Key-Homomorphic Signatures: Definitions and Applications to Multiparty Signatures and Non-Interactive Zero-Knowledge

David Derler[1,‡] and Daniel Slamanig[2,‡]

[1] DFINITY
david@dfinity.org
[2] AIT Austrian Institute of Technology
daniel.slamanig@ait.ac.at

Abstract. Key-homomorphic properties of cryptographic objects, i.e., homomorphisms on their key space, have proven to be useful, both from a theoretical as well as a practical perspective. Important cryptographic objects such as pseudorandom functions or (public key) encryption have been studied previously with respect to key-homomorphisms. Interestingly, however, signature schemes have not been explicitly investigated in this context so far.

We close this gap and initiate the study of key-homomorphic signatures, which turns out to be an interesting and versatile concept. In doing so, we firstly propose a definitional framework for key-homomorphic signatures distilling various natural flavours of key-homomorphic properties. Those properties aim to classify existing signature schemes and thus allow to infer general statements about signature schemes from those classes by simply making black-box use of the respective properties. We apply our definitional framework to show elegant and simple compilers from classes of signature schemes admitting different types of key-homomorphisms to a number of other interesting primitives such as ring signature schemes, (universal) designated verifier signature schemes, simulation-sound extractable non-interactive zero-knowledge (NIZK) arguments, and multisignature schemes. Additionally, using the formalisms provided by our framework, we can prove a tight implication from single-user security to key-prefixed multi-user security for a class of schemes admitting a certain key-homomorphism.

Finally, we discuss schemes that provide homomorphic properties on the message space of signatures under different keys in context of key-homomorphisms and present some first constructive results from key-homomorphic schemes.

‡ Work done while with Graz University of Technology.

Table of Contents

Key-Homomorphic Signatures: Definitions and Applications to Multiparty Signatures and Non-Interactive Zero-Knowledge 1
David Derler and Daniel Slamanig
1 Introduction 3
1.1 Contribution 4
1.2 Differences to the Journal Version [DS18] 6
2 Preliminaries 6
3 Key-Homomorphic Signatures 10
3.1 Definitional Framework for Key-Homomorphic Signatures 11
4 Overview of Key-Homomorphic Schemes 14
4.1 Schnorr Signatures [Sch91] 16
4.2 Guillou-Quisquater [GQ88] 16
4.3 BLS Signatures [BLS04] 17
4.4 Katz-Wang Signatures [KW03, GJKW07] 17
4.5 Waters' Signatures [Wat05] 18
4.6 PS Signatures [PS16] 19
4.7 Randomizable SPS by Abe et al. [AGOT14] 20
4.8 Ghadafi's Short SPS [Gha16] 20
5 Applications to Multiparty Signatures 21
5.1 Ring Signatures 21
5.2 Universal Designated Verifier Signatures 26
5.3 Multisignatures 32
6 Applications to Simulation-Sound Extractable NIZK 34
6.1 Weak Simulation-Sound Extractability 40
6.2 Signatures of Knowledge 40
6.3 Performance Advantages 41
7 Tight Multi-User Security from Key-Homomorphisms 43
8 Summary and Conclusion 44
A Homomorphisms on Key and Message Space 50
A.1 Multikey-Homomorphic Signatures from Key-Homomorphisms 51

1 Introduction

The design of cryptographic schemes that possess certain homomorphic properties on their *message space* has witnessed significant research within the last years. In the domain of encryption, the first candidate construction of fully homomorphic encryption (FHE) due to Gentry [Gen09] has initiated a fruitful area of research with important applications to computations on (outsourced) encrypted data. In the domain of signatures, the line of work on homomorphic signatures [JMSW02], i.e., signatures that are homomorphic with respect to the message space, has only quite recently attracted attention. Firstly, due to the introduction of computing on authenticated data [ABC+12]. Secondly, due to the growing interest in the application to verifiable delegation of computations (cf. [Cat14] for a quite recent overview), and, finally, due to the recent construction of fully homomorphic signatures [GVW15, BFS14].

In this paper we are interested in another type of homomorphic schemes, so called key-homomorphic schemes, that is, schemes that possess certain homomorphic properties on their *key space*. Specifically, we study *key-homomorphic signature schemes* and will show that this concept turns out to be a very interesting and versatile tool. Looking ahead, in context of signatures, we can define key-homomorphisms in various different ways. We subsequently sketch two examples to provide a first intuition. One notion is to require that given two signatures for the same message m valid under some pk_1 and pk_2 respectively, one can publicly compute a signature to message m that is valid for a public key pk' that is obtained via some operation on pk_1 and pk_2. Another variant for instance is to require that, given a signature σ to a message m that verifies under pk, σ can be adapted to a signature to m under pk'. Thereby, pk and pk' have a well defined relationship (cf. Section 3 for the details).

Although key-homomorphic signatures have never been discussed or studied explicitly , some implicit use of key-homomorphisms can be found. A recent work by Kiltz et al. [KMP16] introduces a property for canonical identification schemes denoted as random self-reducibility. This basically formalizes the re-randomization of key-pairs as well as adapting parts of transcripts of identification protocols consistently. Earlier, Fischlin and Fleischhacker in [FF13] used re-randomization of key-pairs implicitly in their meta reduction technique against Schnorr signatures. This concept has recently been formalized, yielding the notion of signatures with re-randomizable keys [FKM+16].[1] These signatures with re-randomizable keys are then used as basis of an elegant construction of unlinkable sanitizable signatures (cf. [FKM+16]). Allowing the adversary to also access signatures under re-randomized (related) keys, has earlier been studied in context of security of signature schemes against related-key attacks (RKAs) [BCM11, BPT12]. In context of RKA, the goal is to prevent that signature schemes have key-homomorphic properties that allow to adapt signatures under related keys to signatures under the original key (cf. e.g., [MSM+15]).

[1] In such schemes the EUF-CMA security notion is slightly modified, by additionally allowing the adversary to see signatures under re-randomized keys.

Previous Work on Key-Homomorphic Primitives. While our work is the first that explicitly studies key-homomorphic properties of signatures, some other primitives have already been studied with respect to key-homomorphic properties previously. Applebaum et al. in [AHI11] studied key-homomorphic symmetric encryption schemes in context of related key attacks (RKAs). Recently, Dodis et al. [DMS16] have shown that any such key-homomorphic symmetric encryption schemes implies public key encryption. Rothblum [Rot11] implicitly uses key malleability to construct (weakly) homomorphic public key bit-encryption schemes from private key ones. Goldwasser et al. in [GLW12], and subsequently Tessaro and Wilson in [TW14], use public key encryption schemes with linear homomorphisms over their keys (and some related properties) to construct a weaker form of identity-based encryption (IBE), namely bounded-collusion IBE. Recently, Boneh et al. introduced the most general notion of fully key-homomorphic encryption [BGG+14]. In such a scheme, when given a ciphertext under a public key pk, anyone can translate it into a ciphertext to the same plaintext under public key $(f(\mathsf{pk}), f)$ for any efficiently computable function f. Another line of work recently initiated by Boneh et al. [BLMR13] is concerned with key-homomorphic pseudorandom functions (PRFs) and pseudo random generators (PRGs). Loosely speaking, a secure PRF family $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, is key-homomorphic if the keys live in a group $(\mathcal{K}, +)$, and, given two evaluations $F(k_1, x)$ and $F(k_2, x)$ for the same value under two keys, one can efficiently compute $F(k_1 + k_2, x)$. Such PRFs turn out to yield interesting applications such as distributed PRFs, symmetric key proxy re-encryption or updatable encryption. Continuing the work in this direction, alternative constructions [BP14] and extended functionality in the form of constrained key-homomorphic PRFs have been proposed [BFP+15]. We note that the result from Dodis et al. [DMS16], although not mentioned, answers the open question posed by Boneh et al. [BLMR13] "whether key-homomorphic PRFs whose performance is comparable to real-world block ciphers such as AES exist" in a negative way. Finally, Benhamouda et al. use key-homomorphic smooth projective hash functions to construct aggregator oblivious encryption schemes [BJL16] and inner-product functional encryption schemes [BBL17].

1.1 Contribution

We initiate the study of key-homomorphic signature schemes. In doing so, we propose various natural definitions of key-homomorphic properties for signatures, which can be used to classify existing signature schemes. This allows to infer general statements about signature schemes from the respective classes by simply making black-box use of the respective properties. On our way we rule out certain combinations of key-homomorphism and existing unforgeability notions of signature schemes. We then show how different types of key-homomorphic signature schemes allow to construct more advanced signature schemes and efficient simulation-sound extractable NIZKs, and how they help to strengthen the security of signatures in the multi-user setting. We study existing signature schemes with respect to our key-homomorphic properties and explicitly provide examples for all these classes, so that all our results can readily be instantiated.

From a theoretical viewpoint our results contribute towards establishing a better understanding of the paradigms which are necessary to construct certain schemes and/or to achieve certain security notions. In particular, we start from very mild security requirements and show how to use our framework to amplify those to yield relatively strong security guarantees. From a practical viewpoint, our so obtained constructions compare favorably to existing work: they are conceptually very easy to understand and implement and therefore less prone to wrong usage. At the same time, our results yield instantiations with no or even reduced overhead when compared to existing work.

More specifically, besides our definitional framework for key-homomorphic signatures, which we see as a contribution on its own, the contributions in this paper are as follows:[2]

Generic Compilers. We present compilers from classes of schemes providing different types of key-homomorphisms to ring signatures [RST01], (universal) designated verifier signatures [SBWP03], simulation-sound extractable NIZK arguments [Gro06] (SSE NIZKs henceforth), and compact multisignatures [IN83]. The so obtained constructions, besides being very efficient, are simple from a construction and security analysis point of view. Basically, for ring signatures, (universal) designated verifier signatures and weakly SSE NIZKs, one computes a signature using any suitable key-homomorphic scheme under a freshly sampled key and then proves a simple relation over public keys *only*. For SSE NIZKs we additionally require a strong one-time signature scheme. Compact multisignatures are directly implied by signatures with certain key-homomorphic properties.

Tight Key-Prefixed Multi-User Security. We prove a theorem which tightly relates the single-user existential unforgeability under chosen message attacks (EUF-CMA) security of a class of schemes admitting a particular key-homomorphism to its key-prefixed multi-user EUF-CMA security. This theorem addresses a frequently occurring question in the context of standardization and generalizes existing theorems [Ber15, Lac18] (where such implications are proven for concrete signature schemes) so that it is applicable to a larger class of signature schemes.

(Standard Model & Standard Assumption) Instantiations. We investigate existing signature schemes with respect to whether they admit different types of key-homomorphisms. Using our compilers, this directly yields previously unknown instantiations of all variants of signature schemes mentioned above. Most interestingly, we can show that a variant of Waters' signatures [Wat05] in Type-3 bilinear groups satisfies our notion of perfect adaptability. This, in turn gives us novel and simple constructions of ring signatures, simulation-sound extractable NIZK arguments, and universal designated verifier signatures without random oracles from standard assumptions.[3] For universal designated verifier signatures, we even obtain the first instantiation from standard assumptions in the standard model. Likewise, our general theorem for multi-user security attests

[2] We also present a compact summary of the results in Section 8, Figure 1.

[3] We can use witness-indistinguishable Groth-Sahai [GS08] proofs as argument system and for instance the strong one-time signatures under standard assumptions from Groth [Gro06].

the multi-user security for schemes whose multi-user security was previously unknown. All our instantiations compare favorably to existing constructions regarding conceptual simplicity and/or come at no or even reduced computational overhead. For example, when instantiating our compiler for SSE NIZKs without random oracles this mainly owes to the fact that—compared to earlier work—our techniques yield to a reduced overhead for the part of the system which involves proving knowledge of a signature on the proven statement. That is, we only need to prove a simple relation between two keys (which can be done with a single group element from $\mathbb{G}_1$ when using the aforementioned Waters's variant) and can do the verification in plain instead of proving knowledge of an (encrypted) signature. Additionally, when instantiating our compilers for ring signatures and universal designated verifier signatures without random oracles with Groth Sahai proofs and Waters' signatures they enjoy similar properties regarding easy implementation and only require to prove knowledge of a single element from $\mathbb{G}_1$ when looking at the overhead imposed by the proof (refer to the last paragraph of Section 6 for more details).

Multikey-Homomorphic Signatures. We investigate so called multikey-homomorphic signatures, which provide homomorphic properties on the message space of signatures under different keys[4], in context of key-homomorphisms and present some first constructive results on multikey-homomorphic signatures with a succinct verification key. Since we do not see this as the central contribution of this paper, we postpone the presentation to Appendix A.

1.2 Differences to the Journal Version [DS18]

The journal version of this paper [DS18] does not to include the part on multikey-homomorphic signatures. We include it in this full version as Appendix A.

2 Preliminaries

We denote algorithms by sans-serif letters, e.g., A, B. If not stated otherwise, all algorithms are required to run in polynomial time and return a special symbol $\perp$ on error. By $y \leftarrow \mathsf{A}(x)$, we denote that y is assigned the output of the potentially probabilistic algorithm A on input x and fresh random coins. Similarly, $y \xleftarrow{R} S$ means that an element is sampled uniformly at random from a set S and assigned to y, and we use $\mathcal{Q} \xleftarrow{\cup} z$ as a shorthand for $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{z\}$. We let $[n] := \{1, \ldots, n\}$ and write $\mathsf{Pr}[\Omega : \mathcal{E}]$ to denote the probability of an event $\mathcal{E}$ over the probability space Ω. We use $\mathcal{C}$ to denote challengers of security experiments, and $\mathcal{C}_\kappa$ to make the security parameter explicit. A function $\varepsilon(\cdot) : \mathbb{N} \to \mathbb{R}_{\geq 0}$

[4] Fiore et al. [FMNP16] in independent and concurrent work introduce the concept of multi-key homomorphic authenticators (MACs and signatures). In another more recent and independent work Lai et al. [LTWC18] study different multi-key homomorphic signatures under stronger security guarantees. We defer a discussion about relations to Appendix A.

is called negligible, iff it vanishes faster than every inverse polynomial, i.e., $\forall\ k : \exists\ n_k : \forall\ n > n_k : \varepsilon(n) < n^{-k}$. We use ρ to denote the success ration of an adversary, i.e., the quotient of its success probability and its running time. Finally, we use $\mathsf{poly}(\cdot)$ to denote a polynomial function.

One-Way Functions. Below, we recall the notion of one-way functions.

Definition 1 (One-Way Function). *A function $f : \{0,1\}^* \to \{0,1\}^*$ is called a one-way function, if (1) there exists a PPT algorithm $\mathcal{A}_1$ so that $\forall\ x \in \{0,1\}^* : \mathcal{A}_1(x) = f(x)$, and if (2) for every PPT algorithm $\mathcal{A}_2$ there is a negligible function $\varepsilon(\cdot)$ such that it holds that*

$$\mathsf{Pr}\left[x \xleftarrow{R} \{0,1\}^\kappa,\ x^\star \leftarrow \mathcal{A}_2(1^\kappa, f(x))\ :\ f(x) = f(x^\star)\right] \leq \varepsilon(\kappa).$$

Signature Schemes. Subsequently, we recall the definition of signature schemes.

Definition 2 (Signature Scheme). *A signature scheme Σ is a triple* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *of PPT algorithms, which are defined as follows:*

$\mathsf{KeyGen}(1^\kappa)$: *This algorithm takes a security parameter κ as input and outputs a secret (signing) key sk and a public (verification) key pk with associated message space $\mathcal{M}$ (we may omit to make the message space $\mathcal{M}$ explicit).*

$\mathsf{Sign}(\mathsf{sk}, m)$: *This algorithm takes a secret key sk and a message $m \in \mathcal{M}$ as input and outputs a signature σ.*

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: *This algorithm takes a public key pk, a message $m \in \mathcal{M}$ and a signature σ as input and outputs a bit $b \in \{0,1\}$.*

We note that for a signature scheme many independently generated public keys may be with respect to the same parameters pp, e.g., some elliptic curve group parameters. In such a case we introduce an additional algorithm PGen which is run by some (trusted) party to obtain $\mathsf{pp} \leftarrow \mathsf{PGen}(1^\kappa)$ and key generation requires pp (which implicitly contain the security parameter) to produce keys as $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$. Moreover, we then assume that pp is implicitly available to the Sign and Verify algorithms, e.g., attached to every sk and pk.

A digital signature scheme Σ needs to provide correctness as well as some unforgeability notion. We first present the correctness definition.

Definition 3 (Correctness). *A digital signature scheme Σ is correct, if for all security parameters $\kappa \in \mathbb{N}$, for all $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, for all $m \in \mathcal{M}$, we have that $\mathsf{Pr}\big[\mathsf{Verify}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1\big] = 1$.*

Note that the above definition captures correctness in the perfect sense. This is because the schemes for which we present compilers also define correctness in a perfect sense. We, however, note that our compilers also work with signature schemes with a negligible correctness error if we relax the respective correctness definitions for the schemes which are output by the compilers.

Below, we present two standard unforgeability notions required in our context (ordered from weak to strong). We start with universal unforgeability under no message attacks ($\mathsf{UUF\text{-}NMA}$ security).

Definition 4 (UUF-NMA). *A signature scheme* Σ *is* UUF-NMA *secure, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that*

$$\mathsf{Pr}\left[\begin{array}{l}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa),\ m^\star \xleftarrow{R} \mathcal{M},\\ \sigma^\star \leftarrow \mathcal{A}(\mathsf{pk}, m^\star)\end{array} \ : \ \mathsf{Verify}(\mathsf{pk}, m^\star, \sigma^\star) = 1\right] \leq \varepsilon(\kappa).$$

The most common notion is existential unforgeability under adaptively chosen message attacks (EUF-CMA security).

Definition 5 (EUF-CMA). *A signature scheme* Σ *is* EUF-CMA *secure, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that*

$$\mathsf{Pr}\left[\begin{array}{l}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa),\\ (m^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{pk})\end{array} \ : \ \begin{array}{r}\mathsf{Verify}(\mathsf{pk}, m^\star, \sigma^\star) = 1 \ \wedge\\ m^\star \notin \mathcal{Q}^{\mathsf{Sign}}\end{array}\right] \leq \varepsilon(\kappa),$$

where the environment keeps track of the queries to the signing oracle via $\mathcal{Q}^{\mathsf{Sign}}$.

Security of Multiparty Signatures. Most security definitions for multiparty signatures, i.e,. signatures that involve a set of (potential) signers such as ring signatures or multisignatures, implicitly assume the so called knowledge of secret key (KOSK) assumption, where the adversary is required to reveal the secret keys it uses to the environment. This is important to prevent rogue-key attacks, i.e., attacks where the adversary constructs malicious public keys based on existing public keys in the system so that it is not required to know all the corresponding secret keys.

Ristenpart and Yilek [RY07] introduced and formalized an abstract key-registration concept for multiparty signatures. Any such key-registration protocol is represented as a pair of interactive algorithms $(\mathsf{RegP}, \mathsf{RegV})$. A party registering a key runs RegP with inputs public key pk and private key sk. A certifying authority (CA) runs RegV, where the last message is from RegV to RegP and contains either a pk or $\perp$. In the plain model $\mathsf{RegP}(\mathsf{pk}, \mathsf{sk})$ simply sends pk to the CA and RegV on receiving pk simply returns pk. For the KOSK assumption, $\mathsf{RegP}(\mathsf{pk}, \mathsf{sk})$ simply sends $(\mathsf{pk}, \mathsf{sk})$ to the CA, which checks if $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{KeyGen}(\mathsf{PP})$ and if so replies with pk and $\perp$ otherwise.

To get rid of the KOSK assumption without revealing the secret key in real world protocols, one can require the adversary to prove knowledge of it's secret key in a way that it can be straight-line extracted by the environment. Also note that this does not conflict with very efficient deployments in the standard model: key registration can be interactive (e.g., a PKI may require an interactive proof before issuing a certificate on a public key) and one can use standard techniques without random oracles. We assume this to happen for all our multiparty signature schemes constructed in Section 5. Yet, we do not make it explicit to avoid complicated models and we simply introduce an RKey oracle that allows the adversary to register key pairs. We stress that our goal is not to study multiparty signatures with respect to real world key-registration procedures, as done in [RY07].

Non-Interactive Proof Systems. Now, we recall a standard definition of non-interactive proof systems Π. Our definitions are inspired by [DHLW10, ADK+13].

Therefore, let L_R be an **NP**-language with witness relation R defined as $L_R = \{x \mid \exists\ w : R(x,w) = 1\}$.

Definition 6 (Non-Interactive Proof System). *A non-interactive proof system* Π *is a tuple of algorithms* $(\mathsf{Setup}, \mathsf{Proof}, \mathsf{Verify})$*, which are defined as follows:*

$\mathsf{Setup}(1^\kappa)$: *This algorithm takes a security parameter* κ *as input, and outputs a common reference string* crs.

$\mathsf{Proof}(\mathsf{crs}, x, w)$: *This algorithm takes a common reference string* crs*, a statement* x*, and a witness* w *as input, and outputs a proof* π.

$\mathsf{Verify}(\mathsf{crs}, x, \pi)$: *This algorithm takes a common reference string* crs*, a statement* x*, and a proof* π *as input, and outputs a bit* $b \in \{0,1\}$.

Now, we recall formal definitions of the security properties. We thereby relax our definitions to computationally sound proof systems (argument systems).

Definition 7 (Completeness). *A non-interactive proof system* Π *is* complete*, if for every adversary* $\mathcal{A}$ *it holds that*

$$\Pr\left[\begin{matrix}\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ (x^\star, w^\star) \leftarrow \mathcal{A}(\mathsf{crs}), \\ \pi \leftarrow \mathsf{Proof}(\mathsf{crs}, x^\star, w^\star)\end{matrix} \ :\ \begin{matrix}\mathsf{Verify}(\mathsf{crs}, x^\star, \pi) = 1 \\ \vee\ (x^\star, w^\star) \notin R\end{matrix}\right] = 1.$$

Definition 8 (Soundness). *A non-interactive proof system* Π *is* sound*, if for every PPT adversary* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that*

$$\Pr\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ (x^\star, \pi^\star) \leftarrow \mathcal{A}(\mathsf{crs}) \ :\ \begin{matrix}\mathsf{Verify}(\mathsf{crs}, x^\star, \pi^\star) = 1 \\ \wedge\ x^\star \notin L_R\end{matrix}\right] \leq \varepsilon(\kappa).$$

Definition 9 (Adaptive Witness-Indistinguishability). *A non-interactive proof system* Π *is* adaptively witness-indistinguishable*, if for every PPT adversary* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that*

$$\Pr\Big[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa),\ b \xleftarrow{R} \{0,1\},\ b^\star \leftarrow \mathcal{A}^{\mathcal{P}(\mathsf{crs},\cdot,\cdot,\cdot,b)}(\mathsf{crs}) \ :\ b = b^\star\Big] \leq \varepsilon(\kappa),$$

where $\mathcal{P}(\mathsf{crs}, x, w_0, w_1, b) \coloneqq \mathsf{Proof}(\mathsf{crs}, x, w_b)$*, and* $\mathcal{P}$ *returns* $\perp$ *if* $(x, w_0) \notin R\ \vee\ (x, w_1) \notin R$.

If $\varepsilon = 0$, we have perfect adaptive witness-indistinguishability.

