Key-Homomorphic Signatures and Applications to Multiparty Signatures

David Derler and Daniel Slamanig

IAIK, Graz University of Technology, Austria
{david.derler|daniel.slamanig}@tugraz.at

Abstract. Key-homomorphic properties of cryptographic objects 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 generalize larger classes of existing signature schemes, which makes it possible to infer general statements about signature schemes from those classes by simply making black-box usage of the respective properties. We then employ our definitional framework to show elegant and simple compilers from classes of schemes satisfying different types of key-homomorphisms to a number of other interesting primitives such as ring signature schemes, (universal) designated verifier signature schemes and multisignature schemes.

Moreover, we introduce the notion of multikey-homomorphic signatures. Such schemes provide homomorphic properties on the message space of signatures under different keys. We discuss key-homomorphisms in this context and present some first constructive results from key-homomorphic schemes. Finally, we discuss some interesting open problems and an application of multikey-homomorphic schemes to verifiable delegation of computations.

Keywords. key-homomorphic signatures · ring signatures · (universal) designated verifier signatures · multisignatures · multikey-homomorphic signatures

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

The authors have been supported by EU H2020 project Prismacloud, grant agreement n°644962.

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. Specifically, we study key-homomorphic signature schemes, that is, signature schemes which are homomorphic with respect to the key space. As we will show in this paper, this concept turns out to be a very interesting and versatile tool.

Previous Work. While we are the first to explicitly study 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 bounded-collusion identity-based encryption (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.

When switching to the field of signatures, we can define key-homomorphisms in various different ways, of which we subsequently sketch two 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]. In such schemes the $\mathsf{EUF\text{-}CMA}$ security notion is slightly tweaked, by additionally allowing the adversary to see signatures under re-randomized keys. Such 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 this context, 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]).

Contribution. Now, we briefly summarize the contributions in this paper:

– We initiate the study of key-homomorphic signature schemes. In doing so, we propose various natural definitions of key-homomorphic signatures, generalizing larger classes of existing signature schemes. This generalization makes it possible to infer general statements about signature schemes from those classes by simply making black-box usage of the respective properties. Thereby, we rule out certain combinations of key-homomorphism and unforgeability notions.
– We employ our definitional framework to present compilers from classes of schemes providing different types of key-homomorphisms to other interesting variants of signature schemes such as ring signatures, (universal) designated verifier signatures or multisignatures. The so obtained constructions, besides being very efficient, are simple and elegant from a construction and security analysis point of view. Basically, for ring signatures and (universal) designated verifier signatures, 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*. Multisignatures are directly implied by signatures with certain key-homomorphic properties.
– We introduce the notion of multikey-homomorphic signatures. Such schemes provide homomorphic properties on the message space of signatures under different keys. This can be seen as a step towards establishing the signature

counterpart of multikey (fully) homomorphic encryption [LTV12, CM15, MW16, PS16a, BP16]. We discuss key-homomorphisms in this context and present some first constructive results from key-homomorphic signatures that yield multikey-homomorphic signatures with a succinct verification key. Finally, we discuss some interesting open problems and highlight that multikey-homomorphic signatures have interesting applications in verifiable delegation of computations.

- As a contribution of independent interest, we strengthen the security model of universal designated verifier signatures by proposing a stronger designated verifier unforgeability notion, which we term simulation-sound designated verifier unforgeability. We prove that schemes obtained from our compiler satisfy this strong notion, i.e., we can use a certain class of key-homomorphic signatures in a black-box way to convert them to universal designated verifier signatures which are secure in this strengthened model. This yields numerous instantiations being the first satisfying such a strong notion.

