011. Proceedings* (2011), pp. 449–467

[24] J. Camenisch, A. Lysyanskaya, An efficient system for non-transferable anonymous credentials with optional anonymity revocation, in *Advances in Cryptology—EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6–10, 2001, Proceeding* (2001), pp. 93–118

[25] D. Catalano, M. D. Raimondo, D. Fiore, R. Gennaro, Off-line/on-line signatures: Theoretical aspects and experimental results. in *Public Key Cryptography—PKC 2008, 11th International Workshop on Practice and Theory in Public-Key Cryptography, Barcelona, Spain, March 9–12, 2008. Proceedings, volume 4939 of LNCS* (Springer, 2008), pp. 101–120

[26] M. Chase, M. Kohlweiss, A. Lysyanskaya, S. Meiklejohn, Malleable signatures: New definitions and delegatable anonymous credentials, in *2013 IEEE 27th Computer Security Foundations Symposium* (2014)

[27] S. Chatterjee, A. Menezes, Type 2 structure-preserving signature schemes revisited. IACR ePrint Archive, Report 2014/635 (2014). http://eprint.iacr.org/2014/635. Accessed 10 Sept 2015.

[28] I. Damgård, J. Groth, Non-interactive and reusable non-malleable commitment schemes, in L. L. Larmore and M. X. Goemans, editors, *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9–11, 2003, San Diego, CA, USA* (ACM, 2003), pp. 426–437

[29] A. Escala, J. Groth, Fine-tuning groth-sahai proofs, in *Public-Key Cryptography—PKC 2014—17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26–28, 2014. Proceedings* (2014), pp. 630–649

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

[31] M. Fischlin, Communication-efficient non-interactive proofs of knowledge with online extractors, in V. Shoup, editor, *Advances in Cryptology—CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14–18, 2005, Proceedings, volume 3621 of Lecture Notes in Computer Science* (Springer, 2005), pp. 152–168

[32] G. Fuchsbauer, Commuting signatures and verifiable encryption, in *Advances in Cryptology—EUROCRYPT 2011—30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15–19, 2011. Proceedings* (2011), pp. 224–245

[33] G. Fuchsbauer, C. Hanser, D. Slamanig, Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. Cryptology ePrint Archive, Report 2014/944 (2014). http://eprint.iacr.org/2014/944. Accessed 20 Mar 2016

[34] S. D. Galbraith, K. G. Paterson, N. P. Smart, Pairings for cryptographers. *Discrete Applied Mathematics* **156**(16), 3113–3121 (2008)

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

[36] J. Groth, Fully anonymous group signatures without random oracles, in *Advances in Cryptology—ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, December 2–6, 2007, Proceedings* (2007), pp. 164–180

[37] J. Groth, Efficient fully structure-preserving signatures for large messages, in *Advances in Cryptology—ASIACRYPT 2015—21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29–December 3, 2015, Proceedings, Part I* (2015), pp. 239–259

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

[39] S. Hada, T. Tanaka, On the existence of 3-round zero-knowledge protocols, in H. Krawczyk, editor, *Advances in Cryptology—CRYPTO '98, volume 1462 of LNCS* (Springer, 1998), pp. 354–369. Full version available from IACR e-print archive 1999/009

[40] T. Jager, F. Kohlar, S. Schäge, J. Schwenk, Generic compilers for authenticated key exchange, in *Advances in Cryptology—ASIACRYPT 2010—16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5–9, 2010. Proceedings* (2010), pp. 232–249

[41] B. Libert, T. Peters, M. Joye, M. Yung, Linearly homomorphic structure-preserving signatures and their applications, in R. Canetti and J. Garay, editors, *Advances in Cryptology—CRYPTO, LNCS* (Springer, 2013)

[42] U. M. Maurer, Abstract models of computation in cryptography, in N. P. Smart, editor, *Cryptography and Coding, 10th IMA International Conference, Cirencester, UK, December 19–21, 2005, Proceedings, volume 3796 of Lecture Notes in Computer Science* (Springer, 2005), pp. 1–12

[43] S. Meiklejohn, An extension of the Groth-Sahai proof system, in *Brown University Masters thesis* (2009)

[44] S. Micali, K. Ohta, L. Reyzin, Accountable-subgroup multisignatures: extended abstract, in *CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA, November 6–8, 2001* (2001), pp. 245–254

[45] V. I. Nechaev, Complexity of a determinate algorithm for the discrete logarithm. *Mat. Zametki* **55**(2), 91–101 (1994)

[46] T. Ristenpart, S. Yilek, The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks, in *Advances in Cryptology—EUROCRYPT 2007, 26th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Barcelona, Spain, May 20–24, 2007, Proceedings* (2007), pp. 228–245

[47] R. L. Rivest, A. Shamir, Y. Tauman, How to leak a secret, in *Advances in Cryptology—ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9–13, 2001, Proceedings* (2001), pp. 552–565

[48] V. Shoup, Lower bounds for discrete logarithms and related problems, in *EUROCRYPT, volume 1233 of LNCS* (1997), pp. 256–266

[49] N. Smart, F. Vercauteren, On computable isomorphisms in efficient asymmetric pairing-based systems. *Discrete Applied Mathematics* **155**(4), 538 – 547 (2007)

[50] Y. Wang, Z. Zhang, T. Matsuda, G. Hanaoka, K. Tanaka, How to obtain fully structure-preserving (automorphic) signatures from structure-preserving ones. in J. H. Cheon and T. Takagi, editors, *Advances in Cryptology—ASIACRYPT 2016—22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4–8, 2016, Proceedings, Part II, volume 10032 of Lecture Notes in Computer Science* (2016), pp. 465–495