Definition 10 (Adaptive Zero-Knowledge). *A non-interactive proof system* Π *is* adaptively zero-knowledge*, if there exists a PPT simulator* $S = (\mathsf{S}_1, \mathsf{S}_2)$ *such that for every adversary* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that*

$$\left|\begin{matrix}\Pr\Big[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa) \ :\ \mathcal{A}^{\mathcal{P}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) = 1\Big] \ - \\ \Pr\Big[(\mathsf{crs}, \tau) \leftarrow \mathsf{S}_1(1^\kappa) \ :\ \mathcal{A}^{\mathcal{S}(\mathsf{crs},\tau,\cdot,\cdot)}(\mathsf{crs}) = 1\Big]\end{matrix}\right| \leq \varepsilon(\kappa),$$

where, τ *denotes a simulation trapdoor. Thereby,* $\mathcal{P}$ *and* $\mathcal{S}$ *return* $\perp$ *if* $(x, w) \notin R$ *or* $\pi \leftarrow \mathsf{Proof}(\mathsf{crs}, x, w)$ *and* $\pi \leftarrow \mathsf{S}_2(\mathsf{crs}, \tau, x)$*, respectively, otherwise.*

If $\varepsilon = 0$, we have perfect adaptive zero-knowledge. It is easy to show that adaptive zero-knowledge implies adaptive witness indistinguishability.

Definition 11 (Proof of Knowledge). *A non-interactive proof system Π is a proof of knowledge, if there exists a PPT extractor $\mathsf{E} = (\mathsf{E}_1, \mathsf{E}_2)$ such that for every PPT adversary $\mathcal{A}$ there is a negligible function $\varepsilon_1(\cdot)$ such that*

$$\left| \begin{array}{l} \mathsf{Pr}\big[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\kappa) \ : \ \mathcal{A}(\mathsf{crs}) = 1\big] \ - \\ \mathsf{Pr}\big[(\mathsf{crs}, \xi) \leftarrow \mathsf{E}_1(1^\kappa) \ : \ \mathcal{A}(\mathsf{crs}) = 1\big] \end{array} \right| \leq \varepsilon_1(\kappa),$$

and for every PPT adversary $\mathcal{A}$ there is a negligible function $\varepsilon_2(\cdot)$ such that

$$\mathsf{Pr}\left[\begin{array}{l} (\mathsf{crs}, \tau) \leftarrow \mathsf{E}_1(1^\kappa), \ (x^\star, \pi^\star) \leftarrow \mathcal{A}(\mathsf{crs}), \\ w \leftarrow \mathsf{E}_2(\mathsf{crs}, \xi, x^\star, \pi^\star) \end{array} : \begin{array}{r} \mathsf{Verify}(\mathsf{crs}, x^\star, \pi^\star) = 1 \ \wedge \\ (x^\star, w) \notin R \end{array} \right] \leq \varepsilon_2(\kappa).$$

Definition 12 (Simulation-Sound Extractability). *An adaptively zero-knowledge non-interactive proof system Π is simulation-sound extractable, if there exists a PPT extractor (S, E) such that for every adversary $\mathcal{A}$ it holds that*

$$\left| \begin{array}{l} \mathsf{Pr}\big[(\mathsf{crs}, \tau) \leftarrow \mathsf{S}_1(1^\kappa) \ : \ \mathcal{A}(\mathsf{crs}, \tau) = 1\big] \ - \\ \mathsf{Pr}\big[(\mathsf{crs}, \tau, \xi) \leftarrow \mathsf{S}(1^\kappa) \ : \ \mathcal{A}(\mathsf{crs}, \tau) = 1\big] \end{array} \right| = 0,$$

and for every PPT adversary $\mathcal{A}$ there is a negligible function $\varepsilon_2(\cdot)$ such that

$$\mathsf{Pr}\left[\begin{array}{l} (\mathsf{crs}, \tau, \xi) \leftarrow \mathsf{S}(1^\kappa), \\ (x^\star, \pi^\star) \leftarrow \mathcal{A}^{\mathcal{S}(\mathsf{crs}, \tau, \cdot)}(\mathsf{crs}), \\ w \leftarrow \mathsf{E}(\mathsf{crs}, \xi, x^\star, \pi^\star) \end{array} : \begin{array}{r} \mathsf{Verify}(\mathsf{crs}, x^\star, \pi^\star) = 1 \ \wedge \\ (x^\star, \pi^\star) \notin \mathcal{Q}_\mathsf{S} \ \wedge \ (x^\star, w) \notin R \end{array} \right] \leq \varepsilon_2(\kappa),$$

where $\mathcal{S}(\mathsf{crs}, \tau, x) := \mathsf{S}_2(\mathsf{crs}, \tau, x)$ and $\mathcal{Q}_\mathsf{S}$ keeps track of the queries to and answers of $\mathcal{S}$.

Note that the definition of simulation-sound extractability of [Gro06] is stronger than ours in the sense that the adversary also gets the trapdoor ξ as input. However, in our context this weaker notion (previously also used in other works such as [DHLW10, ADK+13]) is sufficient.

Definition 13 (Weak Simulation-Sound Extractability). *An adaptively zero-knowledge non-interactive proof system Π is weakly simulation-sound extractable, if it satisfies Definition 12 with the following modified winning condition:* $\mathsf{Verify}(\mathsf{crs}, x^\star, \pi^\star) = 1 \ \wedge \ (x^\star, \cdot) \notin \mathcal{Q}_\mathsf{S} \ \wedge \ (x^\star, w) \notin R$.

3 Key-Homomorphic Signatures

In this section, we introduce a definitional framework for key-homomorphic signature schemes. In doing so, we propose different natural notions and relate the definitions to previous work that already implicitly used functionality that is related or covered by our definitions.[5] Finally, we discuss signatures with homomorphic properties on their key and message space, i.e., we investigate multi-key homomorphic signatures with respect to key-homomorphisms.

[5] We note that the first parts (up to Definition 16) are more general versions of definitions that we earlier have used for constructing specific redactable signatures [DKS16].

3.1 Definitional Framework for Key-Homomorphic Signatures

In the following let $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a signature scheme and the secret and public key elements live in groups $(\mathbb{H}, +)$ and $(\mathbb{E}, \cdot)$, respectively. For these two groups we require that group operations, inversions, membership testing as well as sampling from the uniform distribution are efficient. We start with the notion of an efficiently computable homomorphism between secret keys and public keys.[6] Such a functionality has been implicitly used recently in [FKM+16] to define the notion of signatures with re-randomizable keys.

Definition 14 (Secret Key to Public Key Homomorphism). *A signature scheme Σ provides a secret key to public key homomorphism, if there exists an efficiently computable map $\mu : \mathbb{H} \to \mathbb{E}$ such that for all $\mathsf{sk}, \mathsf{sk}' \in \mathbb{H}$ it holds that $\mu(\mathsf{sk}+\mathsf{sk}') = \mu(\mathsf{sk}) \cdot \mu(\mathsf{sk}')$, and for all $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}$, it holds that $\mathsf{pk} = \mu(\mathsf{sk})$.*

As an illustrative example, think of the discrete logarithm setting, where we often have $\mathsf{sk} \xleftarrow{R} \mathbb{Z}_p$ and $\mathsf{pk} = g^{\mathsf{sk}}$ with g being the generator of some group $\mathbb{G}$ of prime order p. Here, it is obvious that there exists $\mu : \mathsf{sk} \mapsto g^{\mathsf{sk}}$ that is efficiently computable.

Now, we can introduce the first flavour of key-homomorphic signatures. We, thereby, take a similar path as Boneh et al. [BLMR13] in their work on key-homomorphic PRFs, and focus on the class of functions representing linear shifts. We think that limiting ourselves to linear shifts makes the applications of our framework much easier comprehensible, and, therefore, leave an extension to other suitable function classes as future work. We stress that linear shifts represent a finite set of functions, all with the same domain and range, and they usually depend on the public key of the signature scheme (which we will not make explicit). Moreover, they admit an efficient membership test, are efficiently samplable, and, are efficiently computable. Note that since we are talking of shifts, a function can be viewed as a "shift amount" $\Delta \in \mathbb{H}$.

Definition 15 together with the adaptability of signatures (Definition 16) or perfect adaption (Definition 17) are inspired by key-homomorphic encryption schemes [AHI11].

Definition 15 (Key-Homomorphic Signatures). *A signature scheme is called key-homomorphic, if it provides a secret key to public key homomorphism and an additional PPT algorithm* Adapt*, defined as:*

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: *Takes a public key* pk*, a message* m*, a signature* σ*, and a function* Δ *as input, and outputs a public key* pk' *and a signature* σ'*,*

such that for all $\Delta \in \mathbb{H}$ and all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, all messages $m \in \mathcal{M}$ and all σ with $\mathsf{Verify}(\mathsf{pk}, m, \sigma) = 1$ and $(\mathsf{pk}', \sigma') \leftarrow \mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$ it holds that

$$\mathsf{Pr}[\mathsf{Verify}(\mathsf{pk}', m, \sigma') = 1] = 1 \quad \wedge \quad \mathsf{pk}' = \mu(\Delta) \cdot \mathsf{pk}.$$

[6] This is analogous to the use in context of bounded-collusion identity-based encryption (IBE) in [TW14].

An interesting property in the context of key-homomorphic signatures is whether adapted signatures look like freshly generated signatures. Therefore, we introduce two different flavours of such a notion, inspired by the context hiding notion for P-homomorphic signatures [ABC+12, ALP12] as well as the adaptability notion for equivalence class signatures [HS14] from [FHS15].

Definition 16 (Adaptability of Signatures). *A key-homomorphic signature scheme provides adaptability of signatures, if for every $\kappa \in \mathbb{N}$ and every message $m \in \mathcal{M}$, it holds that*

$$[(\mathsf{sk}, \mathsf{pk}), \mathsf{Adapt}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m), \Delta)] ,$$

where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, $\Delta \xleftarrow{R} \mathbb{H}$, *and*

$$[(\mathsf{sk}, \mu(\mathsf{sk})), (\mu(\mathsf{sk}) \cdot \mu(\Delta), \mathsf{Sign}(\mathsf{sk} + \Delta, m)))] ,$$

where $\mathsf{sk} \xleftarrow{R} \mathbb{H}$, $\Delta \xleftarrow{R} \mathbb{H}$, *are identically distributed.*

Remark 1. Kiltz et al. [KMP16] have recently used a notion related to Definition 16 (denoted as random self-reducibility) in context of canonical identification schemes. We observe that when turning canonical identification schemes into signature schemes in the random oracle model (ROM) using the Fiat-Shamir heuristic [FS86], every random-self-reducible scheme also satisfies adaptability. However, the converse is not true as our notion covers a broader class of signature schemes, and in particular including schemes without random oracles as illustrated in Section 4.

Thus, the examples of random-self-reducible canonical identification schemes given in [KMP16] directly yield examples of adaptable signatures in the ROM. In Section 4 we explicitly show that the Schnorr signatures [Sch91] scheme, a signature scheme due to Katz and Wang [KW03, GJKW07] and the Guillou-Quisquater (GQ) signature scheme [GQ88] are adaptable according to the definition above.

An even stronger notion for the indistinguishability of fresh signatures and adapted signatures on the same message is achieved when requiring the distributions to be indistinguishable *even* when the initial signature used in Adapt *is known*. All schemes that satisfy this stronger notion (stated below) also satisfy Definition 16.

Definition 17 (Perfect Adaption). *A key-homomorphic signature scheme provides perfect adaption, if for every $\kappa \in \mathbb{N}$, every message $m \in \mathcal{M}$, it holds that*

$$[\sigma, (\mathsf{sk}, \mathsf{pk}), \mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)] ,$$

where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$, $\Delta \xleftarrow{R} \mathbb{H}$, *and*

$$[\sigma, (\mathsf{sk}, \mu(\mathsf{sk})), (\mu(\mathsf{sk}) \cdot \mu(\Delta), \mathsf{Sign}(\mathsf{sk} + \Delta, m)))] ,$$

where $\mathsf{sk} \xleftarrow{R} \mathbb{H}$, $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$, $\Delta \xleftarrow{R} \mathbb{H}$, *are identically distributed.*

One immediately sees that signatures from random-self-reducible canonical identification schemes, and, thus, Schnorr signatures, GQ signatures, as well as Katz-Wang signatures, do not satisfy Definition 17 as the commitment sent in the first phase remains fixed (cf. Section 4 for details). However, we note that there are various existing schemes that satisfy Definition 17. For example, BLS signatures [BLS04], the recent re-randomizable scheme by Pointcheval and Sanders [PS16], a variant of the well known Waters' signatures [Wat05], and some structure-preserving signature (SPS) schemes[7] (cf. Section 4 for a formal treatment of these schemes).

When looking at Definition 15, one could ask whether it is possible to replace Δ in the Adapt algorithm with its public key $\mu(\Delta)$. However, it is easily seen that the existence of such an algorithm contradicts even the weakest security guarantees the underlying signature scheme would need to provide, i.e., universal unforgeability under no-message attacks ($\mathsf{UUF\text{-}NMA}$ security).

Lemma 1. *There cannot be an* $\mathsf{UUF\text{-}NMA}$ *secure key-homomorphic signature scheme* Σ *for which there exists a modified PPT algorithm* Adapt' *taking* $\mu(\Delta)$ *instead of* Δ *that still satisfies Definition 15.*

Proof. We prove this by showing that any such scheme implies an adversary against $\mathsf{UUF\text{-}NMA}$ security of Σ. Let us assume that an $\mathsf{UUF\text{-}NMA}$ challenger provides a public key $\mathsf{pk}^\star$ and a target message $m^\star$. Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$ being compatible with public key $\mathsf{pk}^\star$, compute $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m^\star)$, then compute $\mathsf{pk}' \leftarrow \mathsf{pk}^\star \cdot \mathsf{pk}^{-1}$ and obtain a forgery $\sigma^\star$ for message $m^\star$ under the target public key $\mathsf{pk}^\star$ by running $(\sigma^\star, \mathsf{pk}^\star) \leftarrow \mathsf{Adapt}(\mathsf{pk}, m^\star, \sigma, \mathsf{pk}')$. □

Now, we move to a definition that covers key-homomorphic signatures where the adaption of a *set of* signatures, each to the same message, to a signature for the same message under a combined public key does not even require the knowledge of the relation between the secret signing keys.

Definition 18 (Publicly Key-Homomorphic Signatures). *A signature scheme is called publicly key-homomorphic, if it provides a secret key to public key homomorphism and an additional PPT algorithm* $\mathsf{Combine}$*, defined as:*

$\mathsf{Combine}((\mathsf{pk}_i)_{i=1}^n, m, (\sigma_i)_{i=1}^n)$: *Takes public keys* $(\mathsf{pk}_i)_{i\in[n]}$*, a message* m*, signatures* $(\sigma_i)_{i\in[n]}$ *as input, and outputs a public key* $\hat{\mathsf{pk}}$ *and a signature* $\hat{\sigma}$*,*

such that for all $n > 1$*, all* $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(1^\kappa))_{i=1}^n$*, all messages* $m \in \mathcal{M}$ *and all* $(\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m))_{i\in[n]}$ *and* $(\hat{\mathsf{pk}}, \hat{\sigma}) \leftarrow \mathsf{Combine}((\mathsf{pk}_i)_{i=1}^n, m, (\sigma_i)_{i=1}^n)$ *it*

[7] SPS [AFG+10] are signatures defined over two groups $\mathbb{G}_1$ and $\mathbb{G}_2$, equipped with a bilinear map (pairing), and messages are vectors of group elements (from either $\mathbb{G}_1$ or, $\mathbb{G}_2$, or both). Public keys and signatures also consist of group elements only and signatures are verified by deciding group membership of their elements and evaluating the pairing on elements from the public key, the message and the signature. They are an important tool for protocol design due to their interoperability with the NIZK proof system by Groth and Sahai [GS08].

holds that

$$\hat{\mathsf{pk}} = \prod_{i=1}^{n} \mathsf{pk}_i \quad \wedge \quad \mathsf{Pr}[\mathsf{Verify}(\hat{\mathsf{pk}}, m, \hat{\sigma}) = 1] = 1.$$

Analogously to Definitions 16 and 17, one can define indistinguishability of fresh and combined signatures for publicly key-homomorphic signatures.

Definition 19 (Public Adaptability of Signatures). *A publicly key-homomorphic signature scheme provides public adaptability of signatures, if for every $\kappa \in \mathbb{N}$, every $n \in \mathsf{poly}(\kappa)$, and every message $m \in \mathcal{M}$, it holds that*

$$\big[(\mathsf{sk}_i, \mathsf{pk}_i)_{i\in[n]}, \mathsf{Combine}((\mathsf{pk}_i)_{i\in[n]}, m, (\mathsf{Sign}(\mathsf{sk}_i, m))_{i\in[n]})\big],$$

where $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(1^\kappa))_{i\in[n]}$, and

$$\big[(\mathsf{sk}_i, \mathsf{pk}_i)_{i\in[n]}, \big(\textstyle\prod_{i\in[n]} \mathsf{pk}_i, \mathsf{Sign}(\sum_{i\in[n]} \mathsf{sk}_i, m)\big)\big],$$

where $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(1^\kappa))_{i\in[n]}$, are identically distributed.

Definition 20 (Perfect Public Adaptability of Signatures). *A publicly key-homomorphic signature scheme provides perfect public adaptability of signatures, if for every $\kappa \in \mathbb{N}$, every $n \in \mathsf{poly}(\kappa)$, and every message $m \in \mathcal{M}$, it holds that*

$$\big[(\mathsf{sk}_i, \mathsf{pk}_i, \sigma_i)_{i\in[n]}, \mathsf{Combine}((\mathsf{pk}_i)_{i\in[n]}, m, (\sigma_i)_{i\in[n]})\big],$$

where $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(1^\kappa), \sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m))_{i\in[n]}$, and

$$\big[(\mathsf{sk}_i, \mathsf{pk}_i, \sigma_i)_{i\in[n]}, \big(\textstyle\prod_{i\in[n]} \mathsf{pk}_i, \mathsf{Sign}(\sum_{i\in[n]} \mathsf{sk}_i, m)\big)\big],$$

where $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(1^\kappa), \sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m))_{i\in[n]}$, are identically distributed.

We want to mention that Definitions 18, 19 and 20 are, for instance, satisfied by BLS signatures or Waters' signatures with shared Waters' hash parameters (cf. [LOS+06]) (cf. Section 4 for a more formal treatment).

4 Overview of Key-Homomorphic Schemes

In this section we present an evaluation of (a selection of) existing signature schemes providing key-homomorphic properties which we compactly subsume in Table 1 and present subsequently.[8]

[8] While our focus is on signature schemes in classic algebraic settings, it is clearly also interesting to look at instantiations of signature schemes in other settings regarding their key-homomorphic properties. A prime example in this context is the lattice setting. Unfortunately, we are not aware of any classical lattice-based signatures scheme (e.g., hash-the-sign signatures [GPV08] or Fiat-Shamir transformed identification schemes [Lyu08]) which exhibits key-homomorphic properties that make it at least adaptable. Thus lattice-based schemes do not seem suitable for our applications. Nevertheless, we consider it as an interesting future work to study lattice-based signatures, or, more generally, the entire zoo of post-quantum signature schemes with respect to key-homomorphisms.

Scheme	A	PA	PPuA	$\mathbb{H}$	$\mathbb{E}$	PoK(Δ)	**Setting**
Schnorr [Sch91]	✓	×	×	$\mathbb{Z}_p$	$\mathbb{G}$	$\mathbb{Z}_p$	(EC)DL
BLS [BLS04]	✓	✓	✓	$\mathbb{Z}_p$	$\mathbb{G}_1$	$\mathbb{Z}_p$	BG
Katz-Wang [KW03, GJKW07]	✓	×	×	$\mathbb{Z}_p$	$\mathbb{G} \times \mathbb{G}$	$\mathbb{Z}_p$	(EC)DL
GQ [GQ88]	✓	×	×	$\mathbb{Z}_N^*$	$\mathbb{Z}_N^*$	$\mathbb{Z}_N^*$	RSA
Waters [Wat05, BFG13]	✓	✓	✓	$\{(h^x, \tilde{g}^x) \mid e(h^x, \tilde{g}) = e(h, \tilde{g}^x), x \in \mathbb{Z}_p,\ h \in \mathbb{G}_1,\ \tilde{g} \in \mathbb{G}_2\}$	$\mathbb{G}_2$	$\mathbb{G}_1$	BG
PS [PS16]	✓	✓	×	$\mathbb{Z}_p \times \mathbb{Z}_p$	$\mathbb{G}_2 \times \mathbb{G}_2$	$\mathbb{Z}_p \times \mathbb{Z}_p$	BG
AGOT [AGOT14]	✓	✓	×	$\mathbb{Z}_p \times \mathbb{Z}_p$	$\mathbb{G}_1 \times \mathbb{G}_1$	$\mathbb{Z}_p \times \mathbb{Z}_p$	BG
Ghadafi (Gha) [Gha16]	✓	✓	×	$\mathbb{Z}_p \times \mathbb{Z}_p$	$\mathbb{G}_2 \times \mathbb{G}_2$	$\mathbb{Z}_p \times \mathbb{Z}_p$	BG

Table 1. Examples of existing signature schemes with key-homomorphic properties. All schemes admit a key-homomorphism. Legend: A . . . adaptable, PA . . . perfectly adaptable, PPuA . . . perfectly publicly adaptable, Dom(Δ) . . . domain where the shift amounts live in, PoK(Δ) . . . domain of elements required to prove knowledge of shift amount. (EC)DL denotes (elliptic curve) discrete logarithm hard groups, RSA denotes RSA groups and BG denotes bilinear groups.

Below we present the schemes discussed in Table 1. Therefore let BGGen be a bilinear group generator which on input of a security parameter 1^κ and a type parameter $\mathsf{t} \in \{1,2,3\}$ outputs a bilinear group description BG. If $\mathsf{t} = 2$, BG is defined as $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, e, g, \tilde{g}, \psi)$, where $\mathbb{G}_1 = \langle g \rangle, \mathbb{G}_2 = \langle \tilde{g} \rangle$, and $\mathbb{G}_T$ are three groups of prime order p with $\kappa = \log_2 p$, e is a bilinear map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, and ψ is an isomorphism $\mathbb{G}_2 \to \mathbb{G}_1$. If $\mathsf{t} = 3$ the isomorphism ψ is missing. If $\mathsf{t} = 1$ we have that $\mathbb{G}_1 = \mathbb{G}_2$ denoted as $\mathbb{G}$. Note that for all the schemes we consider in this section, when given a signature σ for message m under pk, if $\mathsf{Verify}(\mathsf{pk}, m, \sigma) = 1$ then it holds that $\sigma \in \{\mathsf{Sign}(\mathsf{sk}, m)\}$.

4.1 Schnorr Signatures [Sch91]

In Scheme 1 we recall the Schnorr signature scheme.

$\mathsf{PGen}(1^\kappa)$: Choose a group $\mathbb{G}$ of prime order p with $\kappa = \log_2 p$, an elements $g \xleftarrow{R} \mathbb{G}$, and a hash function $H : \mathbb{G} \times \mathcal{M} \to \{0,1\}^n$ uniformly at random from hash function family $\{H_k\}_k$. Set and return $\mathsf{PP} \leftarrow (\mathbb{G}, g, H)$.

$\mathsf{KeyGen}(\mathsf{PP})$: Parse PP as $(\mathbb{G}, g, H)$, choose $x \xleftarrow{R} \mathbb{Z}_p$, set $\mathsf{pk} \leftarrow g^x$, $\mathsf{sk} \leftarrow x$ and output $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as $x \in \mathbb{Z}_p^*$, choose $r \xleftarrow{R} \mathbb{Z}_p$, compute $R \leftarrow g^r$, $c \leftarrow H(R, m)$, $y \leftarrow r + x \cdot c \bmod p$, and output $\sigma \leftarrow (c, y)$

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as g^x and σ as (c, y), verify whether $c = H((g^x)^{-c} g^y, m)$ and output 1 if so and 0 otherwise.

