How to Obtain Fully Structure-Preserving (Automorphic) Signatures from Structure-Preserving Ones

Yuyu Wang[1,2⋆], Zongyang Zhang[2 ⋆⋆], Takahiro Matsuda[2], Goichiro Hanaoka[2], and Keisuke Tanaka[1,3 ⋆⋆⋆]

[1] Tokyo Institute of Technology, Tokyo, Japan
[2] National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
[3] JST CREST, Tokyo, Japan
wang.y.ar@m.titech.ac.jp, zongyang.zhang@gmail.com, t-matsuda@aist.go.jp, hanaoka-goichiro@aist.go.jp, keisuke@is.titech.ac.jp

Abstract. In this paper, we bridge the gap between structure-preserving signatures (SPSs) and fully structure-preserving signatures (FSPSs). In SPSs, all the messages, signatures, and verification keys consist only of group elements, while in FSPSs, even signing keys are required to be a collection of group elements. To achieve our goal, we introduce two new primitives called *trapdoor signature* and *signature with auxiliary key*, both of which can be derived from SPSs. By carefully combining both primitives, we obtain generic constructions of FSPSs from SPSs. Upon instantiating the above two primitives, we get many instantiations of FSPS with unilateral and bilateral message spaces. Different from previously proposed FSPSs, many of our instantiations also have the *automorphic property*, i.e., a signer can sign his own verification key. As by-product results, one of our instantiations has the shortest verification key size, signature size, and lowest verification cost among all previous constructions based on standard assumptions, and one of them is the first FSPS scheme in the *type I* bilinear groups.

Keywords: signature, trapdoor signature, fully structure-preserving, automorphic.

⋆ This author is supported by a JSPS Fellowship for Young Scientists.
⋆⋆ Corresponding author whose current affiliation is Beihang University. The author is partly supported by the NSFC under No. 61303201. The work is done while the author is a postdoctoral researcher of AIST.
⋆⋆⋆ A part of this work was supported by a grant of I-System Co. Ltd., NTT Secure Platform Laboratories, Nomura Research Institute, Input Output Hongkong, and MEXT/JSPS KAKENHI 16H01705.

1 Introduction

1.1 Background

Structure-preserving signatures (SPSs). In [3], Abe et al. initiated the study of SPSs which denote pairing-based signatures where all the verification keys, messages, and signatures consist only of group elements and the verification algorithms only make use of pairing product equations (PPEs) to verify signatures.

SPSs are very useful since they can be combined with other structure-preserving (SP) primitives, e.g., ElGamal encryption [19] and Groth-Sahai proofs [29], to obtain efficient cryptographic protocols such as blind signatures [3,25,24,23], group signatures [3,25,34], homomorphic signatures [33], delegatable anonymous credentials [22], compact verifiable shuffles [17], network coding [6], oblivious transfer [37,14], tightly secure encryption [30,2], and e-cash [7]. Motivated by this, there have been a large deal of works focusing on SPSs (e.g., [3,1]) in the past few years, which provide us with various SPS schemes based on different assumptions and with high efficiency.

Automorphic signatures. In [3], Abe et al. noted that for elaborate applications, the SP property of a signature scheme is not sufficient. In addition, an SPS scheme has to be able to sign its own verification keys, i.e., verification keys have to lie in the message space. They called such kind of SPS *automorphic signature* and gave an instantiation of it, and also provided a generic transformation that converts automorphic signatures for messages of fixed length into ones for messages of arbitrary length.

As argued in [3], since automorphic signatures enable constructions of certification chains (i.e., sequences of verification keys linked by certificates from one key on the next one), they are useful in constructing anonymous proxy signatures and delegatable anonymous credentials. Abe et al. [3] also showed how to combine automorphic signatures with the Groth-Sahai proof system to construct a round-optimal blind signature scheme.

Fully structure-preserving signatures (FSPSs). In [5], Abe et al. introduced FSPSs, where signing keys also consist only of group elements and the correctness of signing keys with respect to verification keys can be verified by PPEs. Since the fully structure-preserving (FSP) property enables efficient signing key extraction, it could help us prevent rogue-key attacks in the public-key infrastructures (PKIs) [36], make anonymous credentials UC-secure [15], achieve privacy in group and ring signatures [10,11,13] in the presence of adversarial keys, and extend delegatable anonymous credentials [8,22,18] with all-or-nothing transferability [16], as noted in [5]. In this paper, we call an automorphic signature scheme that is FSP a *fully automorphic signature (FAS) scheme*.

Abe et al. [5] gave two generic constructions by combining FSPSs unforgeable (UF) against extended random message attacks (xRMA) [1] with other primitives such as one-time SPSs, two-tier SPSs (also called partial one-time SPSs), and trapdoor commitment schemes. Although these constructions are novel and

neat, they suffer from three shortcomings due to the use of specific primitives, which make them less generic.

1. As both constructions require a UF-xRMA secure FSPS scheme and one of them also requires a γ-blinding trapdoor commitment scheme, the underlying assumptions and bilinear map of their instantiations are limited. Concretely speaking, all the signature schemes derived from their constructions have to be based on at least the SXDH and XDLIN assumptions and be in the *type III* bilinear group.
2. For the same reason, the efficiency of their instantiations is also potentially limited by the underlying UF-xRMA secure FSPS scheme and the γ-blinding trapdoor commitment scheme. For example, the verification keys and signatures of their most efficient FSPS scheme consist of more than $10n$ group elements in total if messages consist of n^2 group elements.
3. Their instantiations are not automorphic. The reason is that verification keys of the UF-xRMA secure FSPS scheme (which are also verification keys of the resulting schemes) consist of elements in both source groups, while the resulting signature schemes can only sign messages consisting only of elements in one source group.

Note that Abe et al. [5] also gave a variant of their constructions by combining a UF-xRMA secure signature scheme and a trapdoor commitment scheme with SPSs, which can be treated as a generic transformation from SPSs to FSPSs. If the instantiation of SPS is with a bilateral message space (i.e., messages consist of elements in both source groups), then the resulting signature scheme could be automorphic. However, as far as we know, besides the aforementioned shortcomings, all the previously proposed SPS schemes with a bilateral message space require verification keys to consist of elements in both source groups (except for ones that sign messages of "DDH form" [27,26]), which result in very inefficient FSPS schemes, as noted in [5]. The verification keys and signatures (respectively, the verification algorithm) of the most efficient automorphic instantiation that can be derived from their generic construction consist of more than $12n$ group elements in total (respectively, more than $3n$ PPEs) if the messages consist of $2n^2$ group elements.

Following the work of Abe et al. [5], Groth [28] gave an elegant construction of FSPS, which has the shortest verification keys and signatures, and needs the fewest PPEs for verification. Although this FSPS scheme is the most efficient one as far as we know, it is only known to be secure in the generic group model and is not automorphic.

Up until now, a lot of results are devoted to constructing efficient SPSs under different assumptions, while there are very few FSPS schemes. If we can find a generic method to transform existing SPSs into FSPSs or even FASs without directly using specific primitives, it will greatly alleviate the efforts to construct them from scratch.

1.2 Our Results

Generic construction of FSPS. In this paper, we formalize two extensions to ordinary signatures called *trapdoor signatures (TSs)* and *signatures with auxiliary key (AKSs)*. We show that any well-formed[4] SPS scheme can be converted into a TS scheme satisfying the signing key structure-preserving (SKSP) property, in which signing keys consist only of group elements and the correctness with respect to verification keys can be verified by PPEs, while messages are not necessarily group elements. Furthermore, it is relatively straightforward to show that any SPS scheme with an algebraic key generation algorithm can be converted into a structure-preserving signature with auxiliary keys (SP-AKS). By combining SKSP-TS with SP-AKS, we obtain a generic construction of FSPS.[5] Our construction implies that for any two SPS schemes, if verification keys of one lie in the message space of the other (which is well-formed), then basically, they can be used to construct an FSPS scheme, without using any other specific primitives or additional assumptions. It also implies that most well-formed SPS schemes with a bilateral message space or unilateral verification key space (i.e., the verification keys consist only of elements in one source group) can be converted into an FSPS scheme.

This generic construction is proved to be secure based on building blocks satisfying different security, which allows us to obtain various instantiations of FSPS based on different assumptions.

Efficient instantiations of FSPSs. By extending the definition of AKSs to two-tier signatures with auxiliary keys (TT-AKSs) and substituting AKSs with TT-AKSs in the above generic construction, we obtain another generic construction, which enables us to obtain more efficient instantiations of FSPS. For instance, by using the TS scheme and TT-AKS scheme adapted from the SPS schemes proposed by Kiltz et al. [31,32], we obtain instantiations of FSPS with unilateral and bilateral message spaces. We give an efficiency comparison between our instantiations and the ones proposed in [5] in Table 1.[6] Note that like the FSPS scheme proposed in [28], a signing key in our instantiations consists of $\Omega(n)$ group elements (concretely, $2n+1$ in [28] and $4n+9$ and $8n+13$ in our results), while that in "`AKO+15`" consists only of 4 elements. However, in many applications, the

[4] We refer the reader to Definition 10 for details of well-formed SPSs. As far as we know, all the existing SPS schemes are well-formed.

[5] As in [5], we assume the underlying SKSP-TS scheme and SP-AKS scheme share the common setup algorithm.

[6] The second instantiation in Table 1 is derived from the generic construction described in [5, Section 6.4], where the underlying SPS scheme is the one with bilateral message space in [31] (based on the SXDH assumption). In this instantiation, we have to add a group element denoting the sequence number to every message block. Furthermore, the underlying two-tier signature schemes of the first and third instantiations have the same efficiency, which makes sure that this comparison is fair. If we allow trusted setup besides the bilinear map generation, the sizes of common parameters $|par|$ in these four schemes are 6, 6, 1, and 2 respectively.

size of a signing key does not have to be "extremely short" since typically, a user generates only one proof for knowing a signing key (e.g., in PKIs and group/ring signatures), while proofs for knowing a signature or a verification key/signature pair are required to be generated for multiple times.[7]

	Security	Assumption	$\|m\|$	$\|pk\| + \|par\|$	$\|\sigma\|$	♯ PPE
AKO+15 [5]	Full	SXDH, XDLIN	$(n^2, 0)$	$6n + 17$	$4n + 11$	$n + 5$
	Full	SXDH, XDLIN	(n^2, n^2)	$6n + 47$	$13n + 30$	$5n + 6$
Our results	Full	SXDH	$(n^2, 0)$	$2n + 7$	$4n + 8$	$n + 3$
	Full	SXDH	(n^2, n^2)	$4n + 10$	$8n + 12$	$2n + 4$

Table 1. Comparison between the most efficient instantiations of FSPS based on standard assumptions derived from the main construction in [5] and the most efficient ones derived from our constructions. Notation (x, y) denotes x elements in $\mathbb{G}_1$ and y elements in $\mathbb{G}_2$. We do not count the two generators in the description of bilinear groups when giving the parameters.

Our FSPS schemes in Table 1 can also be based on the $\mathcal{D}_k$-matrix Diffie-Hellman (MDDH) assumptions [20] (see the full paper for the definition), while the parameters become $(|m|, |pk| + |par|, |\sigma|, \sharp\text{PPE}) = (n^2, (2nk + 2k + 3 + \mathsf{RE}(\mathcal{D}_k))k + \mathsf{RE}(\mathcal{D}_k), (3k + 1)n + 4 + 3k + \mathsf{RE}(\mathcal{D}_k), kn + 2k + 1)$ and $(|m|, |pk|+|par|, |\sigma|, \sharp\text{PPE}) = (2n^2, (4nk+3k+3+2\mathsf{RE}(\mathcal{D}_k))k+2\mathsf{RE}(\mathcal{D})_k, 2(3k+1)n+5k+5+2\mathsf{RE}(\mathcal{D}_k), 2kn+3k+1)$, where $\mathsf{RE}(\mathcal{D}_k)$ denotes the minimal number of group elements needed to present a matrix sampled from $\mathcal{D}_k$.

Since our constructions only require the underlying schemes to have properties naturally satisfied by SPSs, further improvement on SPS schemes may contribute to the efficiency of FSPSs more via our constructions than the constructions in [5].

FASs. Since we can convert any (well-formed) SPS scheme into an SKSP-TS scheme and an SP-AKS scheme, our generic constructions also derive many instantiations of FAS from various combinations (including the ones in Table 1). As long as verification keys of the underlying TS scheme consist of no more group elements than messages of the underlying AKS scheme in both source groups, the resulting scheme is fully automorphic.

We can instantiate our first generic construction with the TS scheme and AKS scheme adapted from the SPS scheme proposed by Groth et al. [28] to obtain our most efficient FAS scheme, while the most efficient one from the generic construction in [5] can be obtained by letting the underlying SPS scheme

[7] The argument that the signing key size is not as important as verification/signature size does not spoil the motivation for FSPS. FSPS helps avoid extremely heavy key extraction, i.e., extracting a signing key bit by bit (see Introduction in [8]). However, this does not mean we have to make the extraction extremely light. Allowing checking signing keys by using PPEs and keeping the key size linear with message size is enough to achieve the goal.

be the one in [4] and the underlying one-time SPS scheme the one in [26]. For ease of understanding, we give an efficiency comparison in Table 2.