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, $y \overset{R}{\leftarrow} S$ means that y is sampled uniformly at random from a set S and we use $\mathcal{Q} \overset{\cup}{\leftarrow} z$ 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 $\varepsilon(\cdot) : \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}$ is called negligible, iff it vanishes faster than every inverse polynomial, i.e., $\forall\ k : \exists\ n_k : \forall\ n > n_k : \varepsilon(n) < n^{-k}$. 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. *A function $f : \mathsf{Dom}(f) \rightarrow \mathsf{R}(f)$ is called a one-way function, if (1) there exists a PPT algorithm $\mathcal{A}_1$ so that $\forall\ x \in \mathsf{Dom}(f) : \mathcal{A}_1(x) = f(x)$, and if (2) for every PPT algorithm $\mathcal{A}_2$ there is a negligible function $\varepsilon(\cdot)$ such that it holds that*

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

Unless stated otherwise, we assume $\mathsf{Dom}(f)$ to be efficiently sampleable.

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

Definition 2. *A signature scheme Σ is a triple* (KeyGen, Sign, 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 assume that PP is included in all public keys.

Besides the usual correctness property, Σ needs to provide some unforgeability notion. Below, we present two standard notions required in our context (ordered from weak to strong). We start with universal unforgeabiltity under no message attacks ($\mathsf{UUF\text{-}NMA}$ security).

Definition 3 ($\mathsf{UUF\text{-}NMA}$). *A signature scheme Σ is* $\mathsf{UUF\text{-}NMA}$ *secure, if for all PPT adversaries $\mathcal{A}$ there is a negligible function $\varepsilon(\cdot)$ such that*

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

The most common notion is existential unforgeability under adaptively chosen message attacks ($\mathsf{EUF\text{-}CMA}$ security).

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

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

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

Non-Interactive Proof Systems. Now, we recall a standard definition of non-interactive proof systems (Π). 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 5. *A non-interactive proof system Π is a tuple of algorithms* (Setup, Proof, 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* $b \in \{0, 1\}$*.*

We note that Proof is not required to run in polynomial time. If it, however, is required we talk about a non-interactive argument system. We require Π to be complete, sound, and adaptively witness-indistinguishable. Subsequently, we recall formal definition of those properties.

Definition 6 (Completeness). *A non-interactive proof system* Π *is* complete*, if for every adversary* $\mathcal{A}$ *it holds that*

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

Definition 7 (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*

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

If $\varepsilon = 0$, we have perfect soundness.

Definition 8 (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 $\mathcal{P}(\mathsf{crs}, x, w_0, w_1, b) := \mathsf{Proof}(\mathsf{crs}, x, w_b)$*, 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. Furthermore, we require Π to admit proofs of knowledge, which are defined as follows.

Definition 9 (Proof of Knowledge). *A non-interactive proof system* Π *admits proofs 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*

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

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

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

Security of Multiparty Signatures. In multiparty signature schemes one often relies on the so called knowledge of secret key (KOSK) assumption within security proofs, where the adversary is required to reveal the secret keys it utilizes to the environment. This is important to prevent rogue-key attacks, i.e., attacks where the adversary constructs public keys based on existing public keys in the system and must not know the secret key corresponding to the resulting public keys.

To prevent such rogue-key attacks, 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 a distinguished symbol $\perp$. For instance, 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 $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{KeyGen}(\textsc{pp})$ and if so replies with pk and $\perp$ otherwise.

To resemble the KOSK assumption in real protocols without revealing the secret key, 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. We require this for all our constructions in this paper. 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].

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.[1]

We focus on signature schemes $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$, where the secret and public key elements live in groups $(\mathbb{H}, +)$ and $(\mathbb{G}, \cdot)$, respectively. We start with the notion of an efficiently computable homomorphism between secret keys and public keys in analogy to [TW14]. Such a functionality has been used recently in [FKM$^+$16] to define the notion of signatures with re-randomizable keys.

Definition 10 (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 $\mu : \mathbb{H} \rightarrow \mathbb{G}$ 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 $\mathsf{pk} = \mu(\mathsf{sk})$.*

[1] We note that the first parts (up to Definition 12) of this section are slightly more general versions of definitions from previous work of us (currently in submission) where the focus is, however, not on key-homomorphisms.

We stress that secret keys and public keys may be vectors containing elements of $\mathbb{H}$ and $\mathbb{G}$ respectively. Then, the operations $+$, $\cdot$ and the map μ are applied component wise. To keep the definitions compact, we however do not make that fact explicit.