Scheme 1: Schnorr Signatures

Lemma 2. *Schnorr signatures are adaptable according to Definition 16.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the perfect adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: Let $\Delta \in \mathbb{Z}_p$ and $\mathsf{pk} = g^x$. Return (pk', σ'), where $\mathsf{pk}' \leftarrow g^x \cdot g^\Delta$ and $\sigma' \leftarrow (c, y')$ with $y' \leftarrow y + c \cdot \Delta \bmod p$.

It is immediate that adapted signatures are identical to fresh signatures under $\mathsf{pk}' = g^{x+\Delta}$ as long as the initial signature is unknown. □

4.2 Guillou-Quisquater [GQ88]

In Scheme 2 we recall the GQ signature scheme.

Lemma 3. *GQ signatures are adaptable according to Definition 16.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the perfect adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: Let $\Delta \in \mathbb{Z}_N^*$ and $\mathsf{pk} = X$. Return (pk', σ'), where $\mathsf{pk}' \leftarrow X \cdot \Delta^e \bmod N$ and $\sigma' \leftarrow (c, y')$ with $y' \leftarrow y \cdot \Delta^c \bmod N$.

It is immediate that adapted signatures are identical to fresh signatures under $\mathsf{pk}' = X \cdot \Delta^e \bmod N$ as long as the initial signature is unknown. □

$\mathsf{PGen}(1^\kappa)$: Sample two $\kappa/2$-bit primes p, q, $p \neq q$, set $N = pq$, select an odd prime $e < \varphi(N)$ and a hash function $H : \mathbb{Z}_N \times \mathcal{M} \to \mathbb{Z}_e$ uniformly at random from hash function family $\{H_k\}_k$. Set and return $\mathsf{PP} \leftarrow (N, e)$.

$\mathsf{KeyGen}(\mathsf{PP})$: Parse PP as (N, e), choose $x \xleftarrow{R} \mathbb{Z}_N^*$, compute $X \leftarrow x^e \mod N$ set $\mathsf{pk} \leftarrow X$, $\mathsf{sk} \leftarrow x$ and output $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as $x \in \mathbb{Z}_N^*$, choose $r \xleftarrow{R} \mathbb{Z}_N^*$, compute $R \leftarrow r^e \bmod N$, $c \leftarrow H(R, m)$, $y \leftarrow x^c \cdot r \bmod N$, and output $\sigma \leftarrow (c, y)$

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as X and σ as (c, y), verify whether $R = y^e \cdot X^{-H(R,m)}$, $c = H(R, m)$, $R \in \mathbb{Z}_N^*$ and output 1 if so and 0 otherwise.

Scheme 2: Guillou-Quisquater Signatures

4.3 BLS Signatures [BLS04]

In Scheme 3 we recall BLS signatures in a Type-3 setting (cf. [CHKM10] for a treatment of security of this BLS variant). We stress that the properties which we discuss below are equally valid for the original BLS scheme in [BLS04] instantiated in a Type-2 setting.

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 3)$, choose a hash function $H : \mathcal{M} \to \mathbb{G}_1$ uniformly at random from hash function family $\{H_k\}_k$, set $\mathsf{PP} \leftarrow (\mathsf{BG}, H)$.

$\mathsf{KeyGen}(\mathsf{PP})$: Parse PP as (BG, H), choose $x \xleftarrow{R} \mathbb{Z}_p$, set $\mathsf{pk} \leftarrow \tilde{g}^x$, $\mathsf{sk} \leftarrow x$, and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as $x \in \mathbb{Z}_p^*$ and return $\sigma \leftarrow H(m)^x$.

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as $\tilde{g}^x$, verify whether $e(H(m), \tilde{g}^x) = e(\sigma, \tilde{g})$ and return 1 if so and 0 otherwise.

Scheme 3: Type-3 BLS Signatures

Lemma 4. *BLS signatures are perfectly adaptable according to Definition 17.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the perfect adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: Let $\Delta \in \mathbb{Z}_p$ and $\mathsf{pk} = \tilde{g}^x$. Return (pk', σ'), where $\mathsf{pk}' \leftarrow \tilde{g}^x \cdot \tilde{g}^\Delta$ and $\sigma' \leftarrow \sigma \cdot H(m)^\Delta$.

It is immediate that adapted signatures are identical to fresh signatures under $\mathsf{pk}' = \tilde{g}^{x+\Delta}$. □

Lemma 5. *BLS signatures are perfectly publicly adaptable according to Definition 20.*

Proof. We prove the lemma above by presenting a suitable $\mathsf{Combine}$ algorithm.

$\mathsf{Combine}((\mathsf{pk}_i)_{i=1}^n, m, (\sigma_i)_{i=1}^n)$: Let $\mathsf{pk}_i = \tilde{g}^{x_i}$. Run $\hat{\mathsf{pk}} \leftarrow \prod_{i=1}^n \tilde{g}^{x_i}$, and $\hat{\sigma} \leftarrow \prod_{i=1}^n \sigma_i$ and return $\hat{\mathsf{pk}}$ and $\hat{\sigma}$. □

4.4 Katz-Wang Signatures [KW03, GJKW07]

Katz and Wang in [KW03] presented a signature scheme that enjoys a tight security reduction to the DDH problem. Basically, the public key of the scheme

represents a Diffie-Hellman (DH) tuple and the signature is a non-interactive zero-knowledge proof (obtained using the Fiat-Shamir heuristic) that the public key indeed forms a DH tuple. We present the version from [GJKW07] (Section 4) in Scheme 4, which in contrast to the original one in [KW03] does not include the statement (public key) in the Fiat-Shamir transform.[9]

$\mathsf{PGen}(1^\kappa)$: Choose a group $\mathbb{G}$ of prime order p with $\kappa = \log_2 p$, two elements $g, h \xleftarrow{R} \mathbb{G}$ and a hash function $H : \mathbb{G} \times \mathbb{G} \times \mathcal{M} \to \{0,1\}^n$ uniformly at random from hash function family $\{H_k\}_k$. Set and return $\mathsf{pp} \leftarrow (\mathbb{G}, g, h, H)$.

$\mathsf{KeyGen}(\mathsf{pp})$: Parse pp as $(\mathbb{G}, g, h, H)$, choose $x \xleftarrow{R} \mathbb{Z}_p$, set $\mathsf{pk} \leftarrow (g^x, h^x)$, $\mathsf{sk} \leftarrow x$, and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as $x \in \mathbb{Z}_p^*$, choose $r \xleftarrow{R} \mathbb{Z}_p^*$, set $A \leftarrow g^r$, $B \leftarrow h^r$, $c \leftarrow H(A, B, m)$, compute $s \leftarrow cx + r \bmod p$ and return $\sigma \leftarrow (c, s)$.

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as (y_1, y_2) and σ as (c, s) with $c \in \{0,1\}^n$ and $s \in \mathbb{Z}_p$. Compute $A \leftarrow g^s y_1^{-c}$, $B \leftarrow h^s y_2^{-c}$ and return 1 if $c = H(A, B, m)$ and 0 otherwise.

Scheme 4: Katz-Wang Signatures.

Lemma 6. *Katz-Wang signatures are adaptable according to Definition 16.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: Let $\Delta \in \mathbb{Z}_p$, $\mathsf{pk} = (y_1, y_2)$ and $\sigma = (c, s)$. Return (pk', σ'), where $\mathsf{pk}' \leftarrow (y_1 \cdot g^\Delta, y_2 \cdot h^\Delta)$ and $\sigma' \leftarrow (c, s + c\Delta \pmod p)$.

It is immediate that adapted signatures are identical to fresh signatures under $\mathsf{pk}' = (y_1 \cdot g^\Delta, y_2 \cdot h^\Delta)$ as long as the initial signature is unknown. □

4.5 Waters' Signatures [Wat05]

Below we recall Waters' signatures with shared hashing parameters in the Type-3 bilinear group setting as used in [BFG13] (a similar variant is presented in [CHKM10]). We note that for Waters' signatures without shared hash parameters [Wat05] it seems to be impossible to define an Adapt algorithm satisfying Definition 15.

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 3)$, choose $U = (h, u_0, \ldots u_n) \xleftarrow{R} \mathbb{G}_1^{k+1}$, and define $H : \mathcal{M} \to \mathbb{G}_1$ as $H(m) := u_0 \cdot \prod_{i=1}^n u_i^{m_i}$, where $\mathcal{M} = \{0,1\}^n$. Set $\mathsf{pp} \leftarrow (\mathsf{BG}, U, H)$.

$\mathsf{KeyGen}(\mathsf{pp})$: Parse pp as (BG, U, H), choose $x \xleftarrow{R} \mathbb{Z}_p$, set $\mathsf{pk} \leftarrow \tilde{g}^x$, $\mathsf{sk} \leftarrow (h^x, \tilde{g}^x)$, and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as $(h^x, \tilde{g}^x)$ with $x \in \mathbb{Z}_p^*$, choose $r \xleftarrow{R} \mathbb{Z}_p^*$, set $\alpha \leftarrow h^x \cdot H(m)^r$, $\beta \leftarrow \tilde{g}^r, \gamma \leftarrow g^r$ and return $\sigma \leftarrow (\alpha, \beta, \gamma)$.

$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as $\tilde{g}^x$ and σ as (α, β, γ). Verify whether $e(\alpha, \tilde{g}) = e(h, \tilde{g}^x) \cdot e(H(m), \beta) \ \wedge \ e(\gamma, \tilde{g}) = e(g, \beta)$ and return 1 if it holds and 0 otherwise.

Scheme 5: Waters' Signatures with Shared Hash Parameters

[9] In case the statement is included in the Fiat-Shamir transform, then the scheme is clearly not adaptable.

Lemma 7. *Waters' signatures with shared hash parameters are perfectly adaptable according to Definition 17.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the perfect adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: Let $\Delta = (\Delta_1, \Delta_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ with $e(\Delta_1, \tilde{g}) = e(h, \Delta_2)$ and let $\sigma = (\alpha, \beta, \gamma)$, and $\mathsf{pk} = \tilde{g}^x$. Choose $r' \xleftarrow{R} \mathbb{Z}_p$, and return (pk', σ'), where $\mathsf{pk}' \leftarrow \tilde{g}^x \cdot \Delta_2$ and $\sigma' \leftarrow (\alpha \cdot \Delta_1 \cdot H(m)^{r'}, \beta \cdot \tilde{g}^{r'}, \gamma \cdot g^{r'})$.

Signatures output by Adapt are identically distributed as fresh signatures under randomness $r + r'$ und key $\mathsf{pk} = \tilde{g}^x \cdot \Delta_2$, which proves the lemma. □

Note that the form of the secret key in the scheme above is favourable regarding the extractability properties of the Groth-Sahai proof system. Observe that Δ_2 is implicitly given by the difference of the public keys and thus one only needs to prove knowledge of a *single* group element being Δ_2.

Lemma 8. *Waters' signatures are perfectly publicly adaptable according to Definition 20.*

Proof. We prove the lemma above by presenting a suitable $\mathsf{Combine}$ algorithm.

$\mathsf{Combine}((\mathsf{pk}_i)_{i=1}^n, m, (\sigma_i)_{i=1}^n)$: Let $\sigma_i = (\alpha_i, \beta_i, \gamma_i)$ and $\mathsf{pk}_i = \tilde{g}^{x_i}$. Run $\hat{\mathsf{pk}} \leftarrow \prod_{i=1}^n \tilde{g}^{x_i}$, $r' \xleftarrow{R} \mathbb{Z}_p^*$ and $\hat{\sigma} \leftarrow (\prod_{i=1}^n \alpha_i \cdot H(m)^{r'}, \prod_{i=1}^n \beta_i \cdot \tilde{g}^{r'}, \prod_{i=1}^n \gamma_i \cdot g^{r'})$ and return $\hat{\mathsf{pk}}$ and $\hat{\sigma}$. □

4.6 PS Signatures [PS16]

In Scheme 6 we recall a recent signature scheme from [PS16], which provides perfect adaption, but is not publicly key-homomorphic.

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 3)$ set $\mathsf{pp} \leftarrow \mathsf{BG}$.
$\mathsf{KeyGen}(\mathsf{pp})$: Parse pp as BG, choose $x, y \xleftarrow{R} \mathbb{Z}_p$, compute $\tilde{X} \leftarrow \tilde{g}^x$, $\tilde{Y} \leftarrow \tilde{g}^y$ and set $\mathsf{pk} \leftarrow (\tilde{X}, \tilde{Y})$, $\mathsf{sk} \leftarrow (x, y)$, and return $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as (x, y) with $x, y \in \mathbb{Z}_p^*$, choose $h \xleftarrow{R} \mathbb{G}_1^*$ and return $\sigma \leftarrow (h, h^{x+y \cdot m})$.
$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as $(\tilde{X}, \tilde{Y})$ and σ as (σ_1, σ_2). Check whether $\sigma_1 \neq 1_{\mathbb{G}_1}$ and $e(\sigma_1, \tilde{X} \cdot \tilde{Y}^m) = e(\sigma_2, \tilde{g})$ holds. If both checks hold return 1 and 0 otherwise.

Scheme 6: PS Signatures

Lemma 9. *PS signatures are perfectly adaptable according to Definition 17.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the perfect adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: Parse pk as $(\tilde{X}, \tilde{Y})$, σ as (σ_1, σ_2) and Δ as $(\Delta_1, \Delta_2) \in \mathbb{Z}_p^2$ and choose $r \xleftarrow{R} \mathbb{Z}_p$. Compute $\mathsf{pk}' \leftarrow (\tilde{X} \cdot \tilde{g}^{\Delta_1}, \tilde{Y} \cdot \tilde{g}^{\Delta_2})$ and $\sigma' \leftarrow (\sigma_1^r, (\sigma_2 \cdot \sigma_1^{\Delta_1 + \Delta_2 m})^r)$ and return (pk', σ').

The key $\mathsf{pk}' = (\tilde{g}^{x+\Delta_1}, \tilde{g}^{y+\Delta_2})$ and $\sigma' = (h^r, (h^r)^{x+\Delta_1+m(y+\Delta_2)})$ output by the Adapt algorithm is identically distributed to a fresh signature under randomness h^r and pk'. □

It is easy to see, that PS signatures are, however, not publicly key-homomorphic as independently generated signatures are computed with respect to different bases h with unknown discrete logarithms. Consequently, there is no efficient means to obtain a succinct representation of $\hat{\sigma}$ that is suitable for Verify.

4.7 Randomizable SPS by Abe et al. [AGOT14]

Subsequently, in Scheme 7 we present a rerandomizable secure structure-preserving signature (SPS) scheme from Abe et al. [AGOT14].

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 2)$ and set $\mathsf{PP} \leftarrow \mathsf{BG}$.
$\mathsf{KeyGen}(\mathsf{PP})$: Parse PP as BG, choose $x, y \xleftarrow{R} \mathbb{Z}_p$, compute $X \leftarrow g^x$, $Y \leftarrow g^y$, set $\mathsf{pk} \leftarrow (X, Y)$, $\mathsf{sk} \leftarrow (x, y)$, and return $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{Sign}(\mathsf{sk}, m)$: Parse sk as (x, y) and $m \in \mathbb{G}_2$, choose $r \xleftarrow{R} \mathbb{Z}_p$, compute $\sigma_1 \leftarrow \tilde{g}^r$, $\sigma_2 \leftarrow m^x \cdot \tilde{g}^{r^2+y}$ and output $\sigma \leftarrow (\sigma_1, \sigma_2)$.
$\mathsf{Rand}(m, \sigma)$: Parse σ as (σ_1, σ_2), choose $\alpha \xleftarrow{R} \mathbb{Z}_p^*$, compute $\sigma_1' \leftarrow \sigma_1 \cdot \tilde{g}^\alpha$, $\sigma_2' \leftarrow \sigma_2 \cdot \sigma_1^{2\alpha} \cdot \tilde{g}^{\alpha^2}$ and output $\sigma' \leftarrow (\sigma_1', \sigma_2')$.
$\mathsf{Verify}(\mathsf{pk}, m, \sigma)$: Parse pk as (X, Y), σ as (σ_1, σ_2). Return 1 if $m, \sigma_1, \sigma_2 \in \mathbb{G}_2$ and $e(g, \sigma_2) = e(X, m) \cdot e(\psi(\sigma_1), \sigma_1) \cdot e(Y, \tilde{g})$. Otherwise return 0.

Scheme 7: Rerandomizable SPS

Lemma 10. *The SPS in Scheme 7 is perfectly adaptable according to Definition 17.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the perfect adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$: Parse pk as (X, Y), σ as (σ_1, σ_2) and Δ as $(\Delta_1, \Delta_2) \in \mathbb{Z}_p^2$. Compute $\mathsf{pk}' \leftarrow (X \cdot g^{\Delta_1}, Y \cdot g^{\Delta_2})$ and $\sigma_2' \leftarrow m^{\Delta_1} \cdot \tilde{g}^{\Delta_2}$ and $\sigma' \leftarrow \mathsf{Rand}(m, (\sigma_1, \sigma_2'))$ and return (pk', σ').

Adapted signatures are of the form $(\tilde{g}^{r+\alpha}, m^{x+\Delta_1} \cdot \tilde{g}^{(r+\alpha)^2+(y+\Delta_2)})$, and it is easy to see that they are identical to fresh signatures under $\mathsf{pk}' = (X \cdot g^{\Delta_1}, Y \cdot g^{\Delta_2})$ with respect to randomness $r + \alpha$. □

4.8 Ghadafi's Short SPS [Gha16]

As we will show below, the randomizable SPS by Ghadafi [Gha16] is perfectly adaptable. This scheme is defined for the Type-3 bilinear group setting and signs pairs of messages $(m, \tilde{n}) \in \mathbb{G}_1 \times \mathbb{G}_2$ having the same discrete logarithm with respect to bases $(g, \tilde{g}) \in \mathbb{G}_1 \times \mathbb{G}_2$ so that $e(m, \tilde{g}) = e(g, \tilde{n})$. Since we later require the randomization algorithm presented in [Gha16] in our proof for perfect adaptability, we explicitly include the Rand algorithm when we recall the scheme below.

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 3)$, and return $\mathsf{pp} \leftarrow \mathsf{BG}$.
$\mathsf{KeyGen}(\mathsf{pp})$: Parse pp as BG, choose $x, y \xleftarrow{R} \mathbb{Z}_p$ and set $\mathsf{pk} \leftarrow (\tilde{g}^x, \tilde{g}^y)$, $\mathsf{sk} \leftarrow (x, y)$, and return $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{Sign}(\mathsf{sk}, (m, \hat{n}))$: Parse sk as (x, y) with $x, y \in \mathbb{Z}_p$, choose $\alpha \xleftarrow{R} \mathbb{Z}_p^*$, set $a \leftarrow g^\alpha$, $b \leftarrow m^\alpha$, $c \leftarrow a^x \cdot b^y$. Return $\sigma = (a, b, c)$.
$\mathsf{Rand}(\mathsf{pk}, (m, \tilde{n}), \sigma)$: Parse σ as (a, b, c), select $r \xleftarrow{R} \mathbb{Z}_p$, set $a' \leftarrow a^r$, $b' \leftarrow b^r$, $c' \leftarrow c^r$, and return $\sigma' \leftarrow (a', b', c')$.
$\mathsf{Verify}(\mathsf{pk}, (m, \hat{n}), \sigma)$: Parse pk as $(\tilde{g}^x, \tilde{g}^y)$ and σ as (a, b, c). Return 1 if $a, b, c \in \mathbb{G}_1$, $a \neq 1_{\mathbb{G}_1}$, and the following checks hold, and 0 otherwise:

$$e(m, \tilde{g}) = e(g, \tilde{n}) \ \wedge \ e(a, \tilde{n}) = e(b, \tilde{g}) \ \wedge \ e(c, \tilde{g}) = e(a, \tilde{g}^x) e(b, \tilde{g}^y).$$

Scheme 8: Ghadafi's Short SPS

Lemma 11. *Ghadafi's short SPS are perfectly adaptable according to Definition 17.*

Proof. We prove the lemma above by presenting an Adapt algorithm satisfying the perfect adaptability notion.

$\mathsf{Adapt}(\mathsf{pk}, (m, \tilde{n}), \sigma, \Delta)$: Parse pk as $(\tilde{g}^x, \tilde{g}^y)$ and Δ as $(\Delta_1, \Delta_2) \in \mathbb{Z}_p^2$. Set $\mathsf{pk}' \leftarrow (\tilde{g}^x \tilde{g}^{\Delta_1}, \tilde{g}^y \tilde{g}^{\Delta_2})$, $\sigma' \leftarrow \mathsf{Rand}(\mathsf{pk}', (m, \tilde{n}), (a, b, c \cdot a^{\Delta_1} \cdot b^{\Delta_2}))$, and return pk' and σ'.

Adapted signatures are of the form $((g^\alpha)^r, (m^\alpha)^r, ((g^\alpha)^{x+\Delta_1} \cdot (m^\alpha)^{y+\Delta_2})^r)$, and it is easy to see that they are identical to fresh signatures under $\mathsf{pk}' = (\tilde{g}^x \tilde{g}^{\Delta_1}, \tilde{g}^y \tilde{g}^{\Delta_2})$ with respect to randomness $\alpha \cdot r$. □

5 Applications to Multiparty Signatures

In the following we show how the various key-homomorphic properties introduced in Section 3 can be used in a black-box way to obtain constructions of other interesting variants of signature schemes, namely ring signatures, (universal) designated verifier signatures as well as multisignatures.

5.1 Ring Signatures

Ring signature schemes [RST01] allow a member of an ad-hoc group $\mathcal{R}$ (the so called ring), defined by the member's public verification keys, to anonymously sign a message on behalf of $\mathcal{R}$. Given a ring signature and all public keys for $\mathcal{R}$, one can verify the validity of such a signature with respect to $\mathcal{R}$, but it is infeasible to identify the actual signer, i.e., the signer is unconditionally anonymous. Due to this anonymity feature ring signatures have proven to be an interesting tool for numerous applications, most notable for whistleblowing and recently for providing privacy of transactions in emerging technologies such as cryptocurrencies.[10] The two main lines of work in the design of ring signatures

[10] https://getmonero.org/resources/moneropedia/ringCT.html

target reducing the signature size or removing the requirement for random oracles (e.g., [DKNS04, CGS07, GK15]).

Our framework allows for constructions that do *not* require random oracles and the signature size depends on the design of the relation $\mathcal{R}$ below. Compared to existing generic frameworks (cf. [BKM09, BK10]), it provides an alternative, very simple, generic, and flexible framework to construct ring signatures.

We formally define ring signature schemes (adopting [BKM09]) and note that the model implicitly assumes knowledge of secret keys [RY07] as discussed in Section 2.

Definition 21. *A ring signature scheme* RS *is a tuple* $\mathsf{RS} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *of PPT algorithms, which are defined as follows.*

$\mathsf{Setup}(1^\kappa)$: *This algorithm takes as input a security parameter* κ *and outputs public parameters* pp.

$\mathsf{Gen}(\mathsf{pp})$: *This algorithm takes as input the public parameter* pp *and outputs a keypair* $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}_i, m, \mathcal{R})$: *This algorithm takes as input the public parameters* pp*, a secret key* sk_i*, a message* $m \in \mathcal{M}$ *and a ring* $\mathcal{R} = (\mathsf{pk}_j)_{j\in[n]}$ *of* n *public keys such that* $\mathsf{pk}_i \in \mathcal{R}$*. It outputs a signature* σ.