	Security	Assumption	$\|m\|$	$\|pk\| + \|par\|$	$\|\sigma\|$	♯ PPE
AKO+15 [5]	Full	Generic	(n^2, n^2)	$6n + 23$	$6n + 14$	$3n + 6$
Our result	Full	Generic	$(n^2, 0)$	$2n + 1$	$2n + 5$	$n + 3$

Table 2. Comparison between the most efficient instantiation of FAS derived from the main construction in [5] and the most efficient one derived from our constructions. Both of them are secure in the generic group model.

FSPS (FAS) schemes in the symmetric (type I) bilinear map. We also instantiate our generic constructions with the SPS scheme and the tag-based SPS scheme proposed in [2] to obtain the first FSPS and FAS schemes in the type I bilinear map, the most efficient one of which achieves $(|m|, |pk| + |par|, |\sigma|, \sharp\text{PPE}) = (n^2, 6n + 30, 6n + 12, 2n + 7)$.

1.3 High-level Idea

Our generic construction can be treated as an extension of the well-known EGM paradigm [21]. In this paradigm a signer uses two signature schemes Σ_1 and Σ_2 to sign a message m. It first signs m by using the signing key sk_2 of Σ_2 and then signs the verfication key vk_2 of Σ_2 by using the signing key sk_1 of Σ_1. This paradigm was used to obtain SPSs in [1] and a generic construction of FSPS in [5]. To make sure that the resulting signature scheme is an FSPS scheme, it is natural to require sk_1 to consist only of group elements. This is the reason why Abe et al. [5] instantiated Σ_1 with the xRMA secure signature scheme proposed in [1], which was the only proposed FSPS scheme until then. However, we observe that it is possible to instantiate Σ_1 with all the existing SPS schemes, which also provides us with more options when selecting instantiations of Σ_2 to match Σ_1.

Next, we explain how to choose Σ_1 and Σ_2, and the high level idea of our construction. Roughly speaking, starting from an SPS scheme with a signing key $x \in \mathbb{Z}_p$, we can always derive a signature scheme in which the signing key becomes a group element $X = G^x \in \mathbb{G}$ (where G denotes the generator of $\mathbb{G}$). It is obvious that in this case a message $M \in \mathbb{G}$ cannot be signed by using X since we are not able to compute M^x from X and M. Supposing that $M = G^m$, we can use X to sign m instead of M, i.e., compute X^m instead of M^x when generating a signature. Furthermore, since signatures generated in this way are the same as those generated by the real signing key, and the public key and verification algorithm remain the same, one can verify the signature by using M. We formalize such a signature scheme as a TS scheme. Although such a signature scheme is only "semi"-structure-preserving, we use it to sign the exponent $v \in \mathbb{Z}_p$ of a verification key (called auxiliary keys) of another SPS scheme and use the latter SPS scheme to sign a message $M' \in \mathbb{G}'$. This enables us to obtain an FSPS

scheme. We formalize the latter signature scheme which generates auxiliary keys besides verification/signing key pairs as an AKS scheme.

To verify a signature, one only needs to know $V = G^v$ and M', without knowing v. Furthermore, the original signing key x (called trapdoor key) of the TS scheme is never used in the signing process but is necessary as the reduction algorithm in the security proof signs verification keys without knowing the exponent.

Our main contributions lie in two aspects. First, we formalize the notions of TSs and AKSs in order to adapt the EGM paradigm to construct FSPSs. Second, we show that most of existing SPS schemes can be cast as our extended signatures, and consequently we can obtain a number of FSPSs and FASs based on existing SPSs.

Perhaps interestingly, although most of the previously proposed SPS schemes with a unilateral message space are not automorphic (since their verification keys and messages usually consist of elements in different source groups), when some of them are converted into FSPSs using our method, the resulting schemes become automorphic.[8]

Paper organization. We recall several definitions in Section 2. Then we formalize TSs and AKSs and show how to instantiate them from any (well-formed) SPS scheme in Section 3 and Section 4 respectively, and give generic constructions of FSPSs based on them in Section 5. Finally, we show instantiations of our generic constructions in Section 6.

2 Preliminary

2.1 Notations

In this paper, we let *negl* be negligible functions, $[n]$ the set $\{1, \ldots, n\}$, $\mathbb{N}$ the set of natural numbers, $|X|$ the number of elements in X (where X could be a space, a vector, or a matrix), and $\widetilde{\mathbf{A}}$ the $1 \times mn$ vector $(a_{11}, a_{12}, \ldots a_{1n}, a_{21}, a_{22}, \ldots a_{2n}, \ldots, a_{m1}, a_{m2}, \ldots a_{mn})$ where $\mathbf{A}$ denotes the $m \times n$ matrix $(a_{ij})_{i \in [m], j \in [n]}$. If $\mathbf{A} \in \mathbb{Z}_p^{(k+1) \times k}$ lies in the matrix distribution $\mathcal{D}_k$, then we use $\overline{\mathbf{A}}$ to denote the upper square matrix of $\mathbf{A}$. Furthermore, $\vec{a} \in \mathbb{Z}_p^n$ denotes a column vector by default.

2.2 Pairing Group

In this paper, we let $\mathcal{G}$ be an algorithm that takes as input 1^λ and outputs $gk = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G_1, G_2)$ such that p is a prime satisfying $p = \Theta(2^\lambda)$, $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ are descriptions of groups of order p, G_1 and G_2 generate $\mathbb{G}_1$

[8] When messages and verification keys of the underlying TS scheme consist of elements in $\mathbb{G}_2$ and $\mathbb{G}_1$ respectively and those of the underlying AKS scheme consist of elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, verification keys and messages of the resulting FSPS scheme consist of elements only in $\mathbb{G}_1$.

and $\mathbb{G}_2$ respectively, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is an efficiently computable (non-degenerate) bilinear map. Following [31] and [28], we use the additive notation in [20] such as $e((a+b)[x]_1, [y]_2) = a \cdot e([x]_1, [y]_2) + b \cdot e([x]_1, [y]_2)$ where $[x]_1$ and $[y]_2$ denote G_1^x and G_2^y respectively, and $e([x]_1, [y]_2)$ can be written as $[xy]_T$. Furthermore, $e([\vec{a}]_1^\top, [\vec{b}]_2)$ denotes $\sum_{i=1}^{n} e([a_i]_1, [b_i]_2)$ where $[\vec{a}]_1 = ([a_1]_1, \ldots, [a_n]_1)^\top$ and $[\vec{b}]_2 = ([b_1]_2, \ldots, [b_n]_2)^\top$, and $e([\mathbf{A}]_1^\top, [\mathbf{B}]_2)$ denotes $(e([\vec{a}_i^\top]_1, [\vec{b}_j]_2))_{i \in [n], j \in [n']}$ where $[\mathbf{A}]_1 = ([\vec{a}_1]_1, \ldots, [\vec{a}_n]_1)$ and $[\mathbf{B}]_2 = ([\vec{b}_1]_2, \ldots, [\vec{b}_{n'}]_2)$.

2.3 Signatures

Definition 1 (Signature) *A* signature scheme *consists of four polynomial-time algorithms* Setup, Gen, Sign, *and* Verify. Setup *takes as input a security parameter 1^λ and generates a public parameter par, which determines the message space $\mathcal{M}$ and the randomness space $\mathcal{R}$ for signing.* Gen *is a randomized algorithm that takes as input a public parameter par and outputs a verification/signing key pair (pk, sk).* Sign *is a randomized algorithm that takes as input a signing key sk and a message m, and returns a signature σ.* Verify *is a deterministic algorithm that takes as input a verification key pk, a message M, and a signature σ, and returns* 1 *(accept) or* 0 *(reject).*

The correctness is satisfied if we have $\mathsf{Verify}(pk, m, \mathsf{Sign}(sk, m; r)) = 1$ *for all* $\lambda \in \mathbb{N}$, $par \leftarrow \mathsf{Setup}(1^\lambda)$, $(pk, sk) \leftarrow \mathsf{Gen}(par)$, $m \in \mathcal{M}$, *and* $r \in \mathcal{R}$.

In [3], Abe et al. firstly defined *SPSs*, in which verification keys, messages, and signatures consist only of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$, and signatures are verified by evaluating pairing product equations (PPEs), which are of the form $\sum_{ij} a_{ij} e([x_i]_1, [y_j]_2) = [0]_T$, where a_{ij} is an integer constant for all i and j.

Definition 2 (Structure-preserving signature (SPS)) *A signature scheme is said to be* structure-preserving *over a bilinear group generator $\mathcal{G}$ if we have (a) a public parameter includes a group description gk generated by $\mathcal{G}$, (b) verification keys consist of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$, (c) messages consist of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$, (d) signatures consist of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$, and (e) the verification algorithm consists only of evaluating membership in $\mathbb{G}_1$ and $\mathbb{G}_2$ and relations described by PPEs.*

SPSs are versatile since they mix well with other pairing-based protocols. Especially, they are compatible with the Groth-Sahai proof system [29]. However, as argued by Abe et al. in [3], Groth-Sahai compatibility of a signature scheme is not sufficient for elaborate applications such as anonymous signatures and delegatable anonymous credentials, which require signatures on verification keys to obtain anonymized certification chains. Abe et al. [3] called an SPS scheme that is able to sign its own verification keys an automorphic signature scheme.

Definition 3 (Automorphic signature) *A signature scheme is said to be an* automorphic signature scheme *over a bilinear group generator $\mathcal{G}$ if it is structure-preserving and the elements in $\mathbb{G}_1$ (respectively, $\mathbb{G}_2$) of every verification key are not more than the group elements in $\mathbb{G}_1$ (respectively, $\mathbb{G}_2$) of every message.*

In [5], Abe et al. introduced FSPSs, which also require a signing key to be group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ and the correctness of a signing key with respect to a verification key can be verified by PPEs. Such signatures allow efficient key extraction when combined with non-interactive proofs (e.g., the Groth-Sahai proofs), which may help prevent rogue-key attacks [36], build UC-secure privacy preserving protocols [15], strengthen privacy in group and ring signatures [10,11,13] in the presence of adversarial keys, and extend delegatable anonymous credential systems [8,22,18] with all-or-nothing transferability [16].

Definition 4 (Fully structure-preserving signature (FSPS)) *A structure-preserving signature scheme* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *with the message space* $\mathcal{M}$ *and randomness space* $\mathcal{R}$ *for signing is said to be* fully structure-preserving *if we have (a) signing keys consist only of group elements in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*, and additionally, (b) there exists a polynomial-time deterministic algorithm* $\mathsf{VerifySK}$ *that takes as input a verification/signing key pair and consists only of evaluating membership in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$ *and relations described by PPEs, and it is required that for sufficiently large* $\lambda \in \mathbb{N}$*,* $par \leftarrow \mathsf{Setup}(1^\lambda)$*, the following holds:*

- $\mathsf{VerifySK}(pk, sk) = 1$ *if and only if* $\mathsf{Verify}(pk, m, \mathsf{Sign}(sk, m; r)) = 1$ *holds for all* $m \in \mathcal{M}$ *and* $r \in \mathcal{R}$*.*

In this paper, we call an automorphic signature scheme which is also FSP a *fully automorphic signature (FAS) scheme.*

Definition 5 (Fully automorphic signature (FAS)) *An automorphic signature scheme is said to be* fully automorphic *if it is also fully structure-preserving.*

Due to space limitation, we recall the UF-CMA, UF-RMA, UF-otCMA, and UF-otRMA security of a signature scheme in the full paper.

3 Trapdoor Signatures

3.1 Definition of Trapdoor Signatures

In this section, we formalize the notion of *γ-trapdoor signature (γ-TS) scheme*, whose instantiations are used as building blocks to obtain FSPSs. Different from standard signatures, a TS scheme verifies the correctness of a signature σ on a message $m \in \mathcal{M}$ by taking as input $(\gamma(m) \in \mathcal{M}_\gamma, \sigma)$ where $\gamma : \mathcal{M} \mapsto \mathcal{M}_\gamma$ is an efficiently computable bijection. Furthermore, there exists a trapdoor key with which we can generate a signature on m if we have $\gamma(m)$ but not m itself.

Definition 6 (γ-Trapdoor signature (γ-TS)) *A* γ-trapdoor signature scheme *consists of five polynomial-time algorithms* Setup*,* Gen*,* Sign*,* Verify*, and* TDSign*.* Setup *takes as input a security parameter* 1^λ *and generates a public parameter* par*, which determines the message space* $\mathcal{M}$ *for the signing algorithm, the message space* $\mathcal{M}_\gamma$ *for the verification algorithm, and an efficiently*

computable bijection $\gamma : \mathcal{M} \mapsto \mathcal{M}_\gamma$. Gen *is a randomized algorithm that takes as input par, and outputs a verification/signing key pair* (pk, sk) *and a trapdoor key* tk. Sign *is a randomized algorithm that takes as input a signing key* sk *and a message* $m \in \mathcal{M}$*, and returns a signature* σ*, where the randomness space is denoted by* $\mathcal{R}$. Verify *is a deterministic algorithm that takes as input a verification key* pk*, a message* $M \in \mathcal{M}_\gamma$*, and a signature* σ*, and returns* 1 *(accept) or* 0 *(reject).* TDSign *takes as input a trapdoor key* tk *and a message* $M \in \mathcal{M}_\gamma$*, and returns a signature* σ*. The randomness space of* TDSign *is also* $\mathcal{R}$.