In 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 prime order p group $\mathbb{G}$, 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, where we focus on the class of functions Φ^+ representing linear shifts and note that one could easily adapt our definition to other suitable classes Φ of functions instead of linear shifts. We stress that we consider Φ as 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, Φ admits an efficient membership test, is efficiently samplable, and, its functions are efficiently computable. Definition 11 in combination with the adaptability of signatures (Definition 12) or perfect adaption (Definition 13), can be seen as being in the fashion of key-homomorphic encryption schemes [AHI11].

Definition 11 **(Φ^+-Key-Homomorphic Signatures).** *A signature scheme is called Φ^+-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 $\Delta \in \Phi^+$ as input, and outputs a public key* pk' *and a signature σ',*

such that for all $\Delta \in \Phi^+$ and all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, all messages m and all $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$ and $(\mathsf{pk}', \sigma') \leftarrow \mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta)$ it holds that

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

For simplicity we sometimes identify a function $\Delta \in \Phi^+$ with its "shift amount" $\Delta \in \mathbb{H}$. To illustrate this concept, we take a look at Schnorr signatures.

Schnorr Signatures. Let $\mathbb{G}$ be a group of prime order p generated by g and $H : \{0,1\}^* \to \mathbb{Z}_p$ be a hash function. KeyGen chooses $\mathsf{sk} \xleftarrow{R} \mathbb{Z}_p$ and outputs $(\mathsf{sk}, \mathsf{pk}) \leftarrow (\mathsf{sk}, g^{\mathsf{sk}})$; Sign given sk and message m, chooses $r \xleftarrow{R} \mathbb{Z}_p$, computes $R \leftarrow g^r$, $c = H(R, m)$, $y = r + \mathsf{sk} \cdot c \bmod p$ and outputs $\sigma \leftarrow (c, y)$. Finally, Verify given pk, message m and $\sigma = (c, y)$ outputs 1 if $c = H(\mathsf{pk}^{-c} g^y, m)$, and 0 otherwise. Now, let us adapt a given signature σ to a new public key $\mathsf{pk}' = \mathsf{pk} \cdot g^{\Delta}$ corresponding to $\mathsf{sk}' = \mathsf{sk} + \Delta \bmod p$. Therefore, we simply set $\sigma' \leftarrow (c, y')$ with $y' = y + c \cdot \Delta \bmod p$. It is easy to see that Verify on input $(\mathsf{pk}', m, \sigma')$ will always output 1.

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 from [FHS15] for equivalence class signatures [HS14]. We also note that Kiltz et al. [KMP16] have recently used a notion related to Definition 12 (denoted as random self-reducibility) in context of canonical identification schemes.

Definition 12 (Adaptability of Signatures). *A Φ^+-key-homomorphic signature scheme provides adaptability of signatures, if for every $\kappa \in \mathbb{N}$ and every message m, it holds that* $\mathsf{Adapt}(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m), \Delta)$ *and* $(\mathsf{pk} \cdot \mu(\Delta), \mathsf{Sign}(\mathsf{sk} + \Delta, m))$ *as well as* $(\mathsf{sk}, \mathsf{pk})$ *and* $(\mathsf{sk}', \mu(\mathsf{sk}'))$ *are identically distributed, where* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, $\mathsf{sk}' \overset{R}{\leftarrow} \mathbb{H}$, *and* $\Delta \overset{R}{\leftarrow} \Phi^+$.

Coming back to Schnorr signatures, we immediately see that they are adaptable according to Definition 12 and all schemes that satisfy the stronger notion from Definition 13 below also satisfy this notion.

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.