$\mathsf{Verify}(\mathsf{pp}, m, \sigma, \mathcal{R})$: *This algorithm takes as input the public parameters* pp*, a message* $m \in \mathcal{M}$*, a signature* σ *and a ring* $\mathcal{R}$*. It outputs a bit* $b \in \{0, 1\}$.

A secure ring signature scheme needs to be correct, unforgeable, and anonymous. While we omit the obvious correctness definition, we subsequently provide formal definitions for the remaining properties following [BKM09]. We note that Bender et al. in [BKM09] have formalized multiple variants of these properties, where we always use the strongest one.

Unforgeability requires that without any secret key sk_i that corresponds to a public key $\mathsf{pk}_i \in \mathcal{R}$, it is infeasible to produce valid signatures with respect to arbitrary such rings $\mathcal{R}$. Our unforgeability notion is the strongest notion defined in [BKM09] and is there called *unforgeability w.r.t. insider corruption.*

Definition 22 (Unforgeability). *A ring signature scheme provides unforgeability, if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\varepsilon(\cdot)$ *such that it holds that*

$$\mathsf{Pr}\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa),\\ \{(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{Gen}(\mathsf{pp})\}_{i\in[\mathsf{poly}(\kappa)]},\\ \mathcal{O} \leftarrow \{\mathsf{Sig}(\cdot,\cdot,\cdot), \mathsf{Key}(\cdot)\},\\ (m^\star, \sigma^\star, \mathcal{R}^\star) \leftarrow \mathcal{A}^{\mathcal{O}}(\{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]})\end{array} : \begin{array}{r}\mathsf{Verify}(m^\star, \sigma^\star, \mathcal{R}^\star) = 1 \ \wedge\\ (\cdot, m^\star, \mathcal{R}^\star) \notin \mathcal{Q}^{\mathsf{Sign}} \ \wedge\\ \mathcal{R}^\star \subseteq \{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]\setminus\mathcal{Q}^{\mathsf{Key}}}\end{array}\right] \leq \varepsilon(\kappa),$$

where $\mathsf{Sig}(i, m, \mathcal{R}) := \mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})$*,* Sig *returns* $\perp$ *if* $\mathsf{pk}_i \notin \mathcal{R} \ \vee \ i \notin [\mathsf{poly}(\kappa)]$*, and* $\mathcal{Q}^{\mathsf{Sig}}$ *records the queries to* Sig*. Furthermore,* $\mathsf{Key}(i)$ *returns* sk_i *and* $\mathcal{Q}^{\mathsf{Key}}$ *records the queries to* Key.

Anonymity requires that it is infeasible to tell which ring member produced a certain signature as long as there are at least two honest members in the ring. Our anonymity notion is the strongest notion defined in [BKM09] and is there called *anonymity against full key exposure.*

Definition 23 (Anonymity). *A ring signature scheme provides anonymity, if for all PPT adversaries $\mathcal{A}$ and for all polynomials $n(\cdot)$, there exists a negligible function $\varepsilon(\cdot)$ such that it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa),\\ \{(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{Gen}(\mathsf{pp})\}_{i\in[\mathsf{poly}(\kappa)]},\\ b \xleftarrow{R} \{0,1\},\ \mathcal{O} \leftarrow \{\mathsf{Sig}(\cdot,\cdot,\cdot)\},\\ (m, j_0, j_1, \mathcal{R}, \mathsf{st}) \leftarrow \mathcal{A}^{\mathcal{O}}(\{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]}),\\ \sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_{j_b}, m, \mathcal{R}),\\ b^\star \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{st}, \sigma, \{\mathsf{sk}_i\}_{i\in[\mathsf{poly}(\kappa)]})\end{array} : \begin{array}{r} b = b^\star \ \wedge\\ \{\mathsf{pk}_{j_0}, \mathsf{pk}_{j_1}\} \subseteq \mathcal{R}\end{array}\right] \leq 1/2 + \varepsilon(\kappa),$$

where $\mathsf{Sig}(i, m, \mathcal{R}) \coloneqq \mathsf{Sign}(\mathsf{sk}_i, m, \mathcal{R})$.

Our Construction. In Scheme 9 we present our black-box construction of ring signatures from any key-homomorphic EUF-CMA secure signature scheme Σ with adaptable signatures and any witness indistinguishable argument of knowledge system Π. The idea behind the scheme is as follows. A ring signature for message m with respect to ring $\mathcal{R}$ consists of a signature for $m||\mathcal{R}$ using Σ with a randomly generated key pair together with a proof of knowledge attesting the knowledge of the "shift amount" from the random public key to (at least) one of the public keys in $\mathcal{R}$.[11] Very briefly, unforgeability then holds because—given a valid ring signature—one can always extract a valid signature of one of the ring members. Anonymity holds because the witness indistinguishability of the argument system guarantees that signatures of different ring members are indistinguishable.

Upon signing, we need to prove knowledge of a witness for the following **NP** relation R.

$$((\mathsf{pk}, \mathsf{cpk}, \mathcal{R}), \mathsf{sk}') \in R \iff \exists\ \mathsf{pk}_i \in \mathcal{R} \cup \{\mathsf{cpk}\} \ :\ \mathsf{pk}_i = \mathsf{pk} \cdot \mu(\mathsf{sk}')$$

For the sake of compactness, we assume that the relation is implicitly defined by the scheme. One can obtain a straight forward instantiation by means of disjunctive proofs of knowledge [CDS94] (similar as it is done in many known constructions). Therefore one could use the following **NP** relation R.

$$((\mathsf{pk}, \mathsf{cpk}, \mathcal{R}), \mathsf{sk}') \in R \iff \left(\vee_{\mathsf{pk}_i \in \mathcal{R}}\ \mathsf{pk}_i = \mathsf{pk} \cdot \mu(\mathsf{sk}')\right) \ \vee\ \mathsf{cpk} = \mathsf{pk} \cdot \mu(\mathsf{sk}')$$

Using this approach, however, yields signatures of linear size. To reduce the signature size, one could, e.g., follow the approach of [DKNS04].

Related Subsequent Work. We want to remark that a technique similar to ours was recently also used in a construction of ring signatures in the standard model by Malavolta and Schröder [MS17]. In particular, they rely on the related concept of signatures with re-randomizable keys [FKM+16] to realize a similar functionality as we do. This allows them to achieve simplicity and efficiency gains comparable to ours. Compared to our work, they do not require a setup, but have to rely on a non-falsifiable knowledge of exponent assumption [Nao03]. Such assumption however are very strong and results relying on those assumption

[11] For technical reasons we need an additional public key cpk in the public parameters.

should rather be viewed as intermediate results [GK16]. In contrast, our framework can be instantiated under standard assumptions (using Waters' signatures) at the cost of relying on a setup.

$\mathsf{Setup}(1^\kappa)$: Run $\mathsf{crs} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$, $(\mathsf{csk}, \mathsf{cpk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, set $\mathsf{PP} \leftarrow (1^\kappa, \mathsf{crs}, \mathsf{cpk})$ and return PP.

$\mathsf{Gen}(\mathsf{PP})$: Run $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$ and return $(\mathsf{sk}_i, \mathsf{pk}_i)$.

$\mathsf{Sign}(\mathsf{PP}, \mathsf{sk}_i, m, \mathcal{R})$: Parse PP as $(1^\kappa, \mathsf{crs}, \mathsf{cpk})$ and return $\perp$ if $\mu(\mathsf{sk}_i) \notin \mathcal{R}$. Otherwise, return $\sigma \leftarrow (\delta, \mathsf{pk}, \pi)$, where

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa),\ \delta \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m||\mathcal{R}),\ \text{and}$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}, \mathsf{cpk}, \mathcal{R}), (\mathsf{sk}_i - \mathsf{sk})).$$

$\mathsf{Verify}(\mathsf{PP}, m, \sigma, \mathcal{R})$: Parse PP as $(1^\kappa, \mathsf{crs}, \mathsf{cpk})$ and σ as $(\delta, \mathsf{pk}, \pi)$ and return 1 if the following holds, and 0 otherwise:

$$\Sigma.\mathsf{Verify}(\mathsf{pk}, m||\mathcal{R}, \delta) = 1 \quad \wedge \quad \Pi.\mathsf{Verify}(\mathsf{crs}, (\mathsf{pk}, \mathsf{cpk}, \mathcal{R}), \pi) = 1.$$

Scheme 9: Black-Box Construction of Ring Signatures

Theorem 1. *If Σ is correct, EUF-CMA secure, and provides adaptability of signatures, Π is complete and witness indistinguishable proof of knowledge, then Scheme 9 is correct, unforgeable, and anonymous.*

We show that Theorem 1 holds by proving the subsequent lemmas.

Lemma 12. *If Σ is correct, and Π is complete, then Scheme 9 is correct.*

Lemma 12 follows from inspection and the proof is therefore omitted.

Lemma 13. *If Σ is EUF-CMA secure, and provides adaptability of signatures, and Π is witness indistinguishable, then Scheme 9 is unforgeable.*

Proof. We prove unforgeability using a sequence of games where we let $q_\mathsf{s} \leq \mathsf{poly}(\kappa)$ be the number of Sign queries.

Game 0: The original unforgeability game.

Game 1: As Game 0, but upon setup we store csk and simulate Sign using the following modified algorithm Sign', which additionally takes csk as input:

$\mathsf{Sign}'(\mathsf{PP}, \mathsf{sk}_i, m, \mathcal{R}, \boxed{\mathsf{csk}})$: Parse PP as $(1^\kappa, \mathsf{crs}, \mathsf{cpk})$ and return $\perp$ if $\mu(\mathsf{sk}_i) \notin \mathcal{R}$. Otherwise, return $\sigma \leftarrow (\delta, \mathsf{pk}, \pi)$, where

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa),\ \delta \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m||\mathcal{R}),\ \text{and}$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}, \mathsf{cpk}, \mathcal{R}), (\boxed{\mathsf{csk}} - \mathsf{sk})).$$

Transition - Game 0 $\rightarrow$ Game 1: A distinguisher between $\mathcal{D}^{0\rightarrow 1}$ is a distinguisher for adaptive witness indistinguishability of Π, i.e., $|\mathsf{Pr}[S_0] - \mathsf{Pr}[S_1]| \leq \varepsilon_{\mathsf{wi}}(\kappa)$.

Game 2: As Game 1, but instead of generating crs upon setup, we obtain $(\mathsf{crs}, \xi) \leftarrow \Pi.\mathsf{E}_1(1^\kappa)$ and store ξ.

Transition - Game 1 → Game 2: A distinguisher between Game 1 and 2 distinguishes an honest crs from an extraction crs, i.e., $|\mathsf{Pr}[S_1] - \mathsf{Pr}[S_2]| \leq \varepsilon_{\mathsf{e1}}(\kappa)$.

Game 3: As Game 2, but whenever the adversary outputs a forgery $(m^\star, \sigma^\star, \mathcal{R}^\star)$, where $\sigma^\star = (\delta^\star, \mathsf{pk}^\star, \pi^\star)$ we extract a witness $\mathsf{sk}' \leftarrow \Pi.\mathsf{E}_2(\mathsf{crs}, \xi, (\mathsf{pk}^\star, \mathsf{cpk}, \mathcal{R}^\star), \pi^\star)$ and abort if the extractor fails.

Transition - Game 2 → Game 3: Game 2 and Game 3 proceed identically, unless the extractor fails, i.e., $|\mathsf{Pr}[S_2] - \mathsf{Pr}[S_3]| \leq \varepsilon_{\mathsf{e2}}(\kappa)$.

Game 4: As Game 3, but we further modify Sign' as follows:

$\mathsf{Sign}'(\mathsf{PP}, \boxed{i}, m, \mathcal{R}, \mathsf{csk})$: Parse PP as $(1^\kappa, \mathsf{crs}, \mathsf{cpk})$ and return $\perp$ if $\mathsf{pk}_i \notin \mathcal{R}$. Otherwise, return $\sigma \leftarrow (\delta, \mathsf{pk}, \pi)$, where

$$\boxed{\mathsf{sk} \xleftarrow{R} \mathbb{H},\ \delta' \leftarrow \Sigma.\mathsf{Sign}(\mathsf{csk}, m||\mathcal{R})},$$
$$\boxed{(\mathsf{pk}, \delta) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{cpk}, m||\mathcal{R}, \delta', -\mathsf{sk})},\ \text{and}$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}, \mathsf{cpk}, \mathcal{R}), (\boxed{\mathsf{sk}})).$$

Transition - Game 3 → Game 4: Under adaptability of signatures, this game change is conceptual, i.e., $\mathsf{Pr}[S_3] = \mathsf{Pr}[S_4]$.

Game 5: As Game 4, but we abort whenever we extract an sk' so that $\mathsf{cpk} = \mathsf{pk} \cdot \mu(\mathsf{sk}')$.

Transition - Game 4 → Game 5: Game 4 and Game 5 proceed identical, unless abort event E_1 happens. For the sake of contradiction assume that E_1 occurs with non-negligible probability. Then we can engage with an $\mathsf{EUF\text{-}CMA}$ challenger $\mathcal{C}_\kappa^{\mathsf{f}}$ to obtain cpk upon setup and simulate Sign as follows:

$\mathsf{Sign}'(\mathsf{PP}, i, m, \mathcal{R}, \boxed{\perp})$: Parse PP as $(1^\kappa, \mathsf{crs}, \mathsf{cpk})$ and return $\perp$ if $\mathsf{pk}_i \notin \mathcal{R}$. Otherwise, return $\sigma \leftarrow (\delta, \mathsf{pk}, \pi)$, where

$$\mathsf{sk} \xleftarrow{R} \mathbb{H},\ \boxed{\delta' \leftarrow \mathcal{C}_{\mathsf{f}}^{\kappa}.\mathsf{Sign}(m||\mathcal{R})},$$
$$(\mathsf{pk}, \delta) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{cpk}, m||\mathcal{R}, \delta', -\mathsf{sk}),\ \text{and}$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}, \mathsf{cpk}, \mathcal{R}), (\mathsf{sk})).$$

Now, whenever E_1 happens, we use the forgery $(m^\star, \sigma^\star, \mathcal{R}^\star)$, where $\sigma^\star = (\delta^\star, \mathsf{pk}^\star, \pi^\star)$ to obtain $(\mathsf{cpk}, \delta) \leftarrow \mathsf{Adapt}(\mathsf{pk}, m^\star||\mathcal{R}^\star, \delta^\star, \mathsf{sk}')$ and return $(m^\star||\mathcal{R}^\star, \delta)$ as an $\mathsf{EUF\text{-}CMA}$ forgery to $\mathcal{C}_{\mathsf{f}}^{\kappa}$ with probability $\mathsf{Pr}[E_1]$. That is, $|\mathsf{Pr}[S_4] - \mathsf{Pr}[S_5]| \leq \epsilon_{\mathsf{f}}(\kappa)$.

Game 6: As Game 5, but we guess the index $i^\star$ the adversary will attack at the beginning of the game, and abort if our guess is wrong.

Transition - Game 5 → Game 6: The success probability in Game 5 is the same as in Game 6, unless our guess is wrong, i.e., $\mathsf{Pr}[S_6] = \frac{1}{\mathsf{poly}(\kappa)} \cdot \mathsf{Pr}[S_5]$.

Game 7: As Game 6, but instead of running KeyGen for user $i^\star$, we engage with an $\mathsf{EUF\text{-}CMA}$ challenger of Σ to obtain $\mathsf{pk}_{i^\star}$.

Transition - Game 6 → Game 7: This change is conceptual, i.e., $\mathsf{Pr}[S_6] = \mathsf{Pr}[S_7]$.

If the adversary outputs a forgery $(m^\star, \sigma^\star, \mathcal{R}^\star)$ in Game 7, we compute $(\mathsf{pk}_{i^\star}, \sigma_{i^\star}) \leftarrow \mathsf{Adapt}(\mathsf{pk}^\star, m^\star||\mathcal{R}^\star, \delta^\star, \mathsf{sk}')$ and return $(\sigma_{i^\star}, m^\star||\mathcal{R}^\star)$ as a valid forgery for Σ. That is, $\mathsf{Pr}[S_7] \leq \varepsilon_{\mathsf{f}}(\kappa)$ and we obtain $\mathsf{Pr}[S_0] \leq \mathsf{poly}(\kappa) \cdot \varepsilon_{\mathsf{f}}(\kappa) + \varepsilon_{\mathsf{wi}}(\kappa) + \varepsilon_{\mathsf{e1}}(\kappa) + \varepsilon_{\mathsf{e2}}(\kappa) + \varepsilon_{\mathsf{f}}(\kappa)$ as a bound for the success probability which concludes the proof. □

Lemma 14. *If Σ provides adaptability of signatures and Π is witness indistinguishable, then Scheme 9 is anonymous.*

Proof. We show that a simulation of the anonymity game for $b = 0$ is indistinguishable from a simulation of the anonymity game with $b = 1$.

Game 0: The anonymity game with $b = 0$.

Game 1: As Game 0, but instead of generating crs upon setup, we obtain crs from a witness indistinguishability challenger $\mathcal{C}_\kappa^{\mathsf{wi}}$ upon Setup.

Transition - Game 0 → Game 1: This change is conceptual, i.e., $\mathsf{Pr}[S_0] = \mathsf{Pr}[S_1]$.

Game 2: As Game 1, but instead of obtaining σ via Sign, we execute the following modified algorithm Sign', which, besides PP, m and $\mathcal{R}$, takes sk_0 and sk_1 as input:

$\mathsf{Sign}'(\mathsf{PP}, \mathsf{sk}_0, \mathsf{sk}_1, m, \mathcal{R})$: Parse PP as $(1^\kappa, \mathsf{crs})$ and return $\perp$ if $\mu(\mathsf{sk}_0) \notin \mathcal{R}$ $\boxed{\vee\ \mu(\mathsf{sk}_1) \notin \mathcal{R}}$. Otherwise, return $\sigma \leftarrow (\delta, \mathsf{pk}, \pi)$, where

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa),\ \delta \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m||\mathcal{R}),\ \text{and}$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}, \mathcal{R}), (\boxed{\mathsf{sk}_1} - \mathsf{sk})).$$

Transition - Game 1 → Game 2: A distinguisher between $\mathcal{D}^{1\to 2}$ is a distinguisher for adaptive witness indistinguishability of Π, i.e., $|\mathsf{Pr}[S_2] - \mathsf{Pr}[S_1]| \leq \varepsilon_{\mathsf{wi}}(\kappa)$.

Game 2 corresponds to the anonymity game for $b = 1$; $|\mathsf{Pr}[S_2] - \mathsf{Pr}[S_0]| \leq \varepsilon_{\mathsf{wi}}(\kappa)$, which proves the lemma. □

5.2 Universal Designated Verifier Signatures

In designated verifier signatures [JSI96] a signer chooses a designated verifier upon signing a message and, given this signature, only the designated verifier is convinced of its authenticity. The idea behind those constructions is to ensure that the designated verifier can "fake" signatures which are indistinguishable from signatures of the original signer. Universal designated verifier signatures (UDVS) [SBWP03] further extend this concept by introducing an additional party, which performs the designation process by converting a conventional signature to a designated-verifier one. There exists a significant body of work on UDVS, and, most notably, in [SS08] it was shown how to convert a large class of signature schemes to UDVS. Latter approach can be seen as related to our approach, yet they do not rely on key-homomorphisms and they only achieve weaker security guarantees.[12]

[12] We also note that [SS08] informally mention that their approach is also useful to construct what they call hierarchical ring signatures. However their paradigm is not useful to construct ring signatures as we did in the previous section.

While one can interpret designated verifier signatures as a special case of ring signatures where $|\mathcal{R}| = 2$, i.e., the ring is composed of the public keys of signer and designated verifier (as noted in [RST01, BKM09]), there seems to be no obvious black-box relation turning ring signatures into UDVS. Mainly, since UDVS require the functionality to convert standard signatures to designated verifier ones.[13]

To this end, we explicitly treat constructions of UDVS from key-homomorphic signatures subsequently. We start by recalling the security model from [SBWP03] including some notational adaptations and a strengthened version of the DV-unforgeability notion which we introduce here.

Definition 24. *A universal designated verifier signature scheme* UDVS *builds up on a conventional signature scheme* $\Sigma = (\mathsf{PGen}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *and additionally provides the PPT algorithms* $(\mathsf{DVGen}, \mathsf{Desig}, \mathsf{Sim}, \mathsf{DVerify})$*, which are defined as follows.*

$\mathsf{DVGen}(\mathsf{PP})$: *This algorithm takes the public parameters* PP *as input and generates and outputs a designated-verifier key pair* $(\mathsf{vsk}, \mathsf{vpk})$.

$\mathsf{Desig}(\mathsf{pk}, \mathsf{vpk}, m, \sigma)$: *This algorithm takes a signer public key* pk*, a designated-verifier public key* vpk*, a message* m*, and a valid signature* σ *as input, and outputs a designated-verifier signature* δ.

$\mathsf{Sim}(\mathsf{pk}, \mathsf{vsk}, m)$: *This algorithm takes a signer public key* pk*, a designated-verifier secret key* vsk*, and a message* m *as input, and outputs a designated-verifier signature* δ.

$\mathsf{DVerify}(\mathsf{pk}, \mathsf{vsk}, m, \delta)$: *This algorithm takes a signer public key* pk*, a designated-verifier secret key* vsk*, a message* m*, and a designated-verifier signature* δ *as input, and outputs a bit* $b \in \{0, 1\}$.

Subsequently we formally recall the security properties, where we omit the obvious correctness notion. For the remaining notions we largely follow [SBWP03, SS08].

DV-unforgeability captures the intuition that it should be infeasible to come up with valid designated verifier signatures where no corresponding original signature exists. Subsequently, we introduce a stronger variant of DV-unforgeability, which we term *simulation-sound DV-unforgeability*. This notion additionally provides the adversary with an oracle to simulate designated-verifier signatures on other messages for the targeted designated verifier. It is easy to see that our notion implies DV-unforgeability in the sense of [SBWP03].

Definition 25 (Simulation-Sound DV-Unforgeability). *An* UDVS *provides simu-lation-sound DV-unforgeability, if for all PPT adversaries* $\mathcal{A}$*, there exists a*

[13] We, however, note that an extension of the UDVS model to universal designated verifier *ring* signatures would be straight forward and also our scheme is extensible using the same techniques as in Scheme 9.

negligible function $\varepsilon(\cdot)$ such that it holds that

$$\Pr\left[\begin{array}{l}\mathsf{PP}\leftarrow\mathsf{PGen}(1^\kappa),\\(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KeyGen}(\mathsf{PP}),\\(\mathsf{vsk},\mathsf{vpk})\leftarrow\mathsf{DVGen}(\mathsf{PP}),\\\mathcal{O}\leftarrow\{\mathsf{Sig}(\mathsf{sk},\cdot),\ \mathsf{Vrfy}(\mathsf{pk},\mathsf{vsk},\cdot,\cdot),\\\mathsf{S}(\mathsf{pk},\mathsf{vsk},\cdot)\},\\(m^\star,\delta^\star)\leftarrow\mathcal{A}^{\mathcal{O}}(\mathsf{pk},\mathsf{vpk})\end{array}\ :\ \begin{array}{r}\mathsf{DVerify}(\mathsf{pk},\mathsf{vsk},m^\star,\delta^\star)=1\ \wedge\\ m^\star\notin\mathcal{Q}^{\mathsf{Sig}}\ \wedge\ m^\star\notin\mathcal{Q}^{\mathsf{Sim}}\end{array}\right]\leq\varepsilon(\kappa),$$

where $\mathsf{Sig}(\mathsf{sk},m):=\mathsf{Sign}(\mathsf{sk},m)$, $\mathsf{Vrfy}(\mathsf{pk},\mathsf{vsk},m,\delta):=\mathsf{DVerify}(\mathsf{pk},\mathsf{vsk},m,\delta)$, *and* $\mathsf{S}(\mathsf{pk},\mathsf{vsk},m):=\mathsf{Sim}(\mathsf{pk},\mathsf{vsk},m)$. *Furthermore, the environment keeps tracks of the messages queried to* Sig *and* S *via* $\mathcal{Q}^{\mathsf{Sig}}$ *and* $\mathcal{Q}^{\mathsf{Sim}}$, *respectively.*

Non-transferability privacy models the requirement that the designated verifier can simulate signatures which are indistinguishable from honestly designated signatures.