The correctness is satisfied if for all $\lambda \in \mathbb{N}$*,* $par \leftarrow \mathsf{Setup}(1^\lambda)$*,* $((pk, sk), tk) \leftarrow \mathsf{Gen}(par)$*, and* $m \in \mathcal{M}$*, we have (a)* $\mathsf{Verify}(pk, \gamma(m), \mathsf{Sign}(sk, m)) = 1$*, and (b)* $\mathsf{Sign}(sk, m; r) = \mathsf{TDSign}(tk, \gamma(m); r)$ *for all* $r \in \mathcal{R}$.

Key generation algorithm $\mathcal{T}_{\mathsf{Gen}}$. We use $\mathcal{T}_{\mathsf{Gen}}$ to denote an algorithm that runs Gen, which is the key generation algorithm of a TS scheme, in the following way. Taking as input a public parameter par, $\mathcal{T}_{\mathsf{Gen}}$ gives par to Gen and obtains an output $((pk, sk), tk)$. Then $\mathcal{T}_{\mathsf{Gen}}$ outputs (pk, tk) as a verification/signing key pair.

For a TS scheme $\Sigma = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign})$, we denote $(\mathsf{Setup}, \mathcal{T}_{\mathsf{Gen}}, \mathsf{TDSign}, \mathsf{Verify})$ by $\mathcal{T}_\Sigma$. According to the syntax of TS, it is not hard to see that $\mathcal{T}_\Sigma$ forms a standard signature scheme whose message space is $\mathcal{M}_\gamma$.

Now we define SKSP-TSs, in which verification keys, signing keys, and signatures (but not necessarily messages) consist only of group elements, and the correctness of signing keys with respect to verifications keys can be verified by PPEs.

Definition 7 (Signing key structure-preserving (SKSP)) *A* γ*-TS scheme* $\Sigma = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign})$ *with message space* $\mathcal{M}$ *is said to be* signing key structure-preserving *over a bilinear group generator* $\mathcal{G}$ *if we have (a)* $\mathcal{T}_\Sigma$ *is an SPS scheme, (b) signing keys (rather than trapdoor keys) consist only of group elements in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*, and (c)* Σ *satisfies the condition (b) in Definition 4, where* $\mathsf{Verify}(pk, m, \mathsf{Sign}(sk, m; r)) = 1$ *is replaced with* $\mathsf{Verify}(pk, \gamma(m), \mathsf{Sign}(sk, m; r)) = 1$.

Note that different from FSPSs, messages are not required to be group elements in SKSP-TSs.

3.2 Security of Trapdoor Signatures

We now define the UF-CMA security of TSs.

Definition 8 (UF-CMA of TSs) *A* γ*-TS scheme* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign})$ *is said to be* unforgeable against chosen message attacks (UF-CMA) *if for every probabilistic polynomial time (PPT) adversary* $\mathcal{A}$*, we have*

$$\Pr[par \leftarrow \mathsf{Setup}(1^\lambda), ((pk, sk), tk) \leftarrow \mathsf{Gen}(par), (M^*, \sigma^*) \leftarrow \mathcal{A}^{\mathrm{SignO}(\cdot)}(par, pk) : M^* \notin \mathcal{Q}_m \wedge M^* \in \mathcal{M}_\gamma \wedge \mathsf{Verify}(pk, M^*, \sigma^*) = 1] \leq negl(\lambda)$$

where $\mathrm{SignO}(\cdot)$ *is the signing oracle that takes as input* $m \in \mathcal{M}$*, runs* $\sigma \leftarrow \mathsf{Sign}(sk, m)$*, adds* $\gamma(m) \in \mathcal{M}_\gamma$ *to* $\mathcal{Q}_m$*, and returns* σ*.*

Unlike the UF-CMA security of standard signatures, a query m made by an adversary is in $\mathcal{M}$, the signing oracle records $\gamma(m) \in \mathcal{M}_\gamma$, and the message M^* output by the adversary is in $\mathcal{M}_\gamma$.

The UF-CMA security of TSs is similar to the F-unforgeability of standard signatures defined by Belenkiy et al. [9]. Moreover, Libert et al. [35] gave an instantiation of F-unforgeable signatures and combined it with a tagged one-time signature scheme proposed by Abe et al. [2] to obtain a very efficient SPS scheme. However, they neither provided generic constructions nor considered the FSP property.

Now we show the relation between the UF-CMA security of $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign})$ and that of $(\mathsf{Setup}, \mathcal{T}_{\mathsf{Gen}}, \mathsf{TDSign}, \mathsf{Verify})$ in Theorem 1. We refer the reader to the full paper for the proof.

Theorem 1 *For a* γ*-TS scheme* $\Sigma = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign})$*, if* $\mathcal{T}_\Sigma = (\mathsf{Setup}, \mathcal{T}_{\mathsf{Gen}}, \mathsf{TDSign}, \mathsf{Verify})$ *is UF-CMA secure, then* Σ *is UF-CMA secure.*

Now we give the definitions of unforgeability against random message attacks (RMA), one-time chosen message attacks (otCMA), and one-time random message attacks (otRMA) of TSs.

Definition 9 (UF-RMA, UF-otCMA, and UF-otRMA of TSs) *The* UF-RMA *security of TSs is the same as the UF-CMA security of TSs except that to answer a signing query,* $\mathrm{SignO}(\cdot)$ *randomly chooses* $m \leftarrow \mathcal{M}$ *itself, runs* $\sigma \leftarrow \mathsf{Sign}(sk, m)$*, adds* $\gamma(m)$ *to* $\mathcal{Q}_m$ *(initialized with* $\emptyset$*), and returns* (m, σ)*.*

The UF-otCMA *(respectively,* UF-otRMA*) security is the same as the UF-CMA (respectively, UF-RMA) security of TSs, except that* $\mathcal{A}$ *is only allowed to make one query to the signing oracle* $\mathrm{SignO}(\cdot)$*.*

3.3 Converting Structure-Preserving Signatures into Signing Key Structure-Preserving Trapdoor Signatures

Before showing our conversion, we define a class of SPSs called well-formed SPSs. Roughly speaking, for a well-formed SPS scheme, it is required that the spaces of randomness and exponents of messages are super-polynomially large in the security parameter, and generating a signature element only involves the group operation, while the scalars of group elements are computed as arithmetic circuits of elements in the signing key and the randomness.

Definition 10 (Well-formed SPS) *For an SPS scheme* Σ*, let* $\mathbb{M}_1 \times \mathbb{M}_2 \times \ldots \times \mathbb{M}_n$ *be the space of exponents (with* $[1]_1$ *and* $[1]_2$ *for bases) of elements in a message,*[9] *and* $\mathbb{R}_1 \times \mathbb{R}_2 \times \ldots \times \mathbb{R}_{n'}$ *the randomness space (for signing), where*

[9] We do not count repeated message spaces, e.g., when messages are of the form $([m]_1, [m]_2)$ where $m \in \mathbb{Z}_p$, we have $n = 1$ and $\mathbb{M}_1 = \mathbb{Z}_p$.

$n, n' \in \mathbb{N}$. Σ is said to be well-formed if (a) for all i, $\mathbb{M}_i, \mathbb{R}_i \subseteq \mathbb{Z}_p$ and $|\mathbb{M}_i|$ and $|\mathbb{R}_i|$ are super-polynomial in the security parameter,[10] *and (b) generating a group element $[B]_b$ where $b \in \{1, 2\}$ in a signature only involves computing*

$$[B]_b = \sum_i (\prod_j a_{ij}^{c_{ij}})[A_i]_b, \tag{1}$$

where $\{[A_i]_b\}_i$ denotes elements appearing in the public parameters, the message, and the signing key, $\{a_{ij}\}_{ij}$ denotes elements (in $\mathbb{Z}_p$) appearing in the signing key and the randomness for signing, and integer constants, and $\{c_{ij}\}_{ij}$ denotes integer constants. Here, elements in $\{[A_i]_b\}_i$ may represent the same variables, and the same argument is made for $\{a_{ij}\}_{ij}$.[11]

Note that there is no requirement on the distributions of the elements other than the space sizes in the above definition, and as far as we know, all the existing SPSs are well-formed. Now we show that any well-formed SPS scheme can be converted into an SKSP-TS scheme.

Theorem 2 *Any well-formed SPS scheme, the messages of which are supposed to be of the form $([\vec{M}]_1, [\vec{N}]_2)$, can be converted into a γ-SKSP-TS scheme for γ defined by $\gamma(\vec{M}, \vec{N}) = ([\vec{M}]_1, [\vec{N}]_2)$.*

Schwartz-Zippel Lemma. Now we introduce Schwartz-Zippel Lemma [38], based on which we will give the proof of Theorem 2,

Lemma 1. *([38]) Let $P \in F[x]$ be a non-zero polynomial of total degree $d \geq 0$ over a field, S a finite subset of F, and r a randomness uniformly chosen from S. Then, we have*

$$\Pr[P(r) = 0] \leq d/|S|.$$

This lemma indicates that a polynomial of degree d over $\mathbb{Z}_p$ has at most d roots.

Proof (of Theorem 2). We divide the proof of Theorem 2 into two parts. In the first part, we show that any well-formed SPS scheme can be converted into a γ-TS scheme satisfying the conditions (a) and (b) of the SKSP property in Definition 7. In the second part, we prove that the converted TS scheme also satisfies the condition (c).

[10] It is not hard to see that an SPS scheme whose messages are of the form, e.g., $([m_1]_1, [m_2]_2)$ where $m_1 = m_2 + 1$ and $m_1, m_2 \in \mathbb{Z}_p$, is not well-formed. However, such a scheme can be easily converted to a well-formed one by letting messages be of the form $([m_1]_1, [m_1]_2)$ and compute $[m_1 + 1]_2$ in signing and verification.

[11] For ease of understanding, we give an example here. Supposing that an element in a signature is generated as $(r_1 s_1 + r_2^2 r_1)[U]_1 + s_2^{-1}[M]_1 + [S]_1$, where (r_1, r_2), $(s_1, s_2, [S]_1)$, $[U]_1$, and $[M]_1$ are respectively element(s) in the randomness, signing key, verification key, and message, then we express the formula as $(a_{11}^{c_{11}} a_{12}^{c_{12}})[A_1]_1 + (a_{21}^{c_{21}} a_{22}^{c_{22}})[A_2]_1 + a_{31}^{c_{31}}[A_3]_1 + [A_4]_1$, where $([A_1]_1, [A_2]_1, [A_3]_1, [A_4]_1, a_{11}, a_{12}, a_{21}, a_{22}, a_{31})$ represents $([U]_1, [U]_1, [M]_1, [S]_1, r_1, s_1, r_2, r_1, s_2)$ and $(c_{11}, c_{12}, c_{21}, c_{22}, c_{31}) = (1, 1, 2, 1, -1)$.

Part I. Let a group element in a signature be generated as Equation (1). For all i such that $\{a_{ij}\}_j$ contains a set of variables in the signing key, denoted by $\{s_{ij}\}_j$, we use c'_{ij} to denote the exponent of s_{ij} in Equation (1), and do the following conversion.

- If $[A_i]_b$ is in the message, then we add $[(\prod_j s_{ij}^{c'_{ij}})]_b$ to the signing key.
- Otherwise (i.e., if $[A_i]_b$ is in the signing key or the verification key), then we add $[(\prod_j s_{ij}^{c'_{ij}})A_i]_b$ to the signing key,

For all other group elements in the signature, we execute the same conversions. Then we remove all elements in $\mathbb{Z}_p$, all repeated elements, and elements never used in signing procedures from the original signing key, and set the original signing key as the trapdoor key.

By using the new signing key, we can generate a signature consisting of group elements in the forms of Equation (1) when taking as input a message consisting of $M_1, M_2, \ldots, N_1, N_2, \ldots \in \mathbb{Z}_p$, which forms the signing algorithm for the resulting γ-TS scheme. Furthermore, taking as input $[M_1]_1, [M_2]_1, \ldots, [N_1]_2, [N_2]_2, \ldots$ and the trapdoor key, we can generate the same signature if the randomness is the same, by using the original signing algorithm. As a result, we have obtained a γ-TS scheme for $\gamma(\vec{M}, \vec{N}) = ([\vec{M}]_1, [\vec{N}]_2)$.

It is straightforward to see that in this γ-TS scheme, the verification keys, signing keys, and signatures consist only of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ and the verification consists only of evaluating membership in $\mathbb{G}_1$ and $\mathbb{G}_2$ and relations described by PPEs. This completes the first part of the proof. Here, the verification key size, signature size, and number of PPEs do not change during the conversion, while the signing key size changes depending on the concrete construction of the SPS scheme.[12]

Part II. Next we prove that for the above γ-TS scheme, there exists an algorithm that can check the correctness of signing keys with respect to verification keys by using only PPEs.

Since a group element in the signature is computed as Equation (1), and a group element in the message $[M]_1$ or $[N]_2$ can be treated as $M[1]_1$ or $N[1]_2$, a PPE in the verification algorithm can be written as

$$\sum_i \Big(\prod_j {x_{ij}}^{d_{ij}} \Big) [X_i]_T = [0]_T, \tag{2}$$

where $\{x_{ij}\}_{ij}$ denotes elements in the randomness, exponents of the message, and integer constants, $\{d_{ij}\}_{ij}$ denotes integer constants, and $\{[X_i]_T\}_i$ denotes pairings between elements in the verification key and the signing key. Here, elements in $\{x_{ij}\}_{ij}$ may represent the same variables, and the same argument is made for $\{[X_i]_T\}_i$.