Definition 13 (Perfect Adaption). *A Φ^+-key-homomorphic signature scheme provides perfect adaption, if for every $\kappa \in \mathbb{N}$, every message m, and every signature $\sigma \leftarrow$* $\mathsf{Sign}(\mathsf{sk}, m)$, *it holds that* $(\sigma, \mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta))$ *and* $(\sigma, \mathsf{pk} \cdot \mu(\Delta), \mathsf{Sign}(\mathsf{sk} + \Delta, m))$ *as well as* $(\mathsf{sk}, \mathsf{pk})$ *and* $(\mathsf{sk}', \mu(\mathsf{sk}'))$ *are identically distributed, where* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, $\mathsf{sk}' \overset{R}{\leftarrow} \mathbb{H}$, *and* $\Delta \overset{R}{\leftarrow} \Phi^+$.

One immediately sees that Schnorr signatures do not satisfy Definition 13 as the randomness r remains fixed. However, we note that there are various existing schemes that satisfy Definition 13. For example, BLS signatures [BLS04] or the recent re-randomizable scheme by Pointcheval and Sanders [PS16b] or the well known Waters signatures [Wat05] to name some (cf. Appendix C for a more formal treatment).

When looking at Definition 11, one could ask whether it is possible to replace Δ in the Adpat 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 Φ^+-key-homomorphic signature scheme Σ for which there exists a modified* Adapt' *algorithm taking $\mu(\Delta)$ instead of Δ that still satisfies Definition 11.*

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 $\mathsf{pk}' = \mathsf{pk}^\star \cdot \mathsf{pk}^{-1}$ 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 14 (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 *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 holds that*

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

Analogously to Definitions 12 and 13, one can define indistinguishability of fresh and combined signatures, but we omit it here as it is straight forward. We want to mention that Definition 14 is, for instance, satisfied by BLS signatures, Waters' signatures with shared Waters' hash parameters (cf. [LOS+06]), as well as the scheme with shared parameters assuming synchronized time in [CHP12] being a variant of the CL signature scheme [CL04] (cf. Appendix C for a more formal treatment).

4 Applications

In this section we show how the various key-homomorphic properties defined in the previous section facilitate the black-box construction of ring signatures, universal designated verifier signatures as well as multisignatures.

4.1 Ring Signatures

Ring signature schemes [RST01] are a variant of signature schemes that 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. Ring signatures have proven to be an interesting tool for numerous applications. The two main lines of work in the design of ring signatures target reducing the signature size or removing the requirement for random oracles (e.g., [DKNS04, CGS07, GK15]). We provide a construction that does not require random oracles and has linear signature size. It provides an alternative very simple generic framework to construct ring signatures in addition to existing ones (cf. [BKM09, BK10]).

Subsequently, 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 15. *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 [BKM09] 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 $\mathcal{R}$.

Definition 16 (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*

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

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.

Definition 17 (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*

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

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

Our Construction. In Scheme 1 we present our black-box construction of ring signatures from any Φ^+-key-homomorphic $\mathsf{EUF\text{-}CMA}$ secure signature scheme Σ with adaptable signatures and any witness indistinguishable proof system admitting proofs of knowledge. The idea behind the scheme is as follows. A ring signature for message m consists of a signature for m 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 the ring $\mathcal{R}$.[2] 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 proof 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.

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

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), one could use the following **NP** relation R.

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

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

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

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

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

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

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

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

Scheme 1: Black-Box Construction of Ring Signatures

[2] For technical reasons we need to include an additional public key cpk into $\mathcal{R}$.

Theorem 1. *If Σ is correct,* $\mathsf{EUF\text{-}CMA}$ *secure, and provides adaptability of signatures,* Π *is complete, witness indistinguishable, and admits proofs of knowledge, then Scheme 1 is correct, unforgeable, and anonymous.*

We prove the theorem above in Appendix A.

Removing the CRS. It is important to note that when opting for an instantiation of Scheme 1 in the ROM one can completely avoid the CRS. Firstly, when using Schnorr proofs made non-interactive using the Fiat-Shamir transform [FS86] as proof system Π (cf. [FKMV12]) one does not require crs. Secondly, instead of including $(\mathsf{csk}, \mathsf{cpk})$ in PP one can use a neat trick by Abe and Okamoto [AO00]. In particular, using a random oracle $H : \{0,1\}^* \rightarrow \mathbb{G}$ one can freshly obtain $\mathsf{cpk} \leftarrow H(\mathcal{R})$ upon signature generation and verification; the reduction is still able to simulate signatures by programming the random oracle.