Definition 26 (Non-Transferability Privacy). *An* UDVS *provides non-transferability privacy, if for all PPT adversaries $\mathcal{A}$, there exists a negligible function $\varepsilon(\cdot)$ such that it holds that*

$$\Pr\left[\begin{array}{l}\mathsf{PP}\leftarrow\mathsf{PGen}(1^\kappa),\ (\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KeyGen}(\mathsf{PP}),\\b\xleftarrow{R}\{0,1\},\ \mathcal{O}\leftarrow\{\mathsf{Sig}(\mathsf{sk},\cdot),\mathsf{RKey}(\cdot,\cdot,\cdot)\},\\(m^\star,\mathsf{st})\leftarrow\mathcal{A}^{\mathcal{O}}(\mathsf{pk}),\ \sigma\leftarrow\mathsf{Sign}(\mathsf{sk},m^\star),\\b^\star\leftarrow\mathcal{A}^{\mathcal{O}\cup\{\mathsf{SoD}(\mathsf{pk},\cdot,m^\star,\sigma,b)\}}(\mathsf{st})\end{array}\ :\ \begin{array}{r}b=b^\star\ \wedge\\ m^\star\notin\mathcal{Q}^{\mathsf{Sig}}\end{array}\right]\leq 1/2+\varepsilon(\kappa),$$

where the oracles are defined as follows:

$\mathsf{Sig}(\mathsf{sk},m)$: *This oracle computes* $\sigma\leftarrow\mathsf{Sign}(\mathsf{sk},m)$ *and returns* σ.

$\mathsf{RKey}(i,\mathsf{vsk},\mathsf{vpk})$: *This oracle checks whether* $\mathtt{DVK}[i]\neq\perp$ *and returns* $\perp$ *if so. Otherwise, it checks whether* (vsk, vpk) *is a valid output of* DVGen *and sets* $\mathtt{DVK}[i]\leftarrow(\mathsf{vsk},\mathsf{vpk})$ *if so.*

$\mathsf{SoD}(\mathsf{pk},i,m,\sigma,b)$: *This oracle obtains* $(\mathsf{vsk},\mathsf{vpk})\leftarrow\mathtt{DVK}[i]$ *and returns* $\perp$ *if no entry for* i *exists. Then, if* $b=0$, *it computes* $\delta\leftarrow\mathsf{Sim}(\mathsf{pk},\mathsf{vsk},m)$, *and, if* $b=1$ *it computes* $\delta\leftarrow\mathsf{Desig}(\mathsf{pk},\mathsf{vpk},m,\sigma)$. *In the end it returns* δ. *This oracle can only be called once.*

Further, the environment maintains a list $\mathcal{Q}^{\mathsf{Sig}}$ *keeping track of the* Sig *queries.*

The notion above captures non-transferability privacy in the sense of [SS08]. This notion can be strengthened to what we call *strong non-transferability privacy* which allows multiple calls to SoD (as in [SBWP03]). While non-transferability privacy is often sufficient in practice, we will prove that our construction provides strong non-transferability privacy (clearly implying non-transferability privacy) to obtain the most general result.

Our Construction. In Scheme 10, we present our construction of UDVS from any key-homomorphic EUF-CMA secure Σ with perfect adaption of signatures, any witness indistinguishable argument of knowledge system Π, and any one-way

function f. Our construction uses the "OR-trick" [JSI96], known from DVS.[14] The basic idea is that the designation given a signature that verifies under pk, generates a fresh signing key pair $(\mathsf{sk}', \mathsf{pk}')$, adapts the signature to $\mathsf{pk} \cdot \mathsf{pk}'$ and then provides a proof that one either knows the secret key sk' or the preimage vsk of a one-way function $\mathsf{vpk} = f(\mathsf{vsk})$, where vpk represents the designated verifiers public key. More formally, upon computing designations and simulations of designated-verifier signatures, we require to prove knowledge of witnesses for the following **NP** relation R:

$$((\mathsf{pk}', \mathsf{vpk}), (\mathsf{sk}', \mathsf{vsk})) \in R \iff \mathsf{pk}' = \mu(\mathsf{sk}') \quad \vee \quad \mathsf{vpk} = f(\mathsf{vsk}).$$

In case of a designation one uses the witness sk', whereas in simulations the witness vsk is used. For brevity we assume that the parameters PP generated upon setup are implicit in every pk and vpk generated by Gen and DVGen respectively. Furthermore, we assume that R is implicitly defined by the scheme. Note that, while UDVS are defined so that the verification algorithm takes vsk as input, our scheme would also work when vsk is replaced by vpk.

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{PP}' \leftarrow \Sigma.\mathsf{PGen}(1^\kappa)$, $\mathsf{crs} \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$, and return $\mathsf{PP} \leftarrow (\mathsf{PP}', \mathsf{crs})$.

$\mathsf{DVGen}(\mathsf{PP})$: Run $\mathsf{vsk} \xleftarrow{R} \{0,1\}^\kappa$, set $\mathsf{vpk} \leftarrow f(\mathsf{vsk})$ and return $(\mathsf{vsk}, \mathsf{vpk})$.

$\mathsf{Desig}(\mathsf{pk}, \mathsf{vpk}, m, \sigma)$: Output $\delta \leftarrow (\mathsf{pk}', \sigma_\mathsf{R}, \pi)$, where

$$(\mathsf{sk}', \mathsf{pk}') \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa), \ (\mathsf{pk}_\mathsf{R}, \sigma_\mathsf{R}) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \mathsf{sk}'),$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}', \mathsf{vpk}), (\mathsf{sk}', \bot)).$$

$\mathsf{Sim}(\mathsf{pk}, \mathsf{vsk}, m)$: Output $\delta \leftarrow (\mathsf{pk}', \sigma_\mathsf{R}, \pi)$, where

$$(\mathsf{sk}_\mathsf{R}, \mathsf{pk}_\mathsf{R}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa), \ \mathsf{pk}' \leftarrow \mathsf{pk}_\mathsf{R} \cdot \mathsf{pk}^{-1}, \ \sigma_\mathsf{R} \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}_\mathsf{R}, m),$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}', f(\mathsf{vsk})), (\bot, \mathsf{vsk})).$$

$\mathsf{DVerify}(\mathsf{pk}, \mathsf{vsk}, m, \delta)$: Parse δ as $(\mathsf{pk}', \sigma_\mathsf{R}, \pi)$ and return 1 if the following holds, and 0 otherwise:

$$\Sigma.\mathsf{Verify}(\mathsf{pk} \cdot \mathsf{pk}', m, \sigma_\mathsf{R}) = 1 \quad \wedge \quad \Pi.\mathsf{Verify}(\mathsf{crs}, (\mathsf{pk}', f(\mathsf{vsk})), \pi) = 1.$$

Scheme 10: Black-Box Construction of UDVS

Theorem 2. *If Σ is EUF-CMA secure and perfectly adapts signatures, f is a one-way function, and Π is a witness indistinguishable proof of knowledge, then Scheme 10 is correct, simulation-sound DV-unforgeable, and provides strong non-transfer-ability privacy.*

We note that if non-transferability privacy is sufficient, Σ only needs to be adaptable. Then, besides the candidate schemes presented in Section 4, one can, e.g., also instantiate Scheme 10 with the very efficient Schnorr signature

[14] We note that our construction is inspired by earlier work of us on a variant of redactable signatures [DKS16].

scheme. We subsequently show that Theorem 2 holds where we note that if non-transferability privacy is sufficient, Σ only needs to be adaptable.

Lemma 15. *If Σ is correct, and Π is complete, then Scheme 10 is correct.*

Lemma 15 follows from inspection and the proof is therefore omitted.

Lemma 16. *If Σ is* EUF-CMA *secure and adapts signatures, f is a one-way function, and Π is a witness indistinguishable proof of knowledge, then Scheme 10 is simulation-sound DV-unforgeable.*

Proof. We followingly bound the success probability of an adversary using a sequence of games, where we let $q_{\mathsf{Sim}} \leq \mathsf{poly}(\kappa)$ be the number Sim queries.

Game 0: The original DV-unforgeability game.

Game 1: As Game 0, but inside the S oracle we execute the following modified Sim algorithm Sim', which additionally takes sk as input.

$\mathsf{Sim}'(\mathsf{pk}, \mathsf{vsk}, m, \boxed{\mathsf{sk}})$: Output $\delta = (\mathsf{pk}', \sigma_{\mathsf{R}}, \pi)$, where

$(\mathsf{sk}_{\mathsf{R}}, \mathsf{pk}_{\mathsf{R}}) \leftarrow \Sigma.\mathsf{KeyGen}(1^{\kappa}),\ \mathsf{pk}' \leftarrow \mathsf{pk}_{\mathsf{R}} \cdot \mathsf{pk}^{-1},\ \sigma_{\mathsf{R}} \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}_{\mathsf{R}}, m),$

$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}', f(\mathsf{vsk})), \boxed{(\mathsf{sk}_{\mathsf{R}} - \mathsf{sk}, \bot)}).$

Transition - Game 0 → Game 1: A distinguisher between $\mathcal{D}^{0\to 1}$ is a distinguisher for adaptive witness indistinguishability of Π, i.e., $|\mathsf{Pr}[S_0] - \mathsf{Pr}[S_1]| \leq \varepsilon_{\mathsf{wi}}(\kappa)$.

Game 2: As Game 1, but instead of generating crs upon PGen, we obtain $(\mathsf{crs}, \xi) \leftarrow \Pi.\mathsf{E}_1(1^{\kappa})$ and store ξ.

Transition - Game 1 → Game 2: A distinguisher between Game 1 and 2 distinguishes an honest crs from an extraction crs, i.e., $|\mathsf{Pr}[S_1] - \mathsf{Pr}[S_2]| \leq \varepsilon_{\mathsf{e1}}(\kappa)$.

Game 3: As Game 2, but whenever the adversary outputs a forgery $(m^\star, \delta^\star)$, where $\delta^\star = (\mathsf{pk}'^\star, \sigma^\star_{\mathsf{R}}, \pi^\star)$ we extract a witness $(\mathsf{sk}'^\star, \mathsf{vsk}'^\star) \leftarrow \Pi.\mathsf{E}_2(\mathsf{crs}, \xi, (\mathsf{pk}'^\star, \mathsf{vpk}^\star), \pi^\star)$ and abort if the extractor fails.

Transition - Game 2 → Game 3: Game 2 and Game 3 proceed identically, unless the extractor fails, i.e., $|\mathsf{Pr}[S_1] - \mathsf{Pr}[S_2]| \leq \varepsilon_{\mathsf{e2}}(\kappa)$.

Game 4: As Game 3, but we further modify Sim' as follows:

$\mathsf{Sim}'(\mathsf{pk}, \mathsf{vsk}, m, \mathsf{sk})$: Output $\delta = (\mathsf{pk}', \sigma_{\mathsf{R}}, \pi)$, where

$\boxed{\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)},$

$\boxed{(\mathsf{sk}', \mathsf{pk}') \leftarrow \Sigma.\mathsf{KeyGen}(1^{\kappa}),\ (\mathsf{pk}_{\mathsf{R}}, \sigma_{\mathsf{R}}) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \mathsf{sk}')},$

$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}', f(\mathsf{vsk})), (\boxed{\mathsf{sk}'}, \bot)).$

Transition Game 3 → Game 4: Under adaptability of signatures, this change is conceptual and $\mathsf{Pr}[S_3] = \mathsf{Pr}[S_4]$.

Game 5: As Game 4, but instead of generating $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(\mathsf{PP}')$, we obtain pk from an EUF-CMA challenger. Further, whenever a signature under pk is required, we use the Sign oracle provided by the challenger.

Transition - Game 4 → Game 5: This change is conceptual, i.e., $\mathsf{Pr}[S_4] = \mathsf{Pr}[S_5]$.

Game 6: As Game 5, but we obtain vpk from a one-wayness challenger and set $\mathsf{vsk} = \bot$. In addition, we simulate the Vrfy oracle by using vpk instead of $f(\mathsf{vsk})$ inside the $\mathsf{DVerify}$ algorithm.

Transition - Game 5 → Game 6: This change is conceptual, i.e., $\mathsf{Pr}[S_5] = \mathsf{Pr}[S_6]$.

In Game 6, we either have extracted $\mathsf{vsk}^\star$ so that $f(\mathsf{vsk}^\star) = \mathsf{vpk}$ and we can output $\mathsf{vsk}^\star$ to the one-wayness challenger, or we have extracted $\mathsf{sk}'^\star$ such that $\mu(\mathsf{sk}'^\star) = \mathsf{pk}'^\star$ and can obtain $(\mathsf{pk}, \sigma) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{pk} \cdot \mathsf{pk}'^\star, m^\star, \sigma_\mathsf{R}^\star, -\mathsf{sk}'^\star)$ and output $(m^\star, \sigma)$ as a forgery for Σ. Taking the union bound yields $\mathsf{Pr}[S_6] \leq \varepsilon_\mathsf{f}(\kappa) + \varepsilon_\mathsf{ow}(\kappa)$, and we obtain $\mathsf{Pr}[S_0] \leq \varepsilon_\mathsf{f}(\kappa) + \varepsilon_\mathsf{ow}(\kappa) + \varepsilon_\mathsf{wi}(\kappa) + \varepsilon_\mathsf{e1}(\kappa) + \varepsilon_\mathsf{e2}(\kappa) +$ which is negligible. □

Lemma 17. *If Σ perfectly adapts signatures, and Π is witness indistinguishable, then Scheme 10 is strongly non-transferable private.*

Proof. We bound the success probability using a sequence of games.

Game 0: The original non-transferability privacy game.

Game 1: As Game 0, but instead of generating crs upon setup, we obtain crs from a witness indistinguishability challenger $\mathcal{C}_\kappa^\mathsf{wi}$ upon Setup.

Transition - Game 0 → Game 1: This change is conceptual, i.e., $\mathsf{Pr}[S_0] = \mathsf{Pr}[S_1]$.

Game 2: As Game 1, but inside SoD we execute the following modified the Desig algorithm Desig' which additionally takes vsk as input:

$$\mathsf{Desig}'(\mathsf{pk}, \mathsf{vpk}, m, \sigma, \boxed{\mathsf{vsk}}) : \text{ Output } \delta \leftarrow (\mathsf{pk}', \sigma_\mathsf{R}, \pi), \text{ where}$$
$$(\mathsf{sk}', \mathsf{pk}') \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa),\ (\mathsf{pk}_\mathsf{R}, \sigma_\mathsf{R}) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \mathsf{sk}'),$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}', \mathsf{vpk}), \boxed{(\bot, \mathsf{vsk})}).$$

Transition - Game 1 → Game 2: A distinguisher between $\mathcal{D}^{1\to 2}$ is a distinguisher for adaptive witness indistinguishability of Π, i.e., $|\mathsf{Pr}[S_2] - \mathsf{Pr}[S_1]| \leq \varepsilon_\mathsf{wi}(\kappa)$.

Game 3: As Game 2, but we further modify Desig' as follows:

$$\mathsf{Desig}'(\mathsf{pk}, \mathsf{vpk}, m, \sigma, \mathsf{vsk}) : \text{ Output } \delta \leftarrow (\mathsf{pk}', \sigma_\mathsf{R}, \pi), \text{ where}$$
$$\boxed{(\mathsf{sk}_\mathsf{R}, \mathsf{pk}_\mathsf{R}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa),\ \mathsf{pk}' \leftarrow \mathsf{pk}_\mathsf{R} \cdot \mathsf{pk}^{-1},\ \sigma_\mathsf{R} \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}_\mathsf{R}, m)},$$
$$\pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (\mathsf{pk}', \mathsf{vpk}), (\bot, \mathsf{vsk})).$$

Transition - Game 2 → Game 3: By the perfect adaption of signatures, this change is conceptual, i.e., $\mathsf{Pr}[S_2] = \mathsf{Pr}[S_3]$.

In Game 3, Desig' is identical to Sim. This means that SoD is simulated independently of b and $|\mathsf{Pr}[S_3] - \mathsf{Pr}[S_0]| \leq \varepsilon_\mathsf{wi}(\kappa)$, which proves the lemma. □

5.3 Multisignatures

A multisignature scheme [IN83] allows a group of signers to jointly compute a compact signature for a message. Well known schemes are the BMS [Bol03] and the WMS [LOS+06] schemes that are directly based on the BLS [BLS04] and variants of the Waters' signature scheme [Wat05] respectively. Both of them are secure under the knowledge of secret key (KOSK) assumption, but can be shown to also be secure under (slightly tweaked) real-world proofs of possession protocols [RY07].

Our construction can be seen as a generalization of the paradigm behind all existing multisignature schemes. Making this paradigm explicit eases the search for new schemes, i.e., one can simply check whether a particular signature scheme is publicly key-homomorphic.

We now give a formal definition of multisignatures, where we follow Ristenpart and Yilek [RY07]. As already noted in Section 2, we use the KOSK modeled via RKey for simplicity. Nevertheless, we stress that we could use any other key-registration that provides extractability or also the extractable key-verification notion by Bagherzandi and Jarecki [BJ08]. This does not make any difference for our subsequent discussion as long as the secret keys are extractable.

Definition 27. *A multisignature scheme* MS *is a tuple* $(\mathsf{PGen}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *of PPT algorithms, which are defined as follows:*

$\mathsf{PGen}(1^\kappa)$: *This parameter generation algorithm takes a security parameter* κ *and produces global parameters* PP *(including the security parameters and a description of the message space* $\mathcal{M}$*).*

$\mathsf{KeyGen}(\mathsf{PP})$: *This algorithm takes the global parameters* PP *as input and outputs a secret (signing) key* sk *and a public (verification) key* pk*.*

$\mathsf{Sign}(\mathsf{PP}, \mathsf{PK}, m, \mathsf{sk}_i)$: *This is an interactive multisignature algorithm executed by a group of signers defined by* PK*, who intend to collectively sign the same message* m*. Each signer* S_i *executes* Sign *on public inputs* PP*, public key multiset* PK*, message* m *and its individual secret key* sk_i*. At the end of the protocol every signer outputs a multisignature* σ*.*

$\mathsf{Verify}(\mathsf{PP}, \mathsf{PK}, m, \sigma)$: *This algorithm takes public parameters* PP*, a public key multiset* PK*, a message* m *and a multisignature* σ *as input and outputs a bit* $b \in \{0, 1\}$*.*

The above tuple of algorithms must satisfy correctness. For any $\kappa \in \mathbb{N}$, any $n \in \mathsf{poly}(\kappa)$, for any $\mathsf{PP} \leftarrow \mathsf{PGen}(1^\kappa)$, for any $\{(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{PP})\}_{i \in [n]}$, for any $m \in \mathcal{M}$, if every signer i, $i \in [n]$, honestly executes $\sigma \leftarrow \mathsf{Sign}(\mathsf{PP}, \mathsf{PK}, m, \mathsf{sk}_i)$, then $\mathsf{Verify}(\mathsf{PP}, \mathsf{PK}, m, \sigma) = 1$.

Besides correctness, we require existential unforgeability under a chosen message attack against a single honest player.

Definition 28 (MSEUF-CMA). *A multisignature scheme* MS *is* MSEUF-CMA *secure, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathsf{PGen}(1^\kappa),\\ (\mathsf{sk}^\star, \mathsf{pk}^\star) \leftarrow \mathsf{KeyGen}(1^\kappa),\\ \mathcal{O} \leftarrow \{\mathsf{Sign}(\cdot,\cdot), \mathsf{RKey}(\cdot,\cdot,\cdot)\},\\ (\mathsf{PK}^\star, m^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pp}, \mathsf{pk}^\star)\end{array} : \begin{array}{r}\mathsf{Verify}(\mathsf{pp}, \mathsf{PK}^\star, m^\star, \sigma^\star) = 1 \wedge\\ \mathsf{pk}^\star \in \mathsf{PK}^\star \ \wedge \ m^\star \notin \mathcal{Q}^{\mathsf{Sign}} \wedge\\ (\mathsf{PK}^\star \setminus \{\mathsf{pk}^\star\}) \setminus \mathcal{Q}^{\mathsf{RKey}} = \emptyset)\end{array}\right] \leq \varepsilon(\kappa),$$

where the environment keeps track of signing and registration queries via $\mathcal{Q}^{\mathsf{Sign}}$ *and* $\mathcal{Q}^{\mathsf{RKey}}$*, respectively. The adversary has access to the following oracles:*

$\mathsf{Sign}(\mathsf{PK}, m)$: *This oracle obtains a public key set* PK *and returns* $\perp$ *if* $\mathsf{pk}^\star \notin \mathsf{PK}$*. Otherwise it simulates the honest signer by running* $\mathsf{Sign}(\mathsf{pp}, \mathsf{PK}, m, \mathsf{sk}^\star)$ *and interacting in a signing protocol with the other signers contained in* PK *(which are controlled by the adversary). In the end it sets* $\mathcal{Q}^{\mathsf{Sign}} \xleftarrow{\cup} m$*.*

$\mathsf{RKey}(\mathsf{sk}, \mathsf{pk})$: *This oracle checks if* $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{KeyGen}(\mathsf{pp})$ *and sets* $\mathcal{Q}^{\mathsf{RKey}} \xleftarrow{\cup} \mathsf{pk}$ *if so.*

Our Construction. We restrict ourselves to non-interactive Sign protocols, which means that every signer S_i locally computes a signatures σ_i and then broadcasts it to all other signers in PK. Furthermore, we consider Σ to work with common parameters PP and in Scheme 11 let us for the sake of presentation assume that $\mathsf{PK} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$ is an ordered set instead of a multiset.

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{pp} \leftarrow \Sigma.\mathsf{PGen}(1^\kappa)$ and return pp.

$\mathsf{KeyGen}(\mathsf{pp})$: Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(\mathsf{pp})$ and return $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{pp}, \mathsf{pk}, m, \mathsf{sk})$: Let $i \in [n]$. Every participating S_i with $\mathsf{pk}_i \in \mathsf{pk}$ proceeds as follows:

- Compute $\sigma_i \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}_i, m)$ and broadcast σ_i.
- Receive all signatures σ_j for $j \neq i$.
- Compute $(\mathsf{pk}, \sigma) \leftarrow \mathsf{Combine}(\mathsf{pk}, m, (\sigma_\ell)_{\ell \in [n]})$ and output σ.