[12] In the worst case, the resulting signing key size is the total number of elements in all $\{[A_i]_b\}_i$.

We now show how to obtain PPEs that check the correctness of signing keys with respect to verification keys as follows. Let $\mathcal{E}$ be the set of all the *distinct* variables in $\{x_{ij}\}_{ij}$ (not including constants). Then for any $x \in \mathcal{E}$, we rewrite Equation (2) as

$$\sum_i x^{d_i}[Y_i]_T = [0]_T, \tag{3}$$

where $\{d_i\}_i$ denotes integer constants and $\{[Y_i]_T\}_i$ denotes elements in $\mathbb{G}_T$. Since the SPS scheme is well-formed, the left hand side of Equation (3) can be treated as a polynomial in x by fixing all $[Y_i]_T$. We rewrite Equation (3) as

$$[P_0]_T + x^1[P_1]_T + \ldots + x^n[P_n]_T = [0]_T, \tag{4}$$

for some fixed polynomial n, where $[P_k]_T$ denotes the sum of coefficients of x^k. According to the definition of well-formed SPSs, since the space of x is super-polynomial (in the security parameter) and n is a polynomial (in the security parameter), the number of possible values of x must be larger than n for sufficiently large security parameters. As a result, if Equation (4) holds for all possible value of x, we have

$$[P_0]_T = [0]_T,\ [P_1]_T = [0]_T,\ \ \ldots,\ [P_n]_T = [0]_T, \tag{5}$$

or the number of roots of Equation (4) could be larger than n, which is against Schwartz-Zippel Lemma. On the other hand, it is obvious that if PPEs in (5) hold, Equation (4) holds for any x. For each $[P_i]_T = 0$, we cancel another variable in $\mathcal{E}$ in the same way. Recursively, all the variables in PPEs in the verification algorithm can be cancelled, and we finally obtain a sequence of PPEs of the form

$$\sum_i c'_i[X'_i]_T = [0]_T,$$

where $\{c'_i\}_i$ denotes integer constants, and $\{[X'_i]_T\}_i$ denotes pairings between elements in the verification key and the signing key, and elements in $\{[X'_i]_T\}_i$ may represent the same variables. Since such collection of PPEs hold *if and only if* PPEs in the verification algorithm holds for all possible randomness and messages, we obtain an algorithm that takes as input verification/signing key pairs and check their correctness using this collection of PPEs.[13]

In conclusion, any well-formed SPS scheme can be converted into an SKSP-TS scheme, completing the proof of Theorem 2. □

Remark. It is not hard to see that the latter half of the proof can also be adopted to show that for a well-formed SPS scheme, if signing keys consist only of group elements, then it is an FSPS scheme.

[13] The number of PPEs we finally obtain is smaller than number of elements in $\{[X_i]_T\}_i$ in PPEs of the form Equation (2).

3.4 Instantiations of Trapdoor Signature

UF-CMA secure TS scheme. Using the conversion described in the proof of Theorem 2, we can convert well-formed SPSs into SKSP-TSs. For ease of understanding, we give an instantiation of γ-TS $\Sigma = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign})$ in Fig. 1, which is converted from the SPS scheme (denoted by $\mathcal{T}_\Sigma = (\mathsf{Setup}, \mathcal{T}_{\mathsf{Gen}}, \mathsf{TDSign}, \mathsf{Verify})$) proposed by Kiltz et al. [31]. Here, $\mathcal{T}_\Sigma$ is UF-CMA secure under the $\mathcal{D}_k$-MDDH assumptions and $\gamma : \mathbb{Z}_p^n \mapsto \mathbb{G}_1^n$ is defined by $\gamma(x_1, \ldots, x_n) = ([x_1]_1, \ldots, [x_n]_1)$, where n denotes the number of group elements in a message.

To generate a signature of $\mathcal{T}_\Sigma$, $\sigma_1 = [(1, \vec{m}^\top)]_1 \mathbf{K} + \vec{r}^\top [\mathbf{P}_0 + \tau \mathbf{P}_1]_1$ is the only part that needs to be operated by using "$\mathbb{Z}_p$-elements" $\mathbf{K} \in \mathbb{Z}_p^{(n+1)\times(k+1)}$ of the signing key. Following our conversion, we replace $\mathbf{K}$ with $[\mathbf{K}]_1$ in the signing key, and keep the original signing key as the trapdoor key. By using $[\mathbf{K}]_1$, we can compute σ_1 as $(1, \vec{m}^\top)[\mathbf{K}]_1 + \vec{r}^\top [\mathbf{P}_0 + \tau \mathbf{P}_1]_1$. Furthermore, we obtain PPEs that check the correctness of signing keys as follows.

$$
\begin{aligned}
& e(\sigma_1, [\mathbf{A}]_2) = e([(1, \vec{m}^\top)]_1, [\mathbf{C}]_2) + e(\sigma_2, [\mathbf{C}_0]_2) + e(\sigma_3, [\mathbf{C}_1]_2), \\
\Rightarrow & e((1, \vec{m}^\top)[\mathbf{K}]_1 + \vec{r}^\top [\mathbf{P}_0 + \tau \mathbf{P}_1]_1, [\mathbf{A}]_2) = e([(1, \vec{m}^\top)]_1, [\mathbf{C}]_2) + e(\vec{r}^\top [\mathbf{B}^\top]_1, [\mathbf{C}_0]_2) \\
& \qquad + e(\vec{r}^\top [\mathbf{B}^\top \tau]_1, [\mathbf{C}_1]_2), \qquad \text{(Rewrite first equation in } \mathsf{Verify}) \\
\Rightarrow & \begin{cases} e((1, \vec{m}^\top)[\mathbf{K}]_1 + \vec{r}^\top [\mathbf{P}_0]_1, [\mathbf{A}]_2) = e([(1, \vec{m}^\top)]_1, [\mathbf{C}]_2) + e(\vec{r}^\top [\mathbf{B}^\top]_1, [\mathbf{C}_0]_2), \\ e(\vec{r}^\top [\mathbf{P}_1]_1, [\mathbf{A}]_2) = e(\vec{r}^\top [\mathbf{B}^\top]_1, [\mathbf{C}_1]_2), \end{cases} \qquad \text{(Cancelling } \tau) \\
\Rightarrow & \begin{cases} e((1, \vec{m}^\top)[\mathbf{K}]_1, [\mathbf{A}]_2) = e([(1, \vec{m}^\top)]_1, [\mathbf{C}]_2), \\ e([\mathbf{P}_0]_1, [\mathbf{A}]_2) = e([\mathbf{B}^\top]_1, [\mathbf{C}_0]_2), \\ e([\mathbf{P}_1]_1, [\mathbf{A}]_2) = e([\mathbf{B}^\top]_1, [\mathbf{C}_1]_2), \end{cases} \qquad \text{(Cancelling } \vec{r}) \\
\overset{*}{\Rightarrow} & \begin{cases} e([\mathbf{K}]_1, [\mathbf{A}]_2) = e([1]_1, [\mathbf{C}]_2)), \\ e([\mathbf{P}_0]_1, [\mathbf{A}]_2) = e([\mathbf{B}^\top]_1, [\mathbf{C}_0]_2), \\ e([\mathbf{P}_1]_1, [\mathbf{A}]_2) = e([\mathbf{B}^\top]_1, [\mathbf{C}_1]_2). \end{cases} \qquad \text{(Cancelling } m)
\end{aligned}
$$

Then we rewrite the second equation $e(\sigma_2, \sigma_4) = e(\sigma_3, [1]_2)$ as $e(\vec{r}^\top [\mathbf{B}^\top]_1, [\tau]_2) = e(\vec{r}^\top [\mathbf{B}^\top \tau]_1, [1]_2)$. By cancelling $\vec{r}$ and τ, we obtain $e([\mathbf{B}^\top]_1, [1]_2) = e([\mathbf{B}^\top]_1, [1]_2)$, which is trivial.[14]

Finally, we obtain the algorithm $\mathsf{VerifySK}$ checking correctness of signing keys with respect to verification keys via the above three PPEs (derived from $\overset{*}{\Rightarrow}$).

Theorem 3 *The instantiation described in Fig. 1 is a UF-CMA secure γ-SKSP-TS scheme under the $\mathcal{D}_k$-MDDH assumptions.*

The SKSP property of this instantiation is implied by Theorem 2 and the UF-CMA security is implied by Theorem 1. We refer the reader to the full paper for the proof of Theorem 3.

[14] Note that for simplicity, we sometimes directly canceled vectors in the above conversion, instead of following the proof of Theorem 2 to cancel elements one by one.

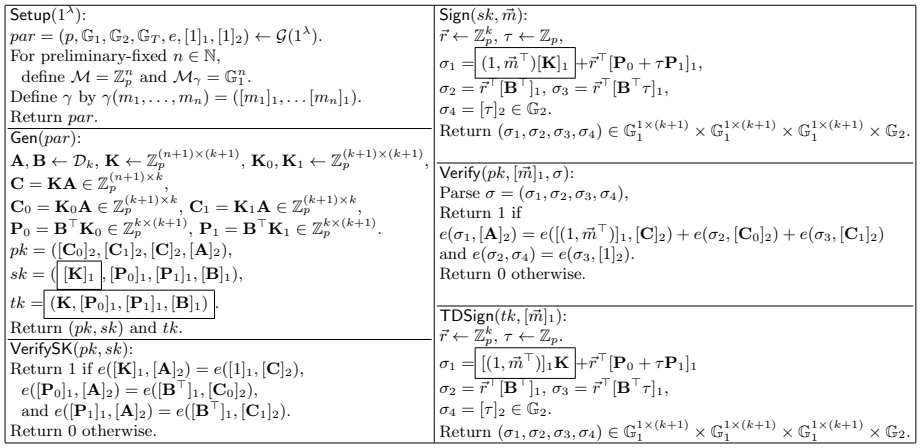

Fig. 1. A UF-CMA secure γ-TS scheme adapted from [31, Section 4.2]. The boxes indicate the main differences from the original scheme in [31].

UF-otRMA secure TS scheme. In Fig. 2, we give another instantiation of TS which satisfies the UF-otRMA security under the $\mathcal{D}_k$-MDDH assumptions. This scheme is converted from the UF-otRMA secure SPS scheme in [31]. The proof of correctness is straightforward and the correctness of a signing key with respect to a verification key can be verified by $\mathsf{VerifySK}$ via $e([\mathbf{K}]_1, [\mathbf{A}]_2) = e([1]_1, [\mathbf{C}]_2)$.

Unlike the UF-CMA security proved in Theorem 1, the UF-otRMA security of $\Sigma = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign})$ is not automatically implied by the UF-otRMA security of $\mathcal{T}_\Sigma = (\mathsf{Setup}, \mathcal{T}_{\mathsf{Gen}}, \mathsf{TDSign}, \mathsf{Verify})$. However, according to [31], the proof of the UF-otRMA security of $\mathcal{T}_\Sigma$ remains valid even when an adversary sees the exponents of the messages from the signing oracle, which implies the UF-otRMA security of Σ. We refer the reader to [31] for details of the proof.

4 (Two-tier) Signatures with Auxiliary Key(s)

In this section, we introduce AKSs which are used as building blocks to achieve our generic construction of FSPS. In Section 4.1, we give the definition of AKSs, define their properties, and give an instantiation of AKS. In Section 4.2, we extend AKS to TT-AKS and give an instantiation of TT-AKS.

4.1 Signature with Auxiliary Key

Definition. Roughly speaking, a γ-AKS scheme is a signature scheme in which the key generation algorithm additionally generates auxiliary keys, and the verification key space and the auxiliary key space have a special (but natural) structure related with γ.

$\mathsf{Setup}(1^\lambda)$: $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathcal{G}(1^\lambda)$. $\mathcal{M} = \mathbb{Z}_p^n$, $\mathcal{M}_\gamma = \mathbb{G}_1^n$. For preliminary-fixed $n \in \mathbb{N}$, define γ by $\gamma(m_1, \ldots, m_n) = ([m_1]_1, \ldots [m_n]_1)$. Return par.	$\mathsf{Sign}(sk, \vec{m})$: $\sigma = \boxed{(1, \vec{m}^\top)[\mathbf{K}]_1}$. Return $\sigma \in \mathbb{G}_1^{1\times k}$.
$\mathsf{Gen}(par)$: $\mathbf{A} \leftarrow \mathcal{D}_k$, $\mathbf{K} \leftarrow \mathbb{Z}_p^{(n+1)\times k}$, $\mathbf{C} = \mathbf{K}\overline{\mathbf{A}} \in \mathbb{Z}_p^{(n+1)\times k}$, $pk = ([\mathbf{C}]_2, [\overline{\mathbf{A}}]_2)$, $sk = \boxed{[\mathbf{K}]_1}$, $tk = \boxed{\mathbf{K}}$. Return (pk, sk) and tk.	$\mathsf{Verify}(pk, [\vec{m}]_1, \sigma)$: Return 1 if $e(\sigma, [\overline{\mathbf{A}}]_2) = e([(1, \vec{m}^\top)]_1, [\mathbf{C}]_2)$. Return 0 otherwise.
$\mathsf{VerifySK}(pk, sk)$: Return 1 if $e([\mathbf{K}]_1, [\overline{\mathbf{A}}]_2) = e([1]_1, [\mathbf{C}]_2)$. Return 0 otherwise.	$\mathsf{TDSign}(tk, [\vec{m}]_1)$: $\sigma = \boxed{[(1, \vec{m}^\top)]_1\mathbf{K}}$. Return $\sigma \in \mathbb{G}_1^{1\times k}$.