$\mathsf{Verify}(\mathsf{pp}, \mathsf{pk}, m, \sigma)$: Return 1 if the following holds and 0 otherwise:

$$\Sigma.\mathsf{Verify}\big(\textstyle\prod_{\mathsf{pk} \in \mathsf{PK}} \mathsf{pk}, m, \sigma\big) = 1.$$

Scheme 11: Black-Box Construction of Multisignatures

Theorem 3. *If* Σ *is correct,* EUF-CMA *secure, and publicly key-homomorphic, then Scheme 11 is* MSEUF-CMA *secure.*

Proof. We show that an efficient adversary $\mathcal{A}$ against MSEUF-CMA can be efficiently turned into an efficient EUF-CMA adversary for Σ. To do so, we simulate the environment for $\mathcal{A}$ by obtaining $\mathsf{pk}^\star$ from an EUF-CMA challenger of Σ, then setting pp accordingly, and starting $\mathcal{A}$ on $(\mathsf{pp}, \mathsf{pk}^\star)$. Additionally, we record the secret keys provided to RKey in a list KEY indexed by the respective public keys, i.e., $\mathtt{KEY}[\mathsf{pk}] \leftarrow \mathsf{sk}$. Whenever a signature with respect to $\mathsf{pk}^\star$ is required we

use the Sign oracle provided by the challenger. Eventually, the adversary outputs $(\mathsf{PK}^\star, m^\star, \sigma^\star)$ such that $\Sigma.\mathsf{Verify}(\prod_{\mathsf{pk}\in\mathsf{PK}^\star}\mathsf{pk}, m^\star, \sigma^\star) = 1$, $\mathsf{pk}^\star \in \mathsf{PK}^\star$, all other keys in $\mathsf{PK}^\star$ were registered, yet $m^\star$ was never queried to the signing oracle. We compute $\mathsf{sk}' \leftarrow \sum_{\mathsf{pk}\in\mathsf{PK}^\star\setminus\{\mathsf{pk}^\star\}} -\mathtt{KEY}[\mathsf{pk}]$, compute $\sigma' \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}', m^\star)$, obtain $(\mathsf{pk}^\star, \sigma) \leftarrow \mathsf{Combine}((\prod_{\mathsf{pk}\in\mathsf{PK}^\star}\mathsf{pk}, \prod_{\mathsf{pk}\in\mathsf{PK}^\star\setminus\{\mathsf{pk}^\star\}}\mathsf{pk}^{-1}), m^\star, (\sigma^\star, \sigma'))$ and output $(m^\star, \sigma)$ as a forgery. □

6 Applications to Simulation-Sound Extractable NIZK

Some of the constructions in the previous sections implicitly use techniques to ensure that even though we have to simulate proofs within our security reduction, we can still extract the required witness for the forgery. In this section we isolate the essence of this techniques and show that they are generally applicable to extend witness indistinguishable argument systems admitting proofs of knowledge to (weak) simulation-sound extractable NIZK arguments (SSE NIZKs) using $\mathsf{EUF\text{-}CMA}$ secure signature schemes that adapt signatures. This makes our techniques useful in a broader range of applications. For the stronger variant of simulation-sound extractability we additionally require strong one-time signatures, while for weak simulation-sound extractability we can avoid them.

We start by defining strong one-time signatures and an efficient instantiation under standard assumptions can be found in [Gro06].

Definition 29 (Strong One-Time Signature Scheme). *A strong one-time signature scheme Σ_{ot} provides the same interface as a conventional signature scheme Σ and satisfies the following unforgeability notion: For all PPT adversaries $\mathcal{A}$ there is a negligible function $\varepsilon(\cdot)$ such that*

$$\mathsf{Pr}\left[\begin{matrix}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa),\\ (m^\star,\sigma^\star) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{pk})\end{matrix} : \begin{matrix}\mathsf{Verify}(\mathsf{pk},m^\star,\sigma^\star)=1 \wedge\\ (m^\star,\sigma^\star)\notin\mathcal{Q}^{\mathsf{Sign}}\end{matrix}\right] \leq \varepsilon(\kappa),$$

where the oracle $\mathsf{Sign}(\mathsf{sk},\mathsf{m}) := \Sigma.\mathsf{Sign}(\mathsf{sk},\mathsf{m})$ can only be called once.

For our construction, first let L be an arbitrary $\mathbf{NP}$-language $L = \{x \mid \exists\ w : R(x,w) = 1\}$, for which we aim to construct a SSE NIZK, and let L' be defined via the $\mathbf{NP}$-relation R':

$$((x,\mathsf{cpk},\mathsf{pk}),(w,\mathsf{csk}-\mathsf{sk})) \in R' \iff (x,w)\in R \ \vee\ \mathsf{cpk} = \mathsf{pk}\cdot\mu(\mathsf{csk}-\mathsf{sk}).$$

In Scheme 12 we present our construction of a SSE NIZK Π_{sse} for L. Our technique is inspired by [GMY03, GMY06, Gro06] and also similar to [DHLW10, ADK+13] but much easier to implement. This is mainly due to the fact that the adaptability of the used signature scheme allows us to get rid of the requirement to prove statements about the validity of the signature, and, instead, only requires us to prove a simple statement demonstrating knowledge of the shift amount.

Essentially, the intuition of our construction is the following. We use a combination of an adaptable $\mathsf{EUF\text{-}CMA}$ secure signature scheme Σ and a strong one-time signature scheme Σ_{ot} to add the required non-malleability guarantees

to the underlying argument system.[15] Upon each proof computation, we use Σ to "certify" the public key of a newly generated key pair of Σ_{ot}. The associated secret key of Σ_{ot} is then used to sign the parts of the proof which must be non-malleable. Adaptability of Σ makes it possible to also use newly generated keys of Σ upon each proof computation. In particular, the relation associated to L' is designed so that the second clause in the OR statement is the "shift amount" required to shift such signatures to signatures under a key cpk in the crs. A proof for $x \in L$ is easy to compute when given w such that $(x, w) \in R$. One does not need a satisfying assignment for the second clause in the OR statement, and can thus compute all signatures under newly generated keys. To simulate proofs, however, we can set up crs in a way that we know csk corresponding to cpk, compute the "shift amount" and use it as a satisfying witness for the second clause in the OR statement. Under this strategy, the witness indistinguishability of the underlying argument system for L', the crs indistinguishability provided by the proof of knowledge property, and the secret-key to public-key homomorphism of Σ guarantees the zero-knowledge property of our argument system for L.

What remains is to argue that we can use the extractor of the underlying argument system for L' as an extractor for L in the simulation-sound extractability setting. In fact, under the strategy we use, we never have to simulate proofs for statements outside L' which is sufficient for the extractor for L' to work with overwhelming probability. Furthermore, we can show that the probability to extract a valid witness for the second clause in the OR statement is negligible, as this either yields a forgery with respect to Σ_{ot} under some $\mathsf{pk}_{\mathsf{ot}}$ previously obtained from the simulator (if the adversary modified any of the non-malleable parts of a proof previously obtained via the simulator) or for Σ under cpk (if $\mathsf{pk}_{\mathsf{ot}}$ has never been certified). Now we know, however, that the extractor for L' works with overwhelming probability by definition, which means that we will extract a satisfying witness for $x \in L$ with overwhelming probability. Formally, we can state:

Theorem 4. *Let Π be a complete, witness indistinguishable non-interactive argument of knowledge system for the language L', let Σ be an $\mathsf{EUF\text{-}CMA}$ secure signature scheme that adapts signatures, and let Σ_{ot} be a strong one-time signature scheme, then the argument system Π_{sse} is a complete and simulation-sound extractable argument system for language L.*

We show that Theorem 4 holds by proving the subsequent lemmas.

Lemma 18. *If Π is complete and Σ is correct, Π_{sse} is complete.*

The lemma above follows from inspection and the proof is therefore omitted.

Lemma 19. *If Π is a witness indistinguishable proof of knowledge, and Σ provides a secret-key to public-key homomorphism, then Π_{sse} is zero-knowledge.*

Proof. We subsequently prove that zero-knowledge follows from witness indistinguishability.

[15] Σ_{ot} is only required as the signatures produced by Σ may be malleable on their own.

$\mathsf{Setup}(1^\kappa)$: Run $\mathsf{crs}_\Pi \leftarrow \Pi.\mathsf{Setup}(1^\kappa)$, $(\mathsf{csk}, \mathsf{cpk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$ and return $\mathsf{crs} \leftarrow (\mathsf{crs}_\Pi, \mathsf{cpk})$.

$\mathsf{Proof}(\mathsf{crs}, x, w)$: Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$, $(\mathsf{sk}_{\mathsf{ot}}, \mathsf{pk}_{\mathsf{ot}}) \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{KeyGen}(1^\kappa)$, and return $\pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$, where

$$\pi_\Pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), (w, \bot)),\ \sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{\mathsf{ot}}),\ \text{and}$$

$$\sigma_{\mathsf{ot}} \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma).$$

$\mathsf{Verify}(\mathsf{crs}, x, \pi)$: Parse π as $(\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$ and return 1 if the following holds and 0 otherwise:

$$\Pi.\mathsf{Verify}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), \pi_\Pi) = 1 \ \wedge\ \Sigma.\mathsf{Verify}(\mathsf{pk}, \mathsf{pk}_{\mathsf{ot}}) = 1 \ \wedge$$

$$\Sigma_{\mathsf{ot}}.\mathsf{Verify}(\mathsf{pk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma) = 1.$$

$\mathsf{S}_1(1^\kappa)$: Run $(\mathsf{crs}_\Pi, \bot) \leftarrow \Pi.\mathsf{E}_1(1^\kappa)$, $(\mathsf{csk}, \mathsf{cpk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$ and return (crs, τ), where

$$\mathsf{crs} \leftarrow (\mathsf{crs}_\Pi, \mathsf{cpk}) \text{ and } \tau \leftarrow \mathsf{csk}.$$

$\mathsf{S}_2(\mathsf{crs}, \tau, x)$: Parse τ as csk, run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$, $(\mathsf{sk}_{\mathsf{ot}}, \mathsf{pk}_{\mathsf{ot}}) \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{KeyGen}(1^\kappa)$, and return $\pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$, where

$$\pi_\Pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), (\bot, \mathsf{csk} - \mathsf{sk})),\ \sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{\mathsf{ot}}),\ \text{and}$$

$$\sigma_{\mathsf{ot}} \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma).$$

$\mathsf{S}(1^\kappa)$: Run $(\mathsf{crs}_\Pi, \xi) \leftarrow \Pi.\mathsf{E}_1(1^\kappa)$, $(\mathsf{csk}, \mathsf{cpk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$ and return $(\mathsf{crs}, \tau, \xi)$, where

$$\mathsf{crs} \leftarrow (\mathsf{crs}_\Pi, \mathsf{cpk}) \text{ and } \tau \leftarrow \mathsf{csk}.$$

$\mathsf{E}(\mathsf{crs}, \xi, x, \pi)$: Run $(w, \bot) \leftarrow \Pi.\mathsf{E}_2(\mathsf{crs}, \xi, x, \pi)$ and return w.

Scheme 12: Simulation-sound extractable NIZK Argument System Π_{sse}.

Game 0: The zero-knowledge game, where we use the real $\mathsf{Proof}(\mathsf{crs}, \cdot, \cdot)$ algorithm on witnesses $(w, \bot)$ to reply to queries of the adversary.

Game 1: As Game 0, but we store csk upon Setup.

Transition Game 0 → Game 1: This change is conceptual, i.e., $\mathsf{Pr}[S_0] = \mathsf{Pr}[S_1]$.

Game 2: As Game 1, but use the following modified Proof algorithm Proof' which additionally takes csk as input:

$\mathsf{Proof}'(\mathsf{crs}, x, w, \boxed{\mathsf{csk}})$: Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$, $(\mathsf{sk}_{\mathsf{ot}}, \mathsf{pk}_{\mathsf{ot}}) \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{KeyGen}(1^\kappa)$, and return $\pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$, where

$$\pi_\Pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), \boxed{(\bot, \mathsf{csk} - \mathsf{sk})}),\ \sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{\mathsf{ot}}),\ \text{and}$$

$$\sigma_{\mathsf{ot}} \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma).$$

Transition - Game 1 → Game 2: We present a hybrid game which shows that both games are indistinguishable under the witness indistinguishability of the argument system. First, we conceptually change the Setup algorithm to Setup' which obtains crs_Π from a witness indistinguishability challenger:

$\mathsf{Setup}(1^\kappa)$: Run $\boxed{\mathsf{crs}_\Pi \leftarrow \mathcal{C}_\kappa^{\mathsf{wi}}}$, $(\mathsf{csk}, \mathsf{cpk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$, return $\mathsf{crs} \leftarrow (\mathsf{crs}_\Pi, \mathsf{cpk})$.

The change above is only conceptual. Furthermore, we use the following Proof'' algorithm instead of Proof':

$\mathsf{Proof}''(\mathsf{crs}, x, w, \boxed{\mathsf{csk}})$: Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$, $(\mathsf{sk}_{\mathsf{ot}}, \mathsf{pk}_{\mathsf{ot}}) \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{KeyGen}(1^\kappa)$, and return $\pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$, where

$\boxed{\pi_\Pi \leftarrow \mathcal{C}_\kappa^{\mathsf{wi}}((x, \mathsf{cpk}, \mathsf{pk}), (w, \perp), (\perp, \mathsf{csk} - \mathsf{sk}))}$, $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{\mathsf{ot}})$, and $\sigma_{\mathsf{ot}} \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma)$.

Now depending of whether the challenger uses the first witness ($b = 0$) or the second witness ($b = 1$) we either simulate Game 1 or Game 2. More precisely, Proof'' produces the identical distribution as Proof if $b = 0$ and the identical distribution to Proof' if $b = 1$. That is $|\mathsf{Pr}[S_1] - \mathsf{Pr}[S_2]| \leq \epsilon_{\mathsf{wi}}(\kappa)$.

Game 3: As Game 2, but we further modify Proof' so that it no longer takes w as input:

$\mathsf{Proof}'(\mathsf{crs}, x, \mathsf{csk})$: Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$, $(\mathsf{sk}_{\mathsf{ot}}, \mathsf{pk}_{\mathsf{ot}}) \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{KeyGen}(1^\kappa)$, and return $\pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$, where

$\pi_\Pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), (\perp, \mathsf{csk} - \mathsf{sk}))$, $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{\mathsf{ot}})$, and $\sigma_{\mathsf{ot}} \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma)$.

Transition - Game 2 → Game 3: This change is conceptual, i.e., $\mathsf{Pr}[S_2] = \mathsf{Pr}[S_3]$.

Game 4: As Game 3, but instead of obtaining crs using Setup, we obtain $(\mathsf{crs}, \tau) \leftarrow \mathsf{S}_1$ (observe that $\tau = \mathsf{csk}$, so we still know csk). Now the setup is already as in the second distribution of the zero-knowledge game.

Transition - Game 3 → Game 4: A crs output by S_1 is indistinguishable from an honest crs under the CRS indistinguishability provided by the proof of knowledge property (observe that S_1 internally uses E_1 to obtain crs). Thus, $|\mathsf{Pr}[S_3] - \mathsf{Pr}[S_4]| \leq \varepsilon_{\mathsf{pok1}}(\kappa)$.

Game 5: As Game 4, but we further modify Proof' as follows:

$\mathsf{Proof}'(\mathsf{crs}, \boxed{\tau}, x)$: $\boxed{\text{Parse } \tau \text{ as } \mathsf{csk}}$. Run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Sigma.\mathsf{KeyGen}(1^\kappa)$, $(\mathsf{sk}_{\mathsf{ot}}, \mathsf{pk}_{\mathsf{ot}}) \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{KeyGen}(1^\kappa)$, and return $\pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$, where

$\pi_\Pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), (\perp, \mathsf{csk} - \mathsf{sk}))$, $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, \mathsf{pk}_{\mathsf{ot}})$, and $\sigma_{\mathsf{ot}} \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma)$.

Now Proof' is equivalent to S_2.

Transition - Game 4 → Game 5: This change is conceptual, i.e., $\mathsf{Pr}[S_4] = \mathsf{Pr}[S_5]$.

In Game 0 we simulate the first distribution of the zero-knowledge game whereas in Game 5 we simulate the second distribution. We have that $|\mathsf{Pr}[S_0] - \mathsf{Pr}[S_5]| \leq \varepsilon_{\mathsf{wi}}(\kappa) + \varepsilon_{\mathsf{pok1}}(\kappa)$ which concludes the proof. □

Now, we have already established the existence of a simulator by proving zero-knowledge and can go on by proving simulation-sound extractability.

Lemma 20. *If Π is a witness indistinguishable proof of knowledge and Σ is* EUF-CMA *secure and adapts signatures, then Π_{sse} is simulation-sound extractable.*

We prove the lemma above by showing that even when the adversary sees simulated proofs for arbitrary statements, we are still able to extract a witness w from a proof $\pi^\star$ for a statement $x^\star$ so that $R(x^\star, w) = 1$ as long as $(x^\star, \pi^\star)$ does not correspond to an query-answer pair of the simulation oracle.

Proof. By Lemma 19, we know that $(\mathsf{S}_1, \mathsf{S}_2)$ is a suitable zero-knowledge simulator. In addition, we have that the output of S is identical to S_1 when restricted to (crs, τ). This completes CRS indistinguishability part of the proof. To prove the second part of simulation-sound extractability we proceed using a sequence of games where we let $q \leq \mathsf{poly}(\kappa)$ be the number of queries to the simulator.

Game 0: The original simulation-sound extractability game.

Game 1: As Game 0, but we engage with an EUF-CMA challenger within S. That is, we execute the following modified S algorithm S':

$$\mathsf{S}'(1^\kappa): \text{ Run } (\mathsf{crs}_\Pi, \xi) \leftarrow \Pi.\mathsf{E}_1(1^\kappa),\ \mathsf{cpk} \leftarrow \mathcal{C}^{\mathsf{f}}_\kappa, \text{ and return } (\mathsf{crs}, \tau, \xi), \text{ where}$$
$$\mathsf{crs} \leftarrow (\mathsf{crs}_\Pi, \mathsf{cpk}) \text{ and } \tau \leftarrow \bot.$$

This also requires us to modify the S_2 algorithm used for simulation to obtain S'_2. Essentially, we leverage the adaptability of signatures to shift signatures obtained from the signing oracle provided by the EUF-CMA challenger under cpk to signatures under a random key. The "shift-amount" is then a valid witness for the relation.

$$\mathsf{S}'_2(\mathsf{crs}, x): \text{ Obtain } \boxed{\mathsf{sk}' \xleftarrow{R} \mathbb{H},\ \mathsf{pk} \leftarrow \mathsf{cpk} \cdot \mu(\mathsf{sk}')}. \text{ Further, run } (\mathsf{sk}_{\mathsf{ot}}, \mathsf{pk}_{\mathsf{ot}}) \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{KeyGen}(1^\kappa), \text{ and return } \pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}}), \text{ where}$$
$$\pi_\Pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), (\bot, \boxed{-\mathsf{sk}'})),\ \boxed{\sigma' \leftarrow \mathcal{C}^{\mathsf{f}}_\kappa.\mathsf{Sign}(\mathsf{pk}_{\mathsf{ot}})},$$
$$\boxed{(\sigma, \bot) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{cpk}, \mathsf{pk}_{\mathsf{ot}}, \sigma', \mathsf{sk}')}, \text{ and}$$
$$\sigma_{\mathsf{ot}} \leftarrow \Sigma_{\mathsf{ot}}.\mathsf{Sign}(\mathsf{sk}_{\mathsf{ot}}, \pi_\Pi || x || \mathsf{pk} || \sigma).$$

Transition - Game 0 → Game 1: Under adaptability of signatures, this change is only conceptual, i.e., $\mathsf{Pr}[S_0] = \mathsf{Pr}[S_1]$.

Game 2: As Game 1, but we further modify S'_2 as follows: we engage with a strong one-time signature challenger in each call and keep a mapping from challengers to keys.

$\mathsf{S}_2'(\mathsf{crs}, x)$: Obtain $\mathsf{sk}' \xleftarrow{R} \mathbb{H}$, $\mathsf{pk} \leftarrow \mathsf{cpk} \cdot \mu(\mathsf{sk}')$. Further, obtain $\boxed{\mathsf{pk}_{\mathsf{ot}} \leftarrow \mathcal{C}_\kappa^{\mathsf{ot}}}$ and return $\pi \leftarrow (\pi_\Pi, \mathsf{pk}, \sigma, \mathsf{pk}_{\mathsf{ot}}, \sigma_{\mathsf{ot}})$, where

$$\pi_\Pi \leftarrow \Pi.\mathsf{Proof}(\mathsf{crs}, (x, \mathsf{cpk}, \mathsf{pk}), (\bot, -\mathsf{sk}')),\ \sigma' \leftarrow \mathcal{C}_\kappa^{\mathsf{f}}.\mathsf{Sign}(\mathsf{pk}_{\mathsf{ot}}),$$

$$(\sigma, \bot) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{cpk}, \mathsf{pk}_{\mathsf{ot}}, \sigma', \mathsf{sk}'), \text{ and}$$

$$\boxed{\sigma_{\mathsf{ot}} \leftarrow \mathcal{C}_\kappa^{\mathsf{ot}}.\mathsf{Sign}(\pi_\Pi || x || \mathsf{pk} || \sigma)}.$$

Transition - Game 1 → Game 2: This change is conceptual, i.e., $\mathsf{Pr}[S_1] = \mathsf{Pr}[S_2]$.

Game 3: As Game 2, but we assume that E_2 used inside E does not fail to extract a valid witness with respect to L' (i.e., we abort if it fails).

Transition - Game 2 → Game 3: We bound the probability that the adversary outputs a tuple $(x^\star, \pi^\star)$ in Game 3 so that E_2 fails. We refer to this event as F_1. For the sake of contradiction assume that $\mathsf{Pr}[F_1]$ is non-negligible. Then we could obtain crs_Π from a proof of knowledge challenger and set $\xi \leftarrow \bot$ within S_1'. Whenever the adversary outputs $(x^\star, \pi^\star)$ we output it to the challenger. Now, the probability for our reduction to win the proof of knowledge game is exactly $\mathsf{Pr}[F_1]$. That is, we have that $|\mathsf{Pr}[S_2] - \mathsf{Pr}[S_3]| \leq \varepsilon_{\mathsf{e2}}(\kappa)$.

Game 4: As Game 3, but we assume that for every tuple $(x^\star, \pi^\star)$ output by the adversary, E never fails to output a witness w so that $R(x, w) = 1$ (i.e., we abort if it fails).

Transition Game 3 → Game 4: We bound the probability that the adversary manages to come up with a tuple $(x^\star, \pi^\star)$, where $\pi^\star = (\pi_\Pi^\star, \mathsf{pk}^\star, \sigma^\star, \mathsf{pk}_{\mathsf{ot}}^\star, \sigma_{\mathsf{ot}}^\star)$, so that we extract $(\bot, \mathsf{sk}_{\mathsf{e}}) \leftarrow \mathsf{E}_2(\mathsf{crs}, \xi, x^\star, \pi^\star)$ inside E. We refer to this event as F_2. If F_2 happens, we obtain a signature $(\sigma_{\mathsf{f}}, \mathsf{cpk}) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{pk}^\star, \mathsf{pk}_{\mathsf{ot}}^\star, \sigma^\star, \mathsf{sk}_{\mathsf{e}})$. By definition of the game we know that $(x^\star, \pi^\star)$ is not a query-answer pair of the simulator. Thus, we have two cases: (1) A signature on $\mathsf{pk}_{\mathsf{ot}}^\star$ was never obtained from the EUF-CMA challenger and we can output $(\mathsf{pk}_{\mathsf{ot}}^\star, \sigma_{\mathsf{f}})$ as a valid EUF-CMA forgery. (2) A signature on $\mathsf{pk}_{\mathsf{ot}}^\star$ was previously obtained. Then we have by definition that either $\pi_\Pi^\star || x^\star || \mathsf{pk}^\star || \sigma^\star$ or $\sigma_{\mathsf{ot}}^\star$ is different from the tuple signed by the strong one-time signature challenger upon simulation and we can output $(\pi_\Pi^\star || x^\star || \mathsf{pk}^\star || \sigma^\star, \sigma_{\mathsf{ot}}^\star)$ as a forgery for the strong one-time signature scheme to the respective challenger. Taking the union bound yields $|\mathsf{Pr}[S_3] - \mathsf{Pr}[S_4]| \leq q \cdot \varepsilon_{\mathsf{ot}}(\kappa) + \varepsilon_{\mathsf{f}}(\kappa)$.