Fig. 2. A UF-otRMA secure γ-SKSP-TS scheme adapted from [31, Section 5.2]. The boxes indicate the main differences from the original scheme in [31].

Definition 11 (γ-signature with auxiliary key (γ-AKS)) *A signature scheme $\Sigma = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ with verification key space $\mathcal{P}_\gamma$ is said to be a γ-AKS scheme for an efficiently computable bijection $\gamma : \mathcal{P} \mapsto \mathcal{P}_\gamma$ if in addition to the verification/signing key pair (pk, sk), Gen also outputs an auxiliary key $ak \in \mathcal{P}$ such that $pk = \gamma(ak)$.*

Security. The UF-(ot)CMA security and UF-(ot)RMA security of γ-AKSs are exactly the same as those of standard signatures except that Gen addtionally generates ak.

Key generation algorithm $\mathcal{U}_{\mathsf{Gen}}$. Similarly to $\mathcal{T}_{\mathsf{Gen}}$ defined in Section 3.1, we use $\mathcal{U}_{\mathsf{Gen}}$ to denote an algorithm that runs Gen, which is the key generation algorithm of a γ-AKS scheme, in the following way.

Taking as input a public parameter par, $\mathcal{U}_{\mathsf{Gen}}$ gives par to Gen and obtains an output $((pk, sk), ak)$. Then $\mathcal{U}_{\mathsf{Gen}}$ outputs (pk, sk) as a verification/signing key pair, without outputting ak. We use $\mathcal{U}_\Sigma$ to denote $(\mathsf{Setup}, \mathcal{U}_{\mathsf{Gen}}, \mathsf{Sign}, \mathsf{Verify})$ when $\Sigma = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$.

Just like SPSs, we consider γ-AKSs with the SP property.

Definition 12 (γ-SP-AKS) *A γ-AKS scheme Σ is said to be a* γ-SP-AKS scheme *if $\mathcal{U}_\Sigma$ is an SPS scheme.*

Converting SPSs into SP-AKSs. It is straightforward to see that any SPS scheme with an algebraic key generation algorithm, public keys of which are supposed to be of the form $([\vec{u}]_1, [\vec{v}]_2)$, can be converted into a γ-SP-AKS scheme, where γ is defined by $\gamma(\vec{u}, \vec{v}) = ([\vec{u}]_1, [\vec{v}]_2)$, since we can force the setup of any SPS to output no common parameter except for the bilinear map description and let the key generation algorithm additionally output $(\vec{u}_1, \vec{v}_2)$.

We now define the random auxiliary key property for AKSs.

Definition 13 (Random auxiliary key property) *A γ-AKS scheme* $(\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *with an auxiliary key space $\mathcal{P}$ is said to satisfy the* random auxiliary key property *if there exists an additional algorithm* AKGen *such that* AKGen *takes as input par and an auxiliary key ak, and outputs a verification/signing key pair (pk, sk) where $\gamma(ak) = pk$. Furthermore, for any PPT adversary $\mathcal{A}$ and all $\lambda \in \mathbb{N}$, we have*

$$|\Pr[par \leftarrow \mathsf{Setup}(1^\lambda) : \mathcal{A}^{\mathrm{GenO}}(par) = 1] - \Pr[par \leftarrow \mathsf{Setup}(1^\lambda) : \mathcal{A}^{\mathrm{AKGenO}}(par) = 1]| \leq negl(\lambda),$$

where GenO *runs* $((pk, sk), ak) \leftarrow \mathsf{Gen}(par)$, *and returns* (pk, sk, ak), *and* AKGenO *uniformly chooses ak from $\mathcal{P}$, runs* $(pk, sk) \leftarrow \mathsf{AKGen}(par, ak)$, *and returns* (pk, sk, ak).

Instantiation of AKS. Now we give an instantiation of AKS satisfying UF-otCMA security under the $\mathcal{D}_k$-MDDH assumptions in Fig. 3. This signature scheme is actually the same as the UF-otCMA secure signature scheme in [31] except that Gen additionally generates exponents of a verification key as an auxiliary key. For this instantiation, the bijection γ is defined by $\gamma(\mathbf{X}) = [\mathbf{X}]_2 \in \mathbb{G}_2^{(n+1)\times k} \times \mathbb{G}_2^{(k+1)\times k}$ for n which denotes the length of a message.

We refer the reader to [31] for the proof of the UF-otCMA security of this instantiation.

$\mathsf{Setup}(1^\lambda)$: $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathcal{G}(1^\lambda)$. For preliminary-fixed $n \in \mathbb{N}$, define $\mathcal{M} = \mathbb{Z}_p^n$, $\mathcal{M}_\gamma = \mathbb{G}_1^n$, $\mathcal{P} = \mathbb{Z}_p^{(n+1)\times k} \times \mathbb{Z}_p^{(k+1)\times k}$, and $\mathcal{P}_\lambda = \mathbb{G}_2^{(n+1)\times k} \times \mathbb{G}_2^{(k+1)\times k}$. Define γ by $\gamma(\mathbf{X}) = [\mathbf{X}]_2 \in \mathbb{G}_2^{(n+1)\times k} \times \mathbb{G}_2^{(k+1)\times k}$. Return par.	$\mathsf{AKGen}(par, ak)$: Parse $ak = (\mathbf{C}, \mathbf{A})$. Let $\mathbf{A} = \begin{pmatrix} \overline{\mathbf{A}} \\ \vec{a}^\top \end{pmatrix}$, $\vec{k} \leftarrow \mathbb{Z}_p^{n+1}$, $\underline{\mathbf{K}} = (\mathbf{C} - \vec{k}\vec{a}^\top)\overline{\mathbf{A}}^{-1}$, $\mathbf{K} = (\underline{\mathbf{K}}, \vec{k})$. $pk = ([\mathbf{C}]_2, [\mathbf{A}]_2)$, $sk = \mathbf{K}$, $ak = (\mathbf{C}, \mathbf{A})$. Return (pk, sk) and ak.
$\mathsf{Gen}(par)$: $\mathbf{A} \leftarrow \mathcal{D}_k$, $\mathbf{K} \leftarrow \mathbb{Z}_p^{(n+1)\times(k+1)}$, $\mathbf{C} = \mathbf{K}\mathbf{A} \in \mathbb{Z}_p^{(n+1)\times k}$. $pk = ([\mathbf{C}]_2, [\mathbf{A}]_2)$, $sk = \mathbf{K}$, and $ak = (\mathbf{C}, \mathbf{A})$. Return (pk, sk) and ak.	$\mathsf{Sign}(sk, [\vec{m}]_1)$: $\sigma = [(1, \vec{m}^\top)]_1 \mathbf{K} \in \mathbb{G}_1^{1\times(k+1)}$. $\mathsf{Verify}(pk, [\vec{m}]_1, \sigma)$: Return 1 if $e(\sigma, [\mathbf{A}]_2) = e([(1, \vec{m}^\top)]_1, [\mathbf{C}]_2)$. Return 0 otherwise.

Fig. 3. A UF-otCMA secure γ-SP-AKS scheme adapted from [31, Section 3].

Theorem 4 *The instantiation described in Fig. 3 satisfies the random auxiliary key property.*

This proof follows from the fact that when the distribution of $\mathbf{C}$ is uniform, the distribution of $\underline{\mathbf{K}} = (\mathbf{C} - \vec{k}\vec{a}^\top)\overline{\mathbf{A}}^{-1}$ is uniform as well. We give the proof of Theorem 4 in the full paper due to page limitation.

4.2 Two-tier Signature with Auxiliary Keys

Definition. Besides AKS, we also give the definition of (γ_p, γ_s)-TT-AKSs, which is the same as that of two-tier signatures [12,1,32] except that the key generation algorithms additionally generate primary/secondary auxiliary keys. The primary/secondary verification key space and the primary/secondary auxiliary key space have a special (but natural) structure related with γ_p/γ_s. Combining SP-TT-AKSs with SKSP-TSs enables us to obtain more efficient instantiations of FSPS and FAS.

Definition 14 ((γ_p, γ_s)-TT-AKS) *A (γ_p, γ_s)-TT-AKS scheme consists of five polynomial-time algorithms* Setup, PGen, SGen, TTSign, *and* $\mathsf{TTVerify}$. Setup *is a randomized algorithm that takes as input 1^λ, and outputs a public parameter par, which determines the message space $\mathcal{M}$, the primary/secondary verification key spaces $\mathcal{P}_\gamma/\mathcal{S}_\gamma$, the primary/secondary auxiliary key spaces $\mathcal{P}/\mathcal{S}$, and the efficiently computable bijections $\gamma_p : \mathcal{P} \mapsto \mathcal{P}_\gamma$ and $\gamma_s : \mathcal{S} \mapsto \mathcal{S}_\gamma$.* PGen *is a randomized algorithm that takes as input par, and outputs a primary verification/signing key pair (Ppk, Psk) where $Ppk \in \mathcal{P}_\gamma$ and a primary auxiliary key $Pak \in \mathcal{P}$.* SGen *is a randomized algorithm that takes as input a primary verification/signing key pair (Ppk, Psk) and a primary auxiliary key Pak, and outputs a secondary verification/signing key pair (opk, osk) where $opk \in \mathcal{S}_\gamma$ and a secondary auxiliary key $oak \in \mathcal{S}$.* TTSign *is a randomized algorithm that takes as input a primary signing key Psk, a secondary signing key osk, and a message m, and returns a signature σ.* $\mathsf{TTVerify}$ *is a deterministic algorithm that takes as input a primary verification key Ppk, a secondary verification key opk, a message m, and a signature σ, and returns 1 (accept) or 0 (reject).*

The correctness is satisfied if for all $\lambda \in \mathbb{N}$, $par \leftarrow \mathsf{Setup}(1^\lambda)$, $((Ppk, Psk), Pak) \leftarrow \mathsf{PGen}(par)$, and $((opk, osk), oak) \leftarrow \mathsf{SGen}(Ppk, Psk, Pak)$, we have (a) $\mathsf{TTVerify}(Ppk, opk, m, \mathsf{TTSign}(Psk, osk, m)) = 1$ for all messages $m \in \mathcal{M}$, and (b) $\gamma_p(Pak) = Ppk$ and $\gamma_s(oak) = opk$.

Unlike the definition of standard two-tier signatures, SGen takes as input (Ppk, Psk, Pak) (instead of (Ppk, Psk)) in the above definition. However, the interface of SGen is not essentially changed since Pak can be treated as part of Psk.

Security. Now we give the definition of unforgeability against two-tier chosen message attacks (UF-TT-CMA).

Definition 15 (UF-TT-CMA) *A TT-AKS scheme* $(\mathsf{PGen}, \mathsf{SGen}, \mathsf{TTSign}, \mathsf{TTVerify})$ *is said to be unforgeable against two-tier chosen message attacks if for any PPT adversary $\mathcal{A}$, we have*

$$\Pr[par \leftarrow \mathsf{Setup}(1^\lambda), ((Ppk, Psk), Pak) \leftarrow \mathsf{PGen}(par), (i^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathrm{TTSignO}(\cdot)}(Ppk) : (i^*, m) \in \mathcal{TQ}_m \wedge m^* \neq m \wedge \mathsf{TTVerify}(Ppk, opk_{i^*}, m^*, \sigma^*) = 1] \leq negl(\lambda),$$

where TTSignO$(\cdot)$ *is the signing oracle that takes a message* $m \in \mathcal{M}$ *as input, runs* $i = i + 1$ *(initialized with* 0*), samples* $(opk_i, osk_i) \leftarrow \mathsf{SGen}(Ppk, Psk, Pak)$*, and computes* $\sigma \leftarrow \mathsf{TTSign}(Psk, osk_i, m)$*. Then it adds* (i, m) *to* $\mathcal{TQ}_m$ *(initialized with* $\emptyset$*) and returns* (opk_i, σ)*.*

Next we define the SP property of TT-AKS as follows.

Definition 16 (Structure-preserving TT-AKS (SP-TT-AKS)) *A TT-AKS scheme is said to be* structure-preserving *over a bilinear group generator* $\mathcal{G}$ *if we have (a) a public parameter includes a group description* gk *generated by* $\mathcal{G}$*, (b) primary and secondary verification keys consist of group elements in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*, (c) messages consist of group elements in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*, and (d) the verification algorithm consists only of evaluating membership in* $\mathbb{G}_1$ *and* $\mathbb{G}_2$ *and relations described by PPEs.*

Converting SP two-tier signatures into SP-TT-AKSs. Like SP-AKSs, SP-TT schemes, primary and secondary verification keys of which are supposed to be of the form $([\vec{u}]_1, [\vec{v}]_2)$ and $([\vec{u}']_1, [\vec{v}']_2)$ respectively, can be converted into a (γ_p, γ_s)-SP-TT-AKS scheme, where γ_p and γ_s are defined as $\gamma_p(\vec{u}, \vec{v}) = ([\vec{u}]_1, [\vec{v}]_2)$ and $\gamma_s(\vec{u}', \vec{v}') = ([\vec{u}']_1, [\vec{v}']_2)$ respectively, as long as the key generation algorithms are algebraic and primary signing keys consist only of elements in $\mathbb{Z}_p$.[15]

We define the random primary and secondary auxiliary key properties of TT-AKSs as follows.