In Game 4, we always extract a witness w such that $R(x^\star, w) = 1$, i.e., $\mathsf{Pr}[S_4] = 0$; Game 0 and Game 4 are computationally indistinguishable. Overall, we obtain $\mathsf{Pr}[S_0] \leq q \cdot \varepsilon_{\mathsf{ot}}(\kappa) + \varepsilon_{\mathsf{f}}(\kappa) + \varepsilon_{\mathsf{e2}}(\kappa)$, which completes the proof. □

Remark 2. It is quite easy to see, that the theorem above could also be proven if Σ only provides the weaker notion of non-adaptive CMA (naCMA, see, e.g., [Kat10]) security, i.e., where the adversary has to define the list of message for which it obtains signatures before seeing the public key of the scheme. That is, if we let q be the number of queries to the simulation oracle, the reduction would generate q one-time signing key pairs and obtain signatures on all the public keys from the naCMA challenger in the beginning (as opposed to freshly generating

the one-time signing key pairs and adaptively obtaining signatures upon each query to the simulation oracle). The reduction stores all the key pairs and signatures and then uses a fresh tuple upon each call to the simulation oracle. We decided to conduct our proof under EUF-CMA instead of naCMA for two reasons: (1) it allows us to present the following construction of weak simulation-sound extractable arguments of knowledge much more succinctly, and (2) one can generically extend naCMA secure signatures to EUF-CMA secure signatures in the standard model [ST01] via chameleon hashing and this extension does not conflict with adaptabiltiy.

6.1 Weak Simulation-Sound Extractability

If one allows the proofs to be malleable and only requires non-malleability with respect to the statements, one can omit the strong one-time signature scheme and directly sign $\pi_{\Pi}||x||\mathsf{pk}$ using Σ. We refer to this modified argument system as Π_{wsse}.

Theorem 5. *Let Π be a complete, witness indistinguishable non-interactive argument of knowledge system for the language L', and let Σ be an* EUF-CMA *secure signature scheme that adapts signatures, then the argument system Π_{wsse} is a complete, weakly simulation-sound extractable argument system for language L.*

Proof (Sketch). The proof for the theorem above is exactly the same as the one for simulation-sound extractability, except that we do not need to engage with challengers for the one-time signature scheme (i.e., in Game 2 nothing is changed) and $\mathsf{Pr}[F_2]$ is exactly the same as extracting a forgery for Σ in the transition between Game 3 and Game 4. □

6.2 Signatures of Knowledge

Also note that using our techniques in a non-black box way directly yields signatures of knowledge [CL06]. That is, a signature of knowledge on a message m with respect to statement x is simply a proof with respect to x, where m is additionally included upon computing the signature using Σ_{ot}, i.e., one signs $\pi_{\Pi}||x||\mathsf{pk}||\sigma||m$. Then one obtains signatures of knowledge in the strong sense [BCC+15], where even the signature (i.e., the proof) is non-malleable. If security in the original sense—the counterpart of weak simulation-sound extractability where the signature (i.e., the proof π) itself may be malleable—is sufficient, one can even omit the strong one-time signature scheme and directly sign $\pi_{\Pi}||x||\mathsf{pk}||m$ using Σ.

As already mentioned in [CL06], a immediate application of signatures of knowledge is the construction of ring signatures. Obtaining such a construction based on our techniques presented in this section can thus be seen as an alternative to the direct construction in Section 5.1.

6.3 Performance Advantages

Our technique provides nice properties when it comes to converting NIWI/NIZK Groth-Sahai proofs [GS08] over pairing product equations to simulation-sound extractable arguments of knowledge. We can thereby use Waters' signatures as described in Section 4.5 for an instantiation under standard assumptions. Our technique constitutes an alternative to known techniques [Gro06, DHLW10, ADK+13] and yields constructions that are much easier to implement with favorable properties regarding efficiency when, e.g., compared to [Gro06, DHLW10, ADK+13] for SSE NIZKs or [BFG13] for signatures of knowledge without random oracles.

When looking at the existing techniques to turn NIWI/NIZK proofs into simulation sound extractable proof systems, one may observe that all of them are similar in the sense that they modify the original language by adding an additional disjunctive clause, which can be used by the simulator to simulate proofs. Consequently, we can compare all the approaches by simply comparing the respective overheads imposed by the additional disjunctive clause.

To make it explicit what we exactly mean by overhead, let us consider the following set of pairing product equations

$$\{e(\underline{x_i}, \hat{g}) = e(h, \hat{x}_i)\}_{i\in[n]},$$

where we underline the variables of which we want to prove knowledge. Roughly speaking, one will only reveal commitments to the underlined values when conducting a Groth-Sahai proof, which is why we will henceforth speak of the underlined values as commitments. What we want to prove in such a disjunctive statement is that we know a satisfying value x_i committed to in $\underline{x_i}$ for at least one $i \in [n]$. Note that we have chosen this set of equations in a way that it is exactly of the form as we require it when using Water's signatures to instantiate our compiler for ring signatures. That is, the first $n-1$ equations represent the statement to be proven[16], whereas the last equation with index n represents the equation required to prove knowledge of the shift amount to cpk. Latter remains the same, irrespective of the statement we want to prove, which is why we simply look at the overhead imposed by this additional equation and the costs for expressing the additional disjunction. We start by discussing how we realize the disjunction. We follow the ideas of [Gro06] and prove the equations

$$\{e(\underline{x_i}, \hat{g}) = e(h, \underline{\hat{x}_i})\}_{i\in[n]}, \tag{1}$$

$$e(g^{-1} \prod_{i\in[\ell]} \underline{g_i}, \hat{g}) = 1, \text{ and} \tag{2}$$

$$\{e(\underline{g_i}, (\hat{x_i})^{-1}\underline{\hat{x}_i}) = 1\}_{i\in[n]}. \tag{3}$$

[16] The actual statement can of course be different if one chooses to use techniques to achieve more compact ring signatures or in case one simply requires a different statement when using SSE NIZKs in other applications.

Equation (1) modifies the original set of equations so that one additionally proves knowledge of $\hat{x}_i$. This set of equations is now trivially satisfiable. To ensure that the prover actually uses a non-trivial witness for at least one of the equations, one needs two additional equations. Equation (2) constitutes a selector equation, which can only be satisfied if at least one $\underline{g_i}$ is a commitment to $g_i \neq 1$. Finally, Equation (3) ensures that the commitment $\underline{\hat{x}_i}$ is actually a commitment to $\hat{x}_i$ for every i where $\underline{g_i}$ is a commitment to some $g_i \neq 1$.

As we are working in the SXDH setting, we have that a single commitment in $\mathbb{G}_i, i \in \{1, 2\}$ requires two group elements in $\mathbb{G}_i$, whereas a proof requires at most 8 elements in $\mathbb{G}_1^4 \times \mathbb{G}_2^4$. The overhead imposed by proving the "shift amount" in an additional disjunctive clause is one additional commitment in the proof of Equation (2) as well as two additional equations to prove, where the costs are two commitments (one in $\mathbb{G}_1$ and one in $\mathbb{G}_2$) and one proof each (Equations (1) and(3)). Furthermore, using the strong one-time signature from [Gro06] we require 4 group elements in $\mathbb{G}_1$ for the verification key and 2 group elements in $\mathbb{Z}_p$ for the signature. All in all, this sums up to an overhead of 14 group elements in $\mathbb{G}_1$ and 8 group elements in $\mathbb{G}_2$ and 2 elements from $\mathbb{Z}_p$. We note that, depending of the form of the relation one desires to prove, one might have to instantiate the disjunctive statement slightly different.

Regarding a comparison to existing work in the context of SSE NIZK, the approaches in [Gro06, DHLW10] are theoretical results. For example, in [Gro06] Groth says *"Caveat: The constants are large, and therefore our schemes are not practical"*, and also Dodis et al. [DHLW10] do not consider an actual instantiation. Abe et al. in [ADK+13] consider a practical instantiation of the Dodis et al. simulation-sound extractable NIZK arguments, but the overhead to achieve simulation-sound extractability is much larger than when using the instantiation of our scheme as detailed above.

In particular, when taking the smallest unbounded simulation-sound extractable[17] scheme SE-NIZK1 from [ADK+13] with the parameters they suggest ($d = 2^{20}$), this yields an overhead of $21 \cdot d + 43 = 463$ group elements. Consequently our approach can be seen as a considerable improvement in practical settings. We present a concrete comparison in Table 2.

Scheme	Overhead	Setting	Note
[ADK+13]	$463 \times \mathbb{G}_1$	symmetric	using SE-NIZK1
Ours	$14 \times \mathbb{G}_1 + 8 \times \mathbb{G}_2 + 2 \times \mathbb{Z}_p$	SXDH	using Waters' signatures

Table 2. Comparison of the overheads required to achieve the SSE property compared to just proving the plain statement. While [ADK+13] is using symmetric pairings, we work in an SXDH setting which is why the approaches are not directly comparable. However, transforming a symmetric scheme to the SXDH setting requires to duplicate every element and maintaining a representation of it in both groups, $\mathbb{G}_1$ and $\mathbb{G}_2$. This should make our performance advantages clear.

While our system currently inherits the non-tight security from the Waters' signature scheme, an interesting future direction would be to investigate whether

[17] What they call unbounded simulation-sound extractability is equivalent to our notion of simulation-sound extractability.

one can achieve such high efficiency and tight security at the same time. Finally, we note that the signatures of knowledge from [BFG13] come close to our signatures of knowledge, but they still have to prove knowledge of one group element more than we have to. They also build upon Waters' signatures, i.e., their security reduction is also non-tight.

Additionally, we remark that in the example above one could additionally reduce the overhead by four elements (either in $\mathbb{G}_1$ or $\mathbb{G}_2$) by applying the optimizations from [EG14].

7 Tight Multi-User Security from Key-Homomorphisms

When using signature schemes in practice, it is often argued that EUF-CMA security does not appropriately capture the requirements appearing in practical settings [GMS02, MS04]. Currently we experience a growing interest in the multi-user setting (e.g., [BJLS16, GHKW16, KMP16]), where an adversary can attack one out of various public keys instead of a single one. This setting is also a frequently discussed topic on the mailing list of the CFRG.[18]

Since many schemes have already been investigated regarding their single-user security, an important question in this context is whether one can infer statements about the multi-user security of a certain scheme based on its single-user security. Without using any further properties of the signature scheme, every naïve reduction looses a factor of N, where N is the number of users in the system [GMS02].[19] Such a reduction is non-tight and drastically reduces the security guarantees a scheme provably provides. Thus, it is important to come up with tight security reductions. This was done in [GMS02], where a tight implication from single-user EUF-CMA to multi-user EUF-CMA for Schnorr signatures was proven. Unfortunately, a flaw in this proof was discovered by Bernstein in [Ber15], where it was also shown that single-user EUF-CMA tightly implies key-prefixed mulit-user EUF-CMA for Schnorr signatures. Recently, Lacharité in [Lac18] showed this tight implication under key-prefixing for BLS [BLS04] signatures and BGLS [BGLS03] aggregate signatures. Subsequent to the work in [Ber15], Kiltz et al. [KMP16] studied multi-user security of random self-reducible canonical identification schemes when turned to signatures in the random oracle model (ROM) using the Fiat-Shamir heuristic. They show that for such schemes single-user security tightly implies multi-user security without key-prefixing. This, in particular, holds for Schnorr signatures.

Our theorem essentially generalizes the work of [Ber15, Lac18] to be applicable to a larger class of signature schemes. For example, using our results from Section 4, it attests the multi-user EUF-CMA security of various variants of Waters' signatures [Wat05] and PS signatures [PS16], which were previously unknown to provide tight multi-user security. Furthermore, it can be seen as orthogonal to the work of [KMP16], where the requirement of key-prefixing is

[18] https://www.ietf.org/mail-archive/web/cfrg/current/maillist.html

[19] For instance, assuming 2^{30} keys in a system, such a reduction loss requires to significantly increase the parameters.

avoided at the cost of tailoring the results to a class of signature schemes from specific canonical identification schemes in the ROM.

Subsequently, we will first recall a definition of multi-user EUF-CMA and then prove Theorem 6, which formalizes the main result of this section.

Definition 30 (MU-EUF-CMA). *A signature scheme* Σ *is* MU-EUF-CMA *secure, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\varepsilon(\cdot)$ *such that*

$$\mathsf{Pr}\left[\begin{matrix}\{(\mathsf{sk}_i,\mathsf{pk}_i)\leftarrow\mathsf{KeyGen}(1^\kappa)\}_{i\in[\mathsf{poly}(\kappa)]},\\ (i^\star,m^\star,\sigma^\star)\leftarrow\mathcal{A}^{\mathsf{Sign}(\cdot,\cdot)}(\{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]})\end{matrix} : \begin{matrix}\mathsf{Verify}(\mathsf{pk}_{i^\star},m^\star,\sigma^\star)=1\ \wedge\\ (i^\star,m^\star)\notin\mathcal{Q}^{\mathsf{Sign}}\end{matrix}\right]\leq\varepsilon(\kappa),$$

where $\mathsf{Sign}(i,m)\coloneqq\Sigma.\mathsf{Sign}(\mathsf{sk}_i,m)$ *and the environment keeps track of the queries to the signing oracle via* $\mathcal{Q}^{\mathsf{Sign}}$*.*

Theorem 6. *Let* $\Sigma=(\mathsf{KeyGen},\mathsf{Sign},\mathsf{Verify})$ *be a signature scheme which provides adaptability of signatures where the success ratio of any* EUF-CMA *adversary is* ρ*. Then the success ratio of any adversary against* MU-EUF-CMA *of* $\Sigma'=(\mathsf{KeyGen}',\mathsf{Sign}',\mathsf{Verify}')$ *is* $\rho'\approx\rho$*, where* $\mathsf{KeyGen}'(1^\kappa)\coloneqq\mathsf{KeyGen}(1^\kappa)$*,* $\mathsf{Sign}'(\mathsf{sk},m)\coloneqq\mathsf{Sign}(\mathsf{sk},\mu(\mathsf{sk})||m)$*, and* $\mathsf{Verify}(\mathsf{pk},m,\sigma)\coloneqq\mathsf{Verify}(\mathsf{pk},\mathsf{pk}||m,\sigma)$*.*

Proof. First, our reduction $\mathcal{R}$ obtains a public key pk_1 from an EUF-CMA challenger $\mathcal{C}$ and initializes an empty list $\mathtt{SK}$. It sets $\mathtt{SK}[1]\leftarrow 0$, and for $2\leq i\leq\mathsf{poly}(\kappa)$, it chooses $\mathtt{SK}[i]\xleftarrow{R}\mathbb{H}$, and sets $\mathsf{pk}_i\leftarrow\mathsf{pk}_1\cdot\mu(\mathtt{SK}[i])$. Then, it starts $\mathcal{A}$ on $\{\mathsf{pk}_i\}_{i\in[\mathsf{poly}(\kappa)]}$ and simulates Sign' inside the $\mathsf{Sign}(\cdot,\cdot)$ oracle as follows (where $\mathcal{C}.\mathsf{Sign}(\cdot)$ denotes the signing oracle provided by $\mathcal{C}$).

$\mathsf{Sign}(i,m)$: Obtain $\sigma\leftarrow\mathcal{C}.\mathsf{Sign}(\mathsf{pk}_i||m)$, compute $(\mathsf{pk}_i,\sigma')\leftarrow\mathsf{Adapt}(\mathsf{pk}_1,\mathsf{pk}_i||m,\sigma,\mathtt{SK}[i])$, and return σ'.

Eventually, $\mathcal{A}$ outputs a forgery $(i^\star,m^\star,\sigma^\star)$, where $(i^\star,m^\star)\notin\mathcal{Q}^{\mathsf{Sign}}$ by definition. Thus, $\mathcal{R}$ has never sent $\mathsf{pk}_{i^\star}||m^\star$ to the sign oracle of $\mathcal{C}$ and can obtain $(\mathsf{pk}_1,\sigma'^\star)\leftarrow\mathsf{Adapt}(\mathsf{pk}_{i^\star},\mathsf{pk}_{i^\star}||m^\star,\sigma^\star,-\mathtt{SK}[i])$ and output $(\mathsf{pk}_{i^\star}||m^\star,\sigma'^\star)$ as an EUF-CMA forgery. Due to adaptability of signatures the simulation of the oracle is perfect; the running time of $\mathcal{R}$ is approximately the same as the running time of $\mathcal{A}$ which concludes the proof. □

Remark 3. It is straight forward to prove such an implication for weaker unforgeability notions. Essentially the security proof would be analogous, but without the need to simulate the signing oracle. Furthermore, it is important to note that for key-recovery attacks, where no signatures need to be simulated, a secret key to public key homomorphism would be sufficient to tightly relate the single-user setting to the key-prefixed multi-user setting.

8 Summary and Conclusion

In this paper we introduce a definitional framework distilling various natural flavours of key-homomorphisms for signatures, and, classify a set of existing

signature schemes according to them. We present simple compilers turning signature schemes admitting particular key-homomorphisms into ring signatures, universal designated verifier signatures, simulation-sound extractable NIZK argument systems, as well as multisignatures. Furthermore, we show that using key-homomorphisms allows us to prove a tight implication from single-user security to key-prefixed multi-user security for the class of schemes admitting a certain key-homomorphism. In Figure 1 we subsume the various relations and implications obtained when using the formalisms provided by our definitional framework. Plugging in concrete signature schemes into our compilers yields to

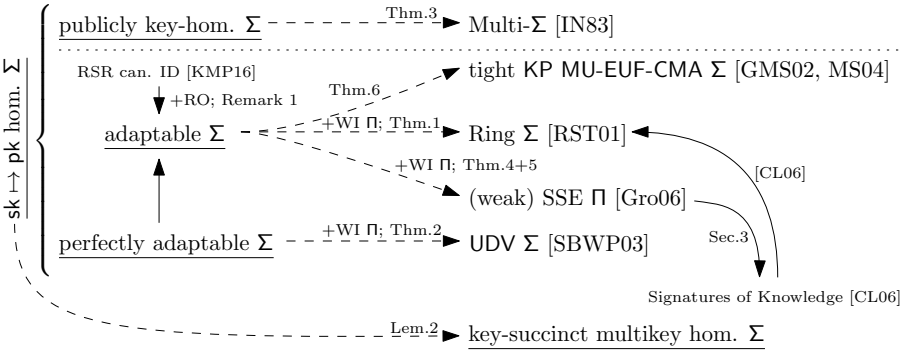

Fig. 1. Summary of the contributions and their relations. Legend: underlined... variant of key-homomorphism introduced in this paper, ——... trivial implication, – – –... shown in this paper, sk ↦ pk hom.... secret-key to public-key homomorphism, Σ... signature scheme, RSR can. ID... random-self-reducible canonical identification scheme, RO... random oracle, WI... witness indistinguishable, Π... non interactive argument system, KP... key-prefixed, MU... multi-user, EUF-CMA... existential unforgeability under chosen message attacks, SSE... simulation-sound extractable NIZKs, UDV... universal designated verifier.

novel instantiations of the various schemes, including instantiations under standard assumptions without random oracles and even standard model instantiations under standard assumptions. They favorably compare to existing instantiations regarding conceptual simplicity and efficiency. Furthermore, our results attest the tight multi-user security of various schemes which were previously unknown to provide tight multi-user security. We subsume these instantations in Table 3.

Compiler	**Hom**	Schnorr	GQ	BLS	Katz-Wang	Waters	PS	AGOT	Gha
Ring Signatures	A	✓	✓	✓	✓	✓	✓	✓	✓
UDV Signatures	A/PA	✓‡	✓‡	✓	✓‡	✓	✓	✓	✓
SSE Π	A	✓	✓	✓	✓	✓	✓	✓	✓
Multisignatures	PKH	×	×	✓	×	✓	×	×	×
tight KP MU-EUF-CMA	A	✓	✓	✓	✓	✓	✓	✓	✓

Table 3. Possible instantiations of our compilers. ‡... compiler achieves non-transferability privacy (strong non-transferability privacy requires perfect adaption).

Finally, we investigate the notion of multikey-homomorphic signatures in context of key-homomorphisms and show that a secret-key to public-key homo-

morphism implies the existence of key-succinct multikey-homomorphic signatures. We consider it to be interesting to find constructions of the various flavors of multikey-homomorphic signatures, but as the focus in this paper lies on key-homomorphisms, we leave a thorough investigation as future work.

While we already presented quite some applications of our framework for key-homomorphic signatures, we believe that our framework will find much more applications. This argument is underpinned by the fact that there is already a first work presenting an additional application [PMT17].

Acknowledgements. The authors have been supported by EU H2020 project PRISMACLOUD, grant agreement n°644962. We thank various anonymous referees for their valuable comments.

References

[ABC+12] Jae Hyun Ahn, Dan Boneh, Jan Camenisch, Susan Hohenberger, Abhi Shelat, and Brent Waters. Computing on authenticated data. In *TCC*, 2012.

[ADK+13] Masayuki Abe, Bernardo David, Markulf Kohlweiss, Ryo Nishimaki, and Miyako Ohkubo. Tagged one-time signatures: Tight security and optimal tag size. In *PKC*, pages 312–331, 2013.

[AFG+10] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In *CRYPTO*, 2010.

[AGOT14] Masayuki Abe, Jens Groth, Miyako Ohkubo, and Mehdi Tibouchi. Structure-preserving signatures from type II pairings. In *Advances in Cryptology - CRYPTO 2014*, pages 390–407, 2014.

[AHI11] Benny Applebaum, Danny Harnik, and Yuval Ishai. Semantic Security under Related-Key Attacks and Applications. In *ICS*, 2011.

[ALP12] Nuttapong Attrapadung, Benoît Libert, and Thomas Peters. Computing on Authenticated Data: New Privacy Definitions and Constructions. In *ASIACRYPT*, 2012.

[BBL17] Fabrice Benhamouda, Florian Bourse, and Helger Lipmaa. CCA-secure inner-product functional encryption from projective hash functions. PKC, 2017.

[BCC+15] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, Jens Groth, and Christophe Petit. Short accountable ring signatures based on DDH. In *ESORICS*, 2015.

[BCM11] Mihir Bellare, David Cash, and Rachel Miller. Cryptography Secure against Related-Key Attacks and Tampering. In *ASIACRYPT*, 2011.

[Ber15] Daniel J. Bernstein. Multi-user schnorr security, revisited. *IACR Cryptology ePrint Archive*, 2015, 2015.

[BF11] Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. In *EUROCRYPT*. 2011.

[BFG13] David Bernhard, Georg Fuchsbauer, and Essam Ghadafi. Efficient signatures of knowledge and DAA in the standard model. In *ACNS*, 2013.