Definition 17 (Random primary/secondary auxiliary key properties) *A* (γ_p, γ_s)*-TT-AKS scheme* $(\mathsf{Setup}, \mathsf{PGen}, \mathsf{SGen}, \mathsf{TTSign}, \mathsf{TTVerify})$ *is said to satisfy the* random primary auxiliary key property *if there exists an additional polynomial-time algorithm* AKPGen *that takes as input par and a primary auxiliary key* Pak*, and outputs a primary verification/signing key pair* (Ppk, Psk) *where* $\gamma_p(Pak) = Ppk$*. Furthermore, for any PPT adversary* $\mathcal{A}$ *and all* $\lambda \in \mathbb{N}$*, we have*

$$|\Pr[par \leftarrow \mathsf{Setup}(1^\lambda) : \mathcal{A}^{\mathrm{PGenO}}(par) = 1] - \Pr[par \leftarrow \mathsf{Setup}(1^\lambda) : \mathcal{A}^{\mathrm{AKPGenO}}(par) = 1]| \leq negl(\lambda),$$

where PGenO *runs* $((Ppk, Psk), Pak) \leftarrow \mathsf{PGen}(par)$ *and returns* $((Ppk, Psk), Pak)$*, and* AKPGenO *uniformly chooses* Pak *from the primary auxiliary key space* $\mathcal{P}$*, runs* $(Ppk, Psk) \leftarrow \mathsf{AKPGen}(par, Pak)$*, and returns* $((Ppk, Psk), Pak)$*.*

Furthermore, it is said to satisfy the random secondary auxiliary key property *if there exists another polynomial-time algorithm* AKSGen *that takes as input a primary verification/signing key pair* (Ppk, Psk)*, a primary auxiliary key* Pak*, and a secondary auxiliary key* oak*, and outputs a secondary verification/signing*

[15] If a primary signing key consists of group elements, PGen may have trouble in outputting secondary auxiliary keys. However, this can be easily solved by forcing PGen to output the exponents of those group elements as part of a primary signing key.

key pair (opk, osk) *where* $\gamma_s(oak) = opk$*. Furthermore, for any PPT adversary* $\mathcal{A}$ *and all* $\lambda \in \mathbb{N}$*, we have*

$$|\Pr[par \leftarrow \mathsf{Setup}(1^\lambda) : \mathcal{A}^{\mathrm{SGenO}(\cdot)}(par) = 1] - \Pr[par \leftarrow \mathsf{Setup}(1^\lambda) : \mathcal{A}^{\mathrm{AKSGenO}(\cdot)}(par) = 1]| \leq negl(\lambda),$$

Here, on input a polynomial $n = n(\lambda)$*,* $\mathrm{SGenO}(\cdot)$ *runs* $((Ppk, Psk), Pak) \leftarrow \mathsf{PGen}(par)$ *and* $((opk_i, osk_i), oak_i) \leftarrow \mathsf{SGen}(Ppk, Psk, Pak)$ *for* $i = 1, \ldots, n$*, and returns* $(Ppk, Psk, Pak, \{(opk_i, osk_i, oak_i)\}_{i=1}^n)$*. On input a polynomial* $n = n(\lambda)$*,* $\mathrm{AKSGenO}(\cdot)$ *runs* $((Ppk, Psk), Pak) \leftarrow \mathsf{PGen}(par)$*, uniformly chooses* oak_i *from the secondary auxiliary key space* $\mathcal{S}$*, runs* $(opk_i, osk_i) \leftarrow \mathsf{AKSGen}(Ppk, Psk, Pak, oak_i)$ *for* $i = 1, \ldots, n$*, and returns* $(Ppk, Psk, Pak, \{(opk_i, osk_i, oak_i)\}_{i=1}^n)$*.*

Instantiation of (γ_p, γ_s)*-SP-TT-AKS.* Now we give an instantiation of (γ_p, γ_s)-SP-TT-AKS satisfying UF-TT-CMA security under the $\mathcal{D}_k$-MDDH assumptions. This signature scheme is the same as the SP two-tier signature scheme in [32] except that PGen and SGen additionally generate the auxiliary keys, and SGen addtionally takes as input the primary auxiliary key. For this instantiation, the bijections (γ_p, γ_s) are defined by $\gamma_p(\mathbf{X}) = [\mathbf{X}]_2 \in \mathbb{G}_2^{n\times k} \times \mathbb{G}_2^{(k+1)\times k}$ and $\gamma_s(\vec{x}) = [\vec{x}]_2 \in \mathbb{G}_2^{1\times k}$ respectively for some fixed integer n which denotes the length of a message.

$\mathsf{Setup}(1^\lambda)$:
$par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathcal{G}(1^\lambda)$.
For preliminary-fixed $n \in \mathbb{N}$,
define $\mathcal{M} = \mathbb{Z}_p^n$, $\mathcal{M}_\gamma = \mathbb{G}_1^n$,
$\mathcal{P} = \mathbb{Z}_p^{n\times k} \times \mathbb{Z}_p^{(k+1)\times k}$, $\mathcal{P}_\gamma = \mathbb{G}_2^{n\times k} \times \mathbb{G}_2^{(k+1)\times k}$,
$\mathcal{S} = \mathbb{Z}_p^{1\times k}$, and $\mathcal{S}_\gamma = \mathbb{G}_2^{1\times k}$.
Define γ_p by $\gamma_p(\mathbf{X}) = [\mathbf{X}]_2 \in \mathbb{G}_2^{n\times k} \times \mathbb{G}_2^{(k+1)\times k}$ and γ_s by $\gamma_s(\vec{x}) = [\vec{x}]_2 \in \mathbb{G}_2^{1\times k}$.
Return par.

$\mathsf{PGen}(par)$:
$\mathbf{A} \leftarrow \mathcal{D}_k$, $\mathbf{K}' \leftarrow \mathbb{Z}_p^{n\times(k+1)}$, $\mathbf{C}' = \mathbf{K}'\mathbf{A} \in \mathbb{Z}_p^{n\times k}$.
$Ppk = ([\mathbf{C}']_2, [\mathbf{A}]_2))$, $Psk = \mathbf{K}'$, $Pak = (\mathbf{C}', \mathbf{A})$.
Return (Ppk, Psk) and Pak.

$\mathsf{SGen}(Ppk, Psk, Pak)$:
$\vec{k} \leftarrow \mathbb{Z}_p^{k+1}$, $\vec{c} = \vec{k}^\top \mathbf{A} \in \mathbb{Z}_p^{1\times k}$.
$opk = [\vec{c}]_2$, $osk = \vec{k}$, $oak = \vec{c}$.
Return (opk, osk) and oak.

$\mathsf{AKPGen}(par, Pak)$:
Parse $Pak = (\mathbf{C}', \mathbf{A})$.
Let $\mathbf{A} = \begin{pmatrix} \underline{\mathbf{A}} \\ \vec{a}^\top \end{pmatrix}$,
$\vec{k}' \leftarrow \mathbb{Z}_p^n$, $\underline{\mathbf{K}}' = (\mathbf{C}' - \vec{k}'\vec{a}^\top)\underline{\mathbf{A}}^{-1} \in \mathbb{Z}_p^{n\times k}$,
$\mathbf{K}' = (\underline{\mathbf{K}}', \vec{k}') \in \mathbb{Z}_p^{n\times(k+1)}$.
$Ppk = ([\mathbf{C}']_2, [\mathbf{A}]_2)$, $Psk = \mathbf{K}'$, $Pak = (\mathbf{C}', \mathbf{A})$.
Return (Ppk, Psk) and Pak.

$\mathsf{AKSGen}(Ppk, Psk, Pak, oak)$:
Parse $Ppk = ([\mathbf{C}']_2, [\mathbf{A}]_2))$, $Psk = \mathbf{K}'$, $Pak = (\mathbf{C}', \mathbf{A})$, and $oak = \vec{c}$.
Let $\mathbf{A} = \begin{pmatrix} \underline{\mathbf{A}} \\ \vec{a}^\top \end{pmatrix}$, $k \leftarrow \mathbb{Z}_p$, $\vec{k}'^\top = (\vec{c} - k\vec{a}^\top)\underline{\mathbf{A}}^{-1}$, $\vec{k}^\top = (\vec{k}'^\top, k)$.
$opk = [\vec{c}]_2$, $osk = \vec{k}$, $oak = \vec{c}$.
Return (opk, osk) and oak.

$\mathsf{TTSign}(Psk, osk, [\vec{m}]_1)$:
$\mathbf{K} = (\vec{k}, \mathbf{K}'^\top)^\top$.
Return $\sigma = [(1, \vec{m}^\top)]_1\mathbf{K} \in \mathbb{G}_1^{1\times(k+1)}$.

$\mathsf{TTVerify}(Ppk, opk, [\vec{m}]_1, \sigma)$:
$[\mathbf{C}]_2 = ([\vec{c}]_2^\top, [\mathbf{C}']_2^\top)^\top$.
Return 1 if $e(\sigma, [\mathbf{A}]_2) = e([(1, \vec{m}^\top)]_1, [\mathbf{C}]_2)$.
Return 0 otherwise.

Fig. 4. A UF-TT-CMA secure (γ_p, γ_s)-SP-TT-AKS scheme adapted from [32, Section 6.1].

Theorem 5 *The instantiation described in Fig. 4 satisfies the random primary and secondary auxiliary key properties.*

This proof follows from the fact that when the distributions of $\mathbf{C}'$ and c are uniform, the distribution of $\underline{\mathbf{K}'} = (\mathbf{C}' - \vec{k}\vec{a}^\top)\overline{\mathbf{A}}^{-1}$ and $\vec{k}' = (\vec{c} - k\vec{a}^\top)\overline{\mathbf{A}}^{-1}$ are uniform as well. We give the proof of Theorem 5 in the full paper due to page limitation.

5 Generic Construction of Fully Structure-Preserving Signatures (and Fully Automorphic Signatures)

In this section, we give generic constructions of FSPSs and FASs from SKSP-TSs and (TT-)AKSs. Such constructions can be derived from SPSs that are based on various assumptions and with different efficiency performance. In Section 5.1, Section 5.2, and Section 5.3, we give three generic constructions of UF-CMA FSPS schemes respectively. The first two constructions are based on SKSP-TSs and SP-AKSs, and the third one is based on SKSP-TSs and SP-TT-AKSs.

5.1 Generic Construction Sig_1: Trapdoor Signature + Signature with Auxiliary Key

We give a generic construction of FSPS (and FASs) based on a γ-SKSP-TS scheme and a γ'-SP-AKS scheme, where γ and γ' satisfy a suitable compatibility that we explain shortly.

Let $\Sigma_t = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign}, \mathsf{VerifySK})$ be a γ-SKSP-TS scheme with message spaces $\mathcal{M}$ and $\mathcal{M}_\gamma$, and $\Sigma_s = (\mathsf{Setup}, \mathsf{Gen}', \mathsf{Sign}', \mathsf{Verify}')$[16] a γ'-SP-AKS scheme with verification key space $\mathcal{M}_\gamma$, auxiliary key space $\mathcal{M}$, and message space $\mathcal{M}'$, and we have $\gamma'(x) = \gamma(x)$. Then the generic construction of FSPS denoted by $\mathsf{Sig}_1 = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ with message space $\mathcal{M}'$ is described as in Fig. 5.

Next we give a theorem for this generic construction.

Theorem 6 *If Σ_t is a UF-CMA secure SKSP-TS scheme, and Σ_s a UF-otCMA secure SP-AKS scheme, then $\mathsf{Sig}_1 = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ is a UF-CMA secure FSPS scheme.*

Proof sketch. The proof of Theorem 6 follows from the fact that if there exists a PPT adversary $\mathcal{A}$ that outputs a successful forgery $(\sigma_1^*, \sigma_2^*, \sigma_3^*)$, where σ_2^* was not queried before (respectively, was queried before), with non-negligible probability, then we can construct a PPT adversary $\mathcal{B}_1$ (respectively, $\mathcal{B}_2$) that breaks the UF-CMA security of Σ_t (respectively, the UF-otCMA security of Σ_s). Note that to answer a query from $\mathcal{A}$, $\mathcal{B}_2$ may have to use the signing key of Σ_t to sign an auxiliary key ak' of Σ_s, while it only learns the corresponding verification key pk' from the challenger. In this case, it signs pk' by using the trapdoor key of Σ_t instead. According to the correctness of a TS scheme, $\mathcal{A}$ cannot distinguish such a signature with an honestly generated one, which means that $\mathcal{B}_2$ can perfectly simulate the signing oracle of $\mathcal{A}$. We refer the reader to the full paper for the proof.

[16] As in [5], we assume that Σ_t and Σ_s share the common setup algorithm Setup.

$\widehat{\mathsf{Setup}}(1^\lambda)$: Run $par \leftarrow \mathsf{Setup}(1^\lambda)$. Determine the message spaces $\mathcal{M}$ and $\mathcal{M}_\gamma$ for Σ_t. Define $\gamma : \mathcal{M} \mapsto \mathcal{M}_\gamma$. Determine the message space $\mathcal{M}'$, verification key space $\mathcal{M}_\gamma$, and auxiliary key space $\mathcal{M}$ for Σ_s. Define $\gamma' : \mathcal{M} \mapsto \mathcal{M}_\gamma$ where $\gamma'(x) = \gamma(x)$. Return par.	$\widehat{\mathsf{Sign}}(sk, M)$: $((pk', sk'), ak') \leftarrow \mathsf{Gen}'(par)$. $\sigma_1 \leftarrow \mathsf{Sign}(sk, ak')$. $\sigma_2 = pk'$. $\sigma_3 \leftarrow \mathsf{Sign}'(sk', M)$. Return $\sigma = (\sigma_1, \sigma_2, \sigma_3)$.
$\widehat{\mathsf{Gen}}(par)$: $((pk, sk), tk) \leftarrow \mathsf{Gen}(par)$. Return (pk, sk).	$\widehat{\mathsf{Verify}}(pk, M, \sigma)$: Parse $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ and $\sigma_2 = pk'$. Return 1 if $\mathsf{Verify}(pk, \sigma_2, \sigma_1) = 1$ and $\mathsf{Verify}'(pk', M, \sigma_3) = 1$. Return 0 otherwise.
$\widehat{\mathsf{VerifySK}}(pk, sk)$: Return 1 if $\mathsf{VerifySK}(pk, sk) = 1$. Return 0 otherwise.	

Fig. 5. Generic construction Sig_1: TS + AKS (UF-otCMA).

UF-RMA secure TSs + UF-otCMA secure AKSs. Now we give another theorem showing that for the generic construction in Fig. 5, the security of the TS scheme can be weakened to the UF-RMA security if the AKS scheme satisfies the random auxiliary key property.

Theorem 7 *If Σ_t is a UF-RMA secure SKSP-TS scheme, and Σ_s a UF-otCMA secure SP-AKS scheme satisfying the random auxiliary key property, then* $\mathsf{Sig}_1 = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ *is a UF-CMA secure FSPS scheme.*

Proof sketch. The proof sketch of Theorem 7 is the same as that of Theorem 6 except that $\mathcal{B}_1$ is against the UF-RMA security of Σ_t instead of the UF-CMA security. To answer a query from $\mathcal{A}$, $\mathcal{B}_1$ makes a query to the signing oracle of Σ_t to obtain a randomly chosen auxiliary key ak' and the corresponding signature σ_1. Then $\mathcal{B}_1$ runs the additional algorithm AKGen (defined in Definition 13) on input (par, ak') to generate a verification/signing key pair (pk', sk'), which is indistinguishable from an honestly generated one according to the random auxiliary key property. Then it lets pk' be σ_2 and use sk' to sign the message. We refer the reader to the full paper for the proof of Theorem 7.

Instantiations of Sig_1. By combining the UF-CMA (respectively, UF-otRMA) secure TS scheme in Fig. 1 (respectively, Fig. 2) with the UF-otCMA secure AKS scheme in Fig. 3 (where $\mathbb{G}_1$ and $\mathbb{G}_2$ are swapped), we obtain an FSPS scheme satisfying UF-CMA (respectively, UF-otCMA) security. We refer the reader to the full paper for the resulting signature schemes.

Furthermore, by converting other previously proposed SPSs into SKSP-TSs and SP-AKSs, we obtain various FSPSs. We list some of them in Table 3 in Section 6.

5.2 Variation of Sig_1: Trapdoor Signature + Signature with Auxiliary Key (UF-CMA)

Now we give a variation of the generic construction in Fig. 6 by letting Σ_s be a UF-CMA secure SP-AKS scheme and sign n message blocks with one signing key. Each block is signed with an element indicating its number. This change reduces the signature and verification key sizes from $\Omega(n^2)$ to $\Omega(n)$ when signing n^2 group elements.

Let $\Sigma_t = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign}, \mathsf{VerifySK})$ be a γ-SKSP-TS scheme with message spaces $\mathcal{M}$ and $\mathcal{M}_\gamma$, and $\Sigma_s = (\mathsf{Setup}, \mathsf{Gen}', \mathsf{Sign}', \mathsf{Verify}')$[17] a γ-SP-AKS scheme with verification key space $\mathcal{M}_\gamma$, auxiliary key space $\mathcal{M}$, and message space $\mathcal{M}' \times \mathcal{M}_I$, where $\mathcal{M}_I$ is the space for elements indicating numbers of blocks. Then a generic construction of FSPS denoted by $\mathsf{Sig}_1^* = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ with message space $\mathcal{M}'^n$, where n is some fixed integer, is described as in Fig. 6.

$\widehat{\mathsf{Setup}}(1^\lambda)$: Run $par \leftarrow \mathsf{Setup}(1^\lambda)$. Determine the message spaces $\mathcal{M}$ and $\mathcal{M}_\gamma$ for Σ_t. Determine the message space $\mathcal{M}' \times \mathcal{M}_I$, verification key space $\mathcal{M}_\gamma$, and auxiliary key space $\mathcal{M}$ for Σ_s. Define $\gamma : \mathcal{M} \mapsto \mathcal{M}_\gamma$. Return par.	$\widehat{\mathsf{Sign}}(sk, \vec{M})$: Parse $\vec{M} = (M_1, \ldots, M_n) \in \mathcal{M}'^n$. $((pk', sk'), ak') \leftarrow \mathsf{Gen}'(par)$. $\sigma_1 \leftarrow \mathsf{Sign}(sk, ak')$. $\sigma_2 = pk'$. $\sigma_{3i} \leftarrow \mathsf{Sign}'(sk', (M_i, I(i)))$ where $I(i) \in \mathcal{M}_I$ for $i = 1, \ldots, n$. $\sigma_3 = (\sigma_{31}, \ldots, \sigma_{3n})$. Return $\sigma = (\sigma_1, \sigma_2, \sigma_3)$.
$\widehat{\mathsf{Gen}}(par)$: $((pk, sk), tk) \leftarrow \mathsf{Gen}(par)$. Return (pk, sk).	$\widehat{\mathsf{Verify}}(pk, \vec{M}, \sigma)$: Parse $\vec{M} = (M_1, \ldots, M_n) \in \mathcal{M}'^n$ and $\sigma = (\sigma_1, \sigma_2, \sigma_3)$. Return 1 if $\mathsf{Verify}(pk, \sigma_2, \sigma_1) = 1$ and $\mathsf{Verify}'(pk', (M_i, I(i)), \sigma_{3i}) = 1$ for all i. Return 0 otherwise.
$\widehat{\mathsf{VerifySK}}(pk, sk)$: Return 1 if $\mathsf{VerifySK}(pk, sk) = 1$. Return 0 otherwise.	

Fig. 6. Generic construction Sig_1^*: TS + AKS (UF-CMA).

For this generic construction, the following two theorems hold.

Theorem 8 *If Σ_t is a UF-CMA secure SKSP-TS scheme, and Σ_s a UF-CMA secure SP-AKS scheme, then $\mathsf{Sig}_1^* = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ is a UF-CMA secure FSPS scheme.*

Theorem 9 *If Σ_t is a UF-RMA secure SKSP-TS scheme, and Σ_s a UF-CMA secure SP-AKS scheme satisfying the random auxiliary key property, then $\mathsf{Sig}_1^* = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ is a UF-CMA secure FSPS scheme.*

We omit the proofs of Theorem 8 and Theorem 9 since they are similar to the proofs of Theorem 6 and Theorem 7, respectively. We list several instantiations of Sig^* in Table 3 in Section 6. Most of them achieve better efficiency than instantiations obtained from Sig_1, and are automorphic.

[17] As in [5], we assume that Σ_t and Σ_s share the common setup algorithm Setup.

5.3 Generic Construction Sig_2: Trapdoor Signature + Two-tier Signature with Auxiliary Keys

In this section, we give another generic construction of FSPS which provides us with FSPSs and FASs based on standard assumptions that have shorter verification keys and signatures.

Let $\Sigma_t = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{TDSign}, \mathsf{VerifySK})$ be a γ-TS scheme with message spaces $\mathcal{M}_p \times \mathcal{M}_s^n$ and $\mathcal{M}_{\gamma p} \times \mathcal{M}_{\gamma s}^n$, $\Sigma_s = (\mathsf{Setup}, \mathsf{PGen}, \mathsf{SGen}, \mathsf{TTSign}, \mathsf{TTVerify})$[18] a (γ_p, γ_s)-TT-AKS with primary/secondary verification key spaces $\mathcal{M}_{\gamma p}/\mathcal{M}_{\gamma s}$, auxiliary key spaces $\mathcal{M}_p/\mathcal{M}_s$, and message space $\mathcal{M}'$, where n is some fixed integer and $(\gamma_p(x_1), \gamma_s(x_2), \ldots, \gamma_s(x_{n+1})) = \gamma(x_1, x_2 \ldots, x_{n+1})$. A generic construction of FSPS denoted by $\mathsf{Sig}_2 = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ with message space $\mathcal{M}'^n$ is as described as in Fig. 7.

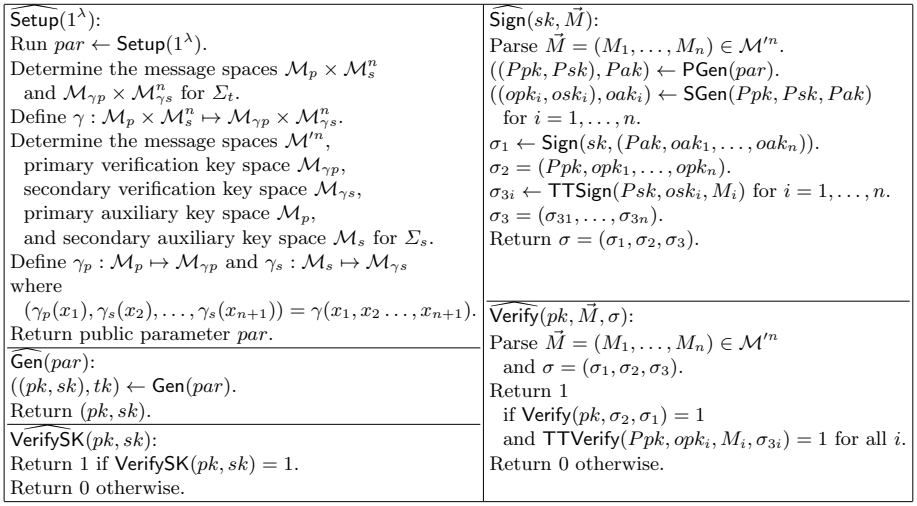

Fig. 7. Generic construction Sig_2: TS + TT-AKS.

For this generic construction, the following two theorems hold.

Theorem 10 *If Σ_t is a UF-CMA secure SKSP-TS scheme, and Σ_s a UF-TT-CMA secure SP-TT-AKS scheme, then $\mathsf{Sig}_2 = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ is a UF-CMA secure FSPS scheme.*

Theorem 11 *If Σ_t is a UF-RMA secure SKSP-TS scheme, and Σ_s a UF-TT-CMA secure SP-AKS scheme satisfying the random primary and secondary auxiliary key properties, then $\mathsf{Sig}_2 = (\widehat{\mathsf{Setup}}, \widehat{\mathsf{Gen}}, \widehat{\mathsf{Sign}}, \widehat{\mathsf{Verify}}, \widehat{\mathsf{VerifySK}})$ is a UF-CMA secure FSPS scheme.*

[18] As in [5], we assume that Σ_t and Σ_s share the common setup algorithm Setup.

The proofs of Theorem 10 and Theorem 11 are similar to the proofs of Theorem 6 and Theorem 7, respectively. We give them in the full paper.

Instantiations of Sig_2. By combining the UF-CMA (respectively, UF-otRMA) secure TS scheme in Fig. 1 (respectively, Fig. 2) with the UF-TT-CMA secure AKS scheme in Fig. 4 (where $\mathbb{G}_1$ and $\mathbb{G}_2$ are swapped), we obtain an FSPS scheme satisfying UF-CMA (respectively, UF-otCMA) security. We refer the reader to the full paper for the resulting signature schemes. Furthermore, we list several instantiations of Sig_2 in Table 3, Section 6.

6 Instantiations

In this section, we give several instantiations derived from our generic constructions, which are summarized in Table 3. For notational convenience, we denote these schemes as `(A)`, `(B)`, `(C)`, `(D)`, `(E)`, `(F)`, `(G)`, `(H)`, `(I)` (see the first column of Table 3) respectively. Many of these instantiations are FAS schemes.[19] It is not hard to see that typically, when signing m^2 group elements, Sig_1 needs $O(n^2)$ verification/signature key elements and $O(1)$ PPEs,[20] while Sig_1^* and Sig_2 need $O(n)$ verification/signature key elements and PPEs.

Besides the UF-CMA secure FSPS schemes in Table 3, we give several UF-otCMA instantiations derived from our generic constructions, which have relatively better efficiency. We refer the reader to the full paper for details.

In Section 6.1, Section 6.2, and Section 6.3, we give remarks on the instantiations of Sig_1, Sig_1^*, and Sig_2, respectively. Due to page limitation, we refer the reader to the full paper for signing key sizes and numbers of pairings required in verification.

6.1 Sig_1: SKSP-TS + SP-AKS

We give parameters of three instantiations for Sig_1, which are `(A)`, `(B)`, and `(C)`. Especially, `(B)` is an FSPS scheme in the type I bilinear map and `(C)` is an FSPS scheme in the generic group model.