[BFKW09] Dan Boneh, David Mandell Freeman, Jonathan Katz, and Brent Waters. Signing a Linear Subspace: Signature Schemes for Network Coding. In *PKC*, 2009.

[BFP+15] Abhishek Banerjee, Georg Fuchsbauer, Chris Peikert, Krzysztof Pietrzak, and Sophie Stevens. Key-Homomorphic Constrained Pseudorandom Functions. In *TCC*, 2015.

[BFS14] Xavier Boyen, Xiong Fan, and Elaine Shi. Adaptively secure fully homomorphic signatures based on lattices. Cryptology ePrint Archive, Report 2014/916, 2014.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully Key-Homomorphic Encryption, Arithmetic Circuit ABE and Compact Garbled Circuits. In *EUROCRYPT*, 2014.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *PKC*, 2014.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT*, 2003.

[BJ08] Ali Bagherzandi and Stanislaw Jarecki. Multisignatures using proofs of secret key possession, as secure as the diffie-hellman problem. In *SCN*, 2008.

[BJL16] Fabrice Benhamouda, Marc Joye, and Benoît Libert. A new framework for privacy-preserving aggregation of time-series data. *ACM Trans. Inf. Syst. Secur.*, 18(3), 2016.

[BJLS16] Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In *EUROCRYPT*, 2016.

[BK10] Zvika Brakerski and Yael Tauman Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. *IACR Cryptology ePrint Archive*, 2010.

[BKM09] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology*, 22(1), 2009.

[BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key Homomorphic PRFs and Their Applications. In *CRYPTO*, 2013.

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4), 2004.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In *PKC*, 2003.

[BP14] Abhishek Banerjee and Chris Peikert. New and Improved Key-Homomorphic Pseudorandom Functions. In *CRYPTO*, 2014.

[BPT12] Mihir Bellare, Kenneth G. Paterson, and Susan Thomson. RKA security beyond the linear barrier: Ibe, encryption and signatures. In *ASIACRYPT*, 2012.

[Cat14] Dario Catalano. Homomorphic signatures and message authentication codes. In *SCN*, 2014.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, 1994.

[CFW14] Dario Catalano, Dario Fiore, and Bogdan Warinschi. Homomorphic signatures with efficient verification for polynomial functions. In *CRYPTO*. 2014.

[CGS07] Nishanth Chandran, Jens Groth, and Amit Sahai. Ring signatures of sub-linear size without random oracles. In *ICALP*, 2007.

[CHKM10] Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3), 2010.

[CL06] Melissa Chase and Anna Lysyanskaya. On signatures of knowledge. In *CRYPTO*, 2006.

[DHLW10] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Efficient public-key cryptography in the presence of key leakage. In *ASIACRYPT*, pages 613–631, 2010.

[DKNS04] Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup. Anonymous identification in ad hoc groups. In *EUROCRYPT*, 2004.

[DKS16] David Derler, Stephan Krenn, and Daniel Slamanig. Signer-Anonymous Designated-Verifier Redactable Signatures for Cloud-Based Data Sharing. In *CANS*, 2016.

[DMS16] Yevgeniy Dodis, Ilya Mironov, and Noah Stephens-Davidowitz. Message transmission with reverse firewalls - secure communication on corrupted machines. In *CRYPTO*, 2016.

[DS18] David Derler and Daniel Slamanig. Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. *Des. Codes Cryptogr.*, 2018.

[EG14] Alex Escala and Jens Groth. Fine-tuning groth-sahai proofs. In *PKC*, 2014.

[FF13] Marc Fischlin and Nils Fleischhacker. Limitations of the meta-reduction technique: The case of schnorr signatures. In *EUROCRYPT*, 2013.

[FHS15] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In *CRYPTO*, 2015.

[FKM+16] Nils Fleischhacker, Johannes Krupp, Giulio Malavolta, Jonas Schneider, Dominique Schröder, and Mark Simkin. Efficient Unlinkable Sanitizable Signatures from Signatures with Re-randomizable Keys. In *PKC*, 2016.

[FMNP16] Dario Fiore, Aikaterini Mitrokotsa, Luca Nizzardo, and Elena Pagnin. Multi-key homomorphic authenticators. In *ASIACRYPT*, 2016.

[Fre12] David Mandell Freeman. Improved security for linearly homomorphic signatures: A generic framework. 2012.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, 1986.

[Gen09] Craig Gentry. Fully Homomorphic Encryption using Ideal Lattices. In *STOC*, 2009.

[Gha16] Essam Ghadafi. Short structure-preserving signatures. In *CT-RSA 2016*, pages 305–321, 2016.

[GHKW16] Romain Gay, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. Tightly cca-secure encryption without pairings. In *EUROCRYPT*, 2016.

[GJKW07] Eu-Jin Goh, Stanislaw Jarecki, Jonathan Katz, and Nan Wang. Efficient signature schemes with tight reductions to the diffie-hellman problems. *J. Cryptology*, 20(4), 2007.

[GK15] Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In *EUROCRYPT*, 2015.

[GK16] Shafi Goldwasser and Yael Tauman Kalai. Cryptographic assumptions: A position paper. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 505–522, 2016.

[GLW12] Shafi Goldwasser, Allison B. Lewko, and David A. Wilson. Bounded-Collusion IBE from Key Homomorphism. In *TCC*, 2012.

[GMS02] Steven D. Galbraith, John Malone-Lee, and Nigel P. Smart. Public key signatures in the multi-user setting. *Inf. Process. Lett.*, 83(5), 2002.

[GMY03] Juan A. Garay, Philip D. MacKenzie, and Ke Yang. Strengthening zero-knowledge protocols using signatures. In *EUROCRYPT*, 2003.

[GMY06] Juan A. Garay, Philip D. MacKenzie, and Ke Yang. Strengthening zero-knowledge protocols using signatures. *J. Cryptology*, 19(2), 2006.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[GQ88] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *CRYPTO*, pages 216–231, 1988.

[Gro06] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT*, 2006.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, 2008.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In *STOC*, 2015.

[HS14] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In *ASIACRYPT*, 2014.

[IN83] K. Itakura and K. Nakamura. A public-key cryptosystem suitable for digital multisignatures. *NEC Research & Development*, 71, 1983.

[JMSW02] Robert Johnson, David Molnar, Dawn Xiaodong Song, and David Wagner. Homomorphic signature schemes. In *CT-RSA*, 2002.

[JSI96] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In *EUROCRYPT*, 1996.

[Kat10] Jonathan Katz. *Digital Signatures*. Springer, 2010.

[KMP16] Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In *CRYPTO*, 2016.

[KW03] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In *CCS*, 2003.

[Lac18] Marie-Sarah Lacharité. Security of BLS and BGLS signatures in a multi-user setting. *Cryptography and Communications*, 10(1):41–58, 2018.

[LOS+06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In *EUROCRYPT*, 2006.

[LTV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *STOC*, 2012.

[LTWC18] Russell W. F. Lai, Raymond K. H. Tai, Harry W. H. Wong, and Sherman S. M. Chow. Multi-key homomorphic signatures unforgeable under insider corruption. In *ASIACRYPT*, 2018. To Appear.

[Lyu08] Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In *Public Key Cryptography - PKC 2008, 11th International Workshop on Practice and Theory in Public-Key Cryptography, Barcelona, Spain, March 9-12, 2008. Proceedings*, pages 162–179, 2008.

[MS04] Alfred Menezes and Nigel P. Smart. Security of signature schemes in a multi-user setting. *Des. Codes Cryptography*, 33(3), 2004.

[MS17] Giulio Malavolta and Dominique Schröder. Efficient ring signatures in the standard model. In *Advances in Cryptology - ASIACRYPT 2017*, pages 128–157, 2017.

[MSM+15] Hiraku Morita, Jacob C. N. Schuldt, Takahiro Matsuda, Goichiro Hanaoka, and Tetsu Iwata. On the security of the schnorr signature scheme and DSA against related-key attacks. In *ICISC*, 2015.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, pages 96–109, 2003.

[PMT17] Elena Pagnin, Aikaterini Mitrokotsa, and Keisuke Tanaka. Anonymous single-round server-aided verification. Cryptology ePrint Archive, Report 2017/794, to appear at Latincrypt 2017, 2017.

[PS16] David Pointcheval and Olivier Sanders. Short Randomizable Signatures. In *CT-RSA*, 2016.

[Rot11] Ron Rothblum. Homomorphic Encryption: From Private-Key to Public-Key. In *TCC*, 2011.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *ASIACRYPT*, 2001.

[RY07] Thomas Ristenpart and Scott Yilek. The Power of Proofs-of-Possession: Securing Multiparty Signatures against Rogue-Key Attacks. In *EUROCRYPT*, 2007.

[SBWP03] Ron Steinfeld, Laurence Bull, Huaxiong Wang, and Josef Pieprzyk. Universal designated-verifier signatures. In *ASIACRYPT*, 2003.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3), 1991.

[SS08] Siamak Fayyaz Shahandashti and Reihaneh Safavi-Naini. Construction of universal designated-verifier signatures and identity-based signatures from standard signatures. In *PKC*, 2008.

[ST01] Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In *CRYPTO*, pages 355–367, 2001.

[TW14] Stefano Tessaro and David A. Wilson. Bounded-collusion identity-based encryption from semantically-secure public-key encryption: Generic constructions with short ciphertexts. In *PKC*, 2014.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, 2005.

A Homomorphisms on Key and Message Space

As already mentioned in Section 1, signature schemes with homomorphic properties on their message space [JMSW02] are well known. With such schemes, it is possible for anyone to derive a signature for a message m' from signatures on messages $(m_i)_{i\in[n]}$ under some public key pk as long as $m' = f(m_1, \ldots, m_n)$ for $f \in \mathcal{F}$, where $\mathcal{F}$ is the set of so called admissible functions (determined by the scheme). Among others (cf. [ABC+12, ALP12]) there are schemes for linear functions [BFKW09, Fre12], polynomial functions of higher degree [BF11, CFW14] and meanwhile even (leveled) fully homomorphic signatures supporting arbitrary functions [GVW15, BFS14]. However, these constructions consider these

homomorphisms under *a single* key. While in context of encryption, constructions working with distinct keys, i.e., so called multikey-homomorphic encryption schemes (e.g., [LTV12]), are known, such a feature have only been studied recently in this work and other independent concurrent works.

Independent and Concurrent Work. Fiore et al. [FMNP16] introduced the concept of multikey-homomorphic authenticators, which also covers multikey-homomorphic signatures. They present a construction of multikey-homomorphic signatures from standard lattices based on the fully homomorphic signatures in [GVW15]. Their model and construction focuses on achieving succinct combined signatures, whereas the focus of our construction (feasibility result) is on achieving succinct combined keys. We also note that the independent and concurrent work of Lai et al. [LTWC18], among others, study multikey-homomorphic signatures with succinct combined keys and signatures even under stronger security model supporting insider corruption. To achieve this, they require rather heavy tools (and assumptions) such as zk-SNARKS. We stress that the aforementioned works do not focus on constructing multikey-homomorphic signatures with signatures supporting key-homomorphisms.

A.1 Multikey-Homomorphic Signatures from Key-Homomorphisms

In this section we initiate the study of so called multikey-homomorphic signatures and in particular propose a definitional framework for such schemes that support a homomorphic property on the message space under *distinct* keys. Our focus is then on investigating key-homomorphic properties in this context.

Below we present and discuss what we call *multikey-homomorphic signatures*, where the homomorphic property on the message space is defined with respect to a class $\mathcal{F}$ of admissible functions (e.g., represented as arithmetic circuits). In contrast to the notions introduced above, which capture additional properties of conventional signature schemes, multikey-homomorphic signatures are a separate building block. To this end we explicitly formalize the algorithms as well as the required correctness and unforgeability notion. We stress that, as the focus of this work lies on key-homomorphic schemes, we will also focus on these aspects in this section. In particular, while we present a general definition of multikey-homomorphic schemes[20], we focus on schemes who use a *succinct* representation of a combined public key in the verification.

Definition 31 (Multikey-Homomorphic Signatures). *A multikey-homomorphic signature scheme for a class $\mathcal{F}$ of admissible functions, is a tuple of the following PPT algorithms:*

$\mathsf{PGen}(1^\kappa)$: *Takes a security parameter κ as input, and outputs parameters* pp.
$\mathsf{KeyGen}(\mathsf{pp})$: *Takes parameters* pp *as input, and outputs a keypair* $(\mathsf{sk}, \mathsf{pk})$ *(we assume that* pp *is included in* pk*).*

[20] Our definition is analogous to the encryption case (e.g., [LTV12]), where the input of a set of public keys into the verification of a combined signature is supported.

$\mathsf{Sign}(\mathsf{sk}, m, \tau)$: *Takes a secret key* sk*, a message* m*, and a tag* τ *as input, and outputs a signature* σ*.*

$\mathsf{Verify}(\mathsf{pk}, m, \sigma, \tau)$: *Takes a public key* pk *a message* m*, a signature* σ*, and a tag* τ *as input, and outputs a bit* b*.*

$\mathsf{Combine}((\mathsf{pk}_i)_{i\in[n]}, (m_i)_{i\in[n]}, f, (\sigma_i)_{i\in[n]}, \tau)$: *Takes public keys* $(\mathsf{pk}_i)_{i\in[n]}$*, messages* $(m_i)_{i\in[n]}$*, a function* $f \in \mathcal{F}$*, signatures* $(\sigma_i)_{i\in[n]}$*, and a tag* τ *as input, and outputs a public key* $\hat{\mathsf{pk}}$ *and a signature* $\hat{\sigma}$*.*

$\mathsf{Verify}'(\hat{\mathsf{pk}}, \hat{m}, f, \hat{\sigma}, \tau)$: *Takes a combined public key* $\hat{\mathsf{pk}}$*, a message* $\hat{m}$*, a function* f*, a signature* $\hat{\sigma}$*, and a tag* τ *as input, and outputs a bit* b*.*

Subsequently, we formalize the security properties one would expect from such schemes.

Definition 32 (Correctness). *A multikey-homomorphic signature scheme for a class* $\mathcal{F}$ *of admissible functions is correct, if for all security parameters* κ*, for all* $1 \leq n \leq \mathsf{poly}(\kappa)$*, all* $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(1^\kappa))_{i\in[n]}$*, all messages* $(m_i)_{i\in[n]}$*, all tags* τ*, all functions* $f \in \mathcal{F}$*, all functions* $f' \notin \mathcal{F}$*, and all signatures* $(\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m_i, \tau))_{i=1}^n$ *and results* $(\hat{\mathsf{pk}}, \hat{\sigma}) \leftarrow \mathsf{Combine}((\mathsf{pk}_i)_{i\in[n]}, (m_i)_{i\in[n]}, f, (\sigma_i)_{i\in[n]}, \tau)$ *it holds that*

$$\begin{aligned}&(\mathsf{Verify}(\mathsf{pk}_i, m_i, \sigma_i, \tau) = 1)_{i\in[n]} \ \wedge \ (\mathsf{pk}_i \in \hat{\mathsf{pk}})_{i\in[n]} \ \wedge \\ &\mathsf{Verify}'(\hat{\mathsf{pk}}, \hat{m}, f, \hat{\sigma}, \tau) = 1 \ \wedge \ \mathsf{Verify}'(\cdot, \cdot, f', \cdot, \cdot) = 0,\end{aligned}$$

where $\hat{m} = f(m_1, \ldots, m_n)$*.*

We note that the predicate "$\in$" for the check $\mathsf{pk} \in \hat{\mathsf{pk}}$ needs to be explicitly defined by every concrete scheme (i.e., it is not necessarily a simple set membership check).

Definition 33 (Unforgeability). *A multikey-homomorphic signature scheme for a class* $\mathcal{F}$ *of admissible functions is unforgeable, if for every PPT adversary* $\mathcal{A}$ *there exists a negligible function* $\epsilon(\cdot)$ *such that it holds that*

$$\mathsf{Pr}\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathsf{PGen}(1^\kappa), \\ (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}), \\ \mathcal{O} \leftarrow \{\mathsf{Sig}(\cdot,\cdot)\}, \\ (\hat{\mathsf{pk}}^\star, \hat{m}^\star, f^\star, \hat{\sigma}^\star, \tau^\star) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk}),\end{array} : \begin{array}{r}\mathsf{Verify}'(\hat{\mathsf{pk}}^\star, \hat{m}^\star, f^\star, \hat{\sigma}^\star, \tau^\star) = 1 \ \wedge \\ \big(\mathsf{pk} \in \hat{\mathsf{pk}}^\star \ \wedge \ \nexists \ m \in \mathcal{M} : \\ (\hat{m}^\star \in \mathsf{R}(f^\star(\cdots, m, \cdots)) \ \wedge \\ (m, \tau^\star) \in \mathcal{Q}^{\mathsf{Sig}})\big) \ \vee \ \hat{m}^\star \notin \mathsf{R}(f^\star)\end{array}\right] \leq \epsilon(\kappa),$$

where $\mathsf{Sig}(m, \tau) := \mathsf{Sign}(\mathsf{sk}, m, \tau)$ *and* $\mathcal{Q}^{\mathsf{Sig}}$ *records the* Sig *queries.*

Observe that Definition 31 neither puts restrictions on the size of signatures $\hat{\sigma}$ nor public keys $\hat{\mathsf{pk}}$. To really benefit from the functionality provided by multikey-homomorphic signatures, one may additionally require that $\hat{\mathsf{pk}}$ is succinct. Inspired by [BGI14], we subsequently provide a formal definition.

Definition 34 (Key Succinctness). *A multikey-homomorphic signature scheme is called key succinct, if for all* $\kappa \in \mathbb{N}$*, for all* $n \leq \mathsf{poly}(\kappa)$*, for all* $\mathsf{pp} \leftarrow \mathsf{PGen}(1^\kappa)$*, for all* $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}))_{i\in[n]}$*, for all* $(m_i)_{i\in[n]} \in \mathcal{M}^n$*, all*

$(\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m_i))_{i\in[n]}$, *all* $(\hat{\mathsf{pk}}, \hat{\sigma}) \leftarrow \mathsf{Combine}((\mathsf{pk}_i)_{i\in[n]}, (m_i)_{i\in[n]}, f, (\sigma_i)_{i\in[n]})$ *it holds that*

$$|\hat{\mathsf{pk}}| \leq \mathsf{poly}(\kappa).$$

It turns out that secret key to public key homomorphic signature schemes already imply the existence of key succinct multikey-homomorphic signature schemes for a class $\mathcal{F}$ of functions with polynomially many members.

Lemma 21. *If there exists an* EUF-CMA *secure secret key to public key homomorphic signature scheme* Σ*, then there exists a key succinct multikey-homomor-phic signature scheme* $\Sigma_{\mathcal{F}}$ *for a class* $\mathcal{F}$ *of functions with polynomially many members.*

Proof. We prove this lemma by constructing such a scheme. In particular, we base the construction on a wrapped version $\Sigma_{\mathcal{F}} = (\mathsf{KeyGen}_{\mathcal{F}}, \mathsf{Sign}_{\mathcal{F}}, \mathsf{Verify}_{\mathcal{F}})$ of the secret key to public key homomorphic signature scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$, where $\mathsf{KeyGen}_{\mathcal{F}}(1^\kappa) := \mathsf{KeyGen}(1^\kappa)$, $\mathsf{Sign}_{\mathcal{F}}(\mathsf{sk}, m, \tau) := \mathsf{Sign}(\mathsf{sk}, m||\tau||\mathcal{F})$ and $\mathsf{Verify}_{\mathcal{F}}(\mathsf{pk}, m, \sigma, \tau) := \mathsf{Verify}(\mathsf{pk}, m||\tau||\mathcal{F}, \sigma)$. Then $\mathsf{Combine}$ and Verify' can be defined as follows:

$\mathsf{Combine}((\mathsf{pk}_i)_{i\in[n]}, (m_i)_{i\in[n]}, f, (\sigma_i)_{i\in[n]}, \tau)$: If $f \notin \mathcal{F}$ return $\perp$. Otherwise, compute $\hat{\sigma} \leftarrow ((\mathsf{pk}_i, m_i, \sigma_i))_{i\in[n]}$ and $\hat{\mathsf{pk}} \leftarrow \prod_{i=1}^n \mathsf{pk}_i$ and return $\hat{\mathsf{pk}}$ and $\hat{\sigma}$.

$\mathsf{Verify}'(\hat{\mathsf{pk}}, \hat{m}, f, \hat{\sigma}, \tau)$: Return 1, if $(\mathsf{Verify}_{\mathcal{F}}(\mathsf{pk}_i, m_i, \sigma_i, \tau) = 1)_{i\in[n]} \ \wedge \ \hat{m} = f(m_1, \ldots, m_n) \ \wedge \ \hat{\mathsf{pk}} = \prod_{i=1}^n \mathsf{pk}_i \ \wedge \ f \in \mathcal{F}$, and 0 otherwise.

It is immediate that correctness holds. For unforgeability, note that since $\mathsf{Verify}'(\hat{\mathsf{pk}}^\star, \hat{m}^\star, f^\star, \hat{\sigma}^\star, \tau^\star) = 1$ by definition, we know that $\hat{\mathsf{pk}} = \prod_{i\in[n]} \mathsf{pk}_i$, where $(\mathsf{pk}_i)_{i\in[n]}$ is contained in the signature. Thus, we can simply engage with an EUF-CMA challenger to obtain pk and simulate the game without knowing sk by using the Sign oracle provided by the EUF-CMA challenger. If the adversary eventually outputs a forgery, we either have an EUF-CMA forgery which happens with negligible probability or a message $\hat{m}^\star \notin \mathsf{R}(f^\star)$ which happens with probability 0 as Verify' does not accept such an input. Thus, the overall success probability of any PPT adversary is negligible. □

While this proves the existence of key succinct multikey-homomorphic signatures, one could also ask for signature succinctness as defined below.

Definition 35 (Signature Succinctness). *A multikey-homomorphic signature scheme is called signature succinct, if for all* $\kappa \in \mathbb{N}$*, for all* $n \leq \mathsf{poly}(\kappa)$*, for all* $\mathsf{pp} \leftarrow \mathsf{PGen}(1^\kappa)$*, for all* $((\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}))_{i\in[n]}$*, for all* $(m_i)_{i\in[n]} \in \mathcal{M}^n$*, all* $(\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m_i))_{i\in[n]}$*, all* $(\hat{\mathsf{pk}}, \hat{\sigma}) \leftarrow \mathsf{Combine}((\mathsf{pk}_i)_{i\in[n]}, (m_i)_{i\in[n]}, f, (\sigma_i)_{i\in[n]})$ *it holds that*

$$|\hat{\sigma}| \leq \mathsf{poly}(\kappa).$$

It seems non-trivial to construct schemes under mild assumptions that satisfy both succinctness definitions, i.e., provide succinct keys *and* signatures. We suspect, that this requires a trick à la Dodis et al. [DKNS04] which was used in the context of constructing constant-size ring signatures. Dodis et al. argue

that the sets of keys (rings) will often be determined a priori in practice and can thus be represented by some compact description. Consequently, only a compact description of such a set of keys is required in the actual signature and the linear dependency on the ring size is removed. A similar argumentation also holds in many application scenarios for multi-key homomorphic signatures (e.g., scenarios where the joint verification key $\hat{\mathsf{pk}}$ is already pre-distributed as also discussed below). We note that when moving to very strong assumptions such as knowledge assumptions, as in Lai et al. [LTWC18], then schemes that satisfy both succinctness definitions can indeed be constructed.