The verification key size $|pk|$ of `(C)` is $(n_1, 0) \leq (n_1^2, 0)$, which makes it automorphic, while its efficiency (considering public parameter size, signature size, and verification cost) is very close to `(G)` (i.e., the FSPS scheme in [28]). As far as we know, `(C)` is the most efficient FAS scheme by now. Note that if we follow the definition of basic signatures in [3], which allows no trusted setup except for bilinear group generation, then `(C)` is not automorphic, and the most efficient FAS scheme becomes `(F)`, in Table 3.

[19] It is not hard to see that FAS schemes in Table 3 may lose the automorphic property when n_1 (or n_2 or n) is an extremely small number. Furthermore, when k (which is independent with the message size) is a large number, the message size has to be made reasonably large to keep the automorphic property.

[20] There may be exceptions, e.g., `(C)` in Table 3.

	Const.	Auto.	Assumption	Parameter	♯ Group element (PPE)
AKO+15 [5]	Generic construction 1	×	SXDH XDLIN	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $(n_1^2 + 5, n_1^2 + 11)$ $(7, 3n_1^2 + 7)$ $2n_1^2 + 7$
AKO+15 [5]	Generic construction 2	×	SXDH XDLIN	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $(6n_1 + 13, 4)$ $(2n_1 + 4, 2n_1 + 7)$ $n_1 + 5$
Gro15 [28]	FSPS scheme	×	Generic	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $(2n_1 - 1, 1)$ $(n_1 + 1, n_1)$ $n_1 + 1$
(A): KPW15 [31] (CMA) + KPW15 [31](otCMA)	Sig_1	×	$\mathcal{D}_k$-MDDH $(\mathbb{G}_1, \mathbb{G}_2)$	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $\left((n_1^2 k + 3k + 3 + \mathsf{RE}(\mathcal{D}_k))k + \mathsf{RE}(\mathcal{D}_k), 0\right)$ $\left(k + 2, (n_1^2 + 4)k + 3 + \mathsf{RE}(\mathcal{D}_k)\right)$ $3k + 1$
(B): ADK+13 [2] (CMA) + ADK+13 [2] (CMA)	Sig_1	×	2-Lin $(\mathbb{G}_1 = \mathbb{G}_2)$	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	n^2 $4n^2 + 60$ $2n^2 + 48$ 14
(C): Gro15 [28] (CMA) + Gro15 [28] (CMA)	Sig_1	√	Generic	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $(2n_1, 1)$ $(n_1 + 2, n_1 + 3)$ $n_1 + 3$
(D): KPW15 [31] (CMA) + KPW15 [31](CMA)	Sig_1^*	√	$\mathcal{D}_k$-MDDH $(\mathbb{G}_1, \mathbb{G}_2)$	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $\left((n_1 k + 2k^2 + 6k + 3 + \mathsf{RE}(\mathcal{D}_k))k + \mathsf{RE}(\mathcal{D}_k), 0\right)$ $\left(3n_1 k + 3n_1 + 1, (n_1 + 2k + 7)k + n_1 + 3 + \mathsf{RE}(\mathcal{D}_k)\right)$ $(2k + 1)(n_1 + 1)$
(E): ADK+13 [2] (CMA) + ADK+13 [2] (CMA)	Sig_1^*	√	2-Lin $(\mathbb{G}_1 = \mathbb{G}_2)$	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	n^2 $4n + 64$ $16n + 36$ $7(n + 1)$
(F): KPW15 [31] (CMA) + KPW15 [32] (TT)	Sig_2	√	$\mathcal{D}_k$-MDDH $(\mathbb{G}_1, \mathbb{G}_2)$	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $\left((2n_1 k + 2k + 3 + \mathsf{RE}(\mathcal{D}_k))k + \mathsf{RE}(\mathcal{D}_k), 0\right)$ $\left((k + 1)n_1 + 1, 2n_1 k + 3k + 3 + \mathsf{RE}(\mathcal{D}_k)\right)$ $kn_1 + 2k + 1$
(G): KPW15 [31] (CMA) (bilateral) + KPW15 [32] (TT)	Sig_2	√	$\mathcal{D}_k$-MDDH $(\mathbb{G}_1, \mathbb{G}_2)$	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	(n_1^2, n_2^2) $\left((2n_1 k + 3k + 3 + \mathsf{RE}(\mathcal{D}_k))k + \mathsf{RE}(\mathcal{D}_k), (2n_2 k + \mathsf{RE}(\mathcal{D}_k))k + \mathsf{RE}(\mathcal{D}_k)\right)$ $\left((k + 1)n_1 + 2n_2 k + k + 2 + \mathsf{RE}(\mathcal{D}_k), (k + 1)n_2 + 2n_1 k + 4k + 3 + \mathsf{RE}(\mathcal{D}_k)\right)$ $k(n_1 + n_2) + 3k + 1$
(H): ADK+13 [2] (CMA) + ADK+13 [2] (TT(TOS))	Sig_2	√	2-Lin $(\mathbb{G}_1 = \mathbb{G}_2)$	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	n^2 $6n + 30$ $6n + 12$ $2n + 7$
(I) ACD+12 [1] (CMA) + ACD+12 [1] (TT)	Sig_2	×	SXDH XDLIN	$\|m\|$ $\|pk\| + \|par\|$ $\|\sigma\|$ ♯ PPE	$(n_1^2, 0)$ $(2n_1 + 14, 7)$ $(2n_1 + 4, 2n_1 + 8)$ $n_1 + 4$

Table 3. Previously proposed FSPSs and FSPSs derived from our work. "Const." is short for "Construction" and "Auto." is short for "Automorphic". We use "(A): KPW15 [31] (CMA) + KPW15 [31] (otCMA)" to denote that the underlying TS (respectively, AKS) scheme of (A) is adapted from the UF-CMA secure (respectively, UF-otCMA secure) SPS scheme in [31]. We use the same argument for others except that the three FSPSs in the top denote the ones proposed in [5] and [28]. Especially, "ADK+13 [2] (TT(TOS))" denotes the tagged one-time signature scheme in [2]. Notation (x, y) denotes x elements in $\mathbb{G}_1$ and y elements in $\mathbb{G}_2$. As noted in Introduction, we do not count the two generators in the bilinear groups in the parameters.

6.2 Sig_1^*: SKSP-TS + SP-AKS (UF-CMA)

We give parameters of two instantiations for Sig_1^*, which are `(D)` and `(E)`, while `(E)` is in the type I bilinear map. Both of them are automorphic.

It is obvious that most instantiations derived from Sig_1 have verification key and signature sizes linear in the message size, which makes them less efficient and not automorphic (since verification keys have larger size than messages). However, as shown in Table 3, as a variation of Sig_1, Sig_1^* allows us to obtain FSPSs with shorter signatures and verification keys if the underlying SP-AKS scheme is UF-CMA secure. This fact shows that many existing SPSs imply the existence of a corresponding efficient FSPS scheme since any well-formed SPS scheme (respectively, SPS scheme with an algebraic key generation algorithm) can be converted into an SKSP-TS (respectively, SP-AKS) scheme.

6.3 Sig_2: SKSP-TS + SP-TT-AKS

We give parameters of four instantiations for Sig_2, which are `(F)`, `(G)`, `(H)`, and `(I)`, while `(H)` is in the type I bilinear map. The only one that is *not* automorphic among them is `(I)`. Here, `(G)` is achieved by using a UF-CMA secure SKSP-TS scheme to sign auxiliary keys of two SP-TT-AKS schemes with verification keys consisting of elements in $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, and `(H)` is achieved by using a SKSP-TS scheme to sign auxiliary keys of the tag based one-time signature scheme in [2]. Tag based one-time signatures can be treated as a special case of two-tier signatures where secondary signing keys are the same as secondary verification keys.

For $k = 1$ (SXDH), we have $(|m|, |pk + par|, |\sigma|, \sharp\text{PPEs}) = (n_1^2, 2n_1 + 7, 4n_1 + 8, n_1 + 3)$ in `(F)`, while the most efficient instantiation given in [5] achieves $(|pk + par|, |\sigma|, \sharp\text{PPEs}) = (6n_1 + 17, 4n_1 + 11, n_1 + 5)$ and is not automorphic. Furthermore, by sacrificing efficiency, `(F)` can be based on weaker assumptions.

`(G)` achieves $(|m|, |pk| + |par|, |\sigma|, \sharp\text{PPEs}) = (n_1^2, 2n_1 + 2n_2 + 10, 4n_1 + 4n_2 + 12, n_1 + n_2 + 4)$ for $k = 1$ (SDXH), which has the shortest verification key size, signature size, and lowest cost in verification among all FSPS and FAS schemes with a bilateral message space based on standard assumptions by now, as far as we know.

`(H)` is the most efficient FSPS and FAS scheme in the *t*ype I bilinear map, as far as we know.

References

1. Abe, M., Chase, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Constant-size structure-preserving signatures: Generic constructions and simple assumptions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 4–24. Springer (2012)
2. Abe, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Tagged one-time signatures: Tight security and optimal tag size. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 312–331. Springer (2013)

3. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer (2010)
4. Abe, M., Groth, J., Haralambiev, K., Ohkubo, M.: Optimal structure-preserving signatures in asymmetric bilinear groups. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 649–666. Springer (2011)
5. Abe, M., Kohlweiss, M., Ohkubo, M., Tibouchi, M.: Fully structure-preserving signatures and shrinking commitments. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 35–65. Springer (2015)
6. Attrapadung, N., Libert, B., Peters, T.: Computing on authenticated data: New privacy definitions and constructions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 367–385. Springer (2012)
7. Baldimtsi, F., Chase, M., Fuchsbauer, G., Kohlweiss, M.: Anonymous transferable e-cash. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 101–124. Springer (2015)
8. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer (2009)
9. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: P-signatures and noninteractive anonymous credentials. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 356–374. Springer (2008)
10. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer (2003)
11. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: The case of dynamic groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer (2005)
12. Bellare, M., Shoup, S.: Two-tier signatures, strongly unforgeable signatures, and fiat-shamir without random oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. Lecture Notes in Computer Science, vol. 4450, pp. 201–216. Springer (2007)
13. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. J. Cryptology 22(1), 114–138 (2009)
14. Camenisch, J., Dubovitskaya, M., Enderlein, R.R., Neven, G.: Oblivious transfer with hidden access control from attribute-based encryption. In: Visconti, I., Prisco, R.D. (eds.) SCN 2012. LNCS, vol. 7485, pp. 559–579. Springer (2012)
15. Camenisch, J., Krenn, S., Shoup, V.: A framework for practical universally composable zero-knowledge protocols. In: ASIACRYPT 2011. pp. 449–467 (2011)
16. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer (2001)
17. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable proof systems and applications. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 281–300. Springer (2012)
18. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable signatures: New definitions and delegatable anonymous credentials. In: CSF 2014. pp. 199–213. IEEE (2014)
19. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Information Theory 31(4), 469–472 (1985)
20. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.L.: An algebraic framework for Diffie-Hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 129–147. Springer (2013)

21. Even, S., Goldreich, O., Micali, S.: On-line/off-line digital signatures. J. Cryptology 9(1), 35–67 (1996)
22. Fuchsbauer, G.: Commuting signatures and verifiable encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 224–245. Springer (2011)
23. Fuchsbauer, G., Hanser, C., Kamath, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model from weaker assumptions. In: Zikas, V., Prisco, R.D. (eds.) SCN 2016. Lecture Notes in Computer Science, vol. 9841, pp. 391–408. Springer (2016)
24. Fuchsbauer, G., Hanser, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. Lecture Notes in Computer Science, vol. 9216, pp. 233–253. Springer (2015)
25. Fuchsbauer, G., Vergnaud, D.: Fair blind signatures without random oracles. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 16–33. Springer (2010)
26. Ghadafi, E.: More efficient structure-preserving signatures - or: Bypassing the type-III lower bounds. Cryptology ePrint Archive, Report 2016/255 (2016)
27. Ghadafi, E.: Short structure-preserving signatures. In: Sako, K. (ed.) CT-RSA 2016. LNCS, vol. 9610, pp. 305–321. Springer (2016)
28. Groth, J.: Efficient fully structure-preserving signatures for large messages. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 239–259. Springer (2015)
29. Groth, J., Sahai, A.: Efficient noninteractive proof systems for bilinear groups. SIAM J. Comput. 41(5), 1193–1232 (2012)
30. Hofheinz, D., Jager, T.: Tightly secure signatures and public-key encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 590–607. Springer (2012)
31. Kiltz, E., Pan, J., Wee, H.: Structure-preserving signatures from standard assumptions, revisited. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 275–295. Springer (2015)
32. Kiltz, E., Pan, J., Wee, H.: Structure-preserving signatures from standard assumptions, revisited. IACR Cryptology ePrint Archive 2015, 604 (2015)
33. Libert, B., Peters, T., Joye, M., Yung, M.: Linearly homomorphic structure-preserving signatures and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 289–307. Springer (2013)
34. Libert, B., Peters, T., Yung, M.: Group signatures with almost-for-free revocation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 571–589. Springer (2012)
35. Libert, B., Peters, T., Yung, M.: Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 296–316. Springer (2015)
36. Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multisignatures: extended abstract. In: Reiter, M.K., Samarati, P. (eds.) CCS 2001. pp. 245–254. ACM (2001)
37. Rial, A., Kohlweiss, M., Preneel, B.: Universally composable adaptive priced oblivious transfer. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 231–247. Springer (2009)
38. Schwartz, J.T.: Fast probabilistic algorithms for verification of polynomial identities. J. ACM 27(4), 701–717 (1980)