4.2 Universal Designated Verifier Signatures

Designated verifier signatures [JSI96] are an interesting variant of signatures, where the 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 quite a lot of work on UDVS, and, most notably, in [SS08] it was shown how to convert a large class of signature schemes to UDVS. Their approach can thus be seen as related to our approach, yet they do not rely on key-homomorphisms and they only achieve weaker security guarantees.[3]

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.[4]

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.

[3] 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.

[4] We, however, note that an extension of the UDVS model to universal designated verifier *ring* signatures would be straight forward and also our scheme would be straight forwardly extensible using the same techniques as in Scheme 1.

Definition 18. *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.*

$\mathsf{DVGen}(\textsc{pp})$: *This algorithm takes the public parameters* $\textsc{pp}$ *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 19 (Simulation-Sound DV-Unforgeability). *An* UDVS *provides DV-unforgeability, if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\varepsilon(\cdot)$ *such that it holds that*

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

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 20 (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*

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

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 2, we present our construction of UDVS from any Φ^+-key-homomorphic EUF-CMA secure Σ with perfect adaption of signatures, any witness indistinguishable Π which admits proofs of knowledge, and any one way function f.[5] Our construction uses the "OR-trick" [JSI96], which is well known in the context of DVS. Upon computing designations and simulations of designated-verifier signatures, we require to prove knowledge of witnesses for the following **NP** relation R:

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

The nice thing when choosing R this way is that we can simulate proofs while the proof system is set up to provide soundness by either using sk or vsk as a simulation trapdoor.[6] For brevity we assume that the parameters $\textsc{pp}$ 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.

[5] We note that our construction borrows ideas from earlier work of us on a variant of redactable signatures (currently in submission).

[6] Note that this is similar to the generic conversion of witness indistinguishable proof systems to zero-knowledge proof systems [FLS90].

$\mathsf{DVGen}(\textsc{pp})$: Run $\mathsf{vsk} \xleftarrow{R} \mathsf{Dom}(f)$, 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 2: 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 witness indistinguishable and admits proofs of knowledge, then Scheme 2 is correct, simulation-sound DV-unforgeable, and provides strong non-transferability privacy.*

We prove the theorem above in Appendix B and note that if non-transferability privacy is sufficient, Σ only needs to be adaptable. Then, one can for instance also instantiate Scheme 2 with the very efficient Schnorr signature scheme.

4.3 Multisignatures

A multisignature scheme [IN83] is a signature scheme that 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] that are directly based on the BLS [BLS04] and 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. For instance, as we show in Appendix C.4, the modified CL signature scheme from [CHP12] provides this homomorphism, and, therefore, directly yields a new instantiation of multisignatures.

We now give a formal definition of multisignatures, where we follow Ristenpart and Yilek [RY07]. But 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 21. *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 paramter generation algorithm takes a security parameter* κ *and produces global parameters* $\textsc{pp}$ *(including the security parameters and a description of the message space* $\mathcal{M}$*).*

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

Sign : *This is an interactive multisignature algorithm executed by a group of signers who intend to sign the same message* m*. Each signer* S_i *executes* Sign *on public inputs* $\textsc{pp}$*, public key multiset* $\textsc{pk}$*, message* m *and secret input its secret* sk_i *and outputs a multisignature* σ*.*

$\mathsf{Verify}(\textsc{pp}, \textsc{pk}, m, \sigma)$: *This algorithm takes public parameters* $\textsc{pp}$*, a public key multiset* $\textsc{pk}$*, 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, which basically states that $\mathsf{Verify}(\textsc{pp}, \textsc{pk}, m, \mathsf{Sign}(\textsc{pp}, \textsc{pk}, m, \mathsf{sk})) = 1$ for any m honestly generated $\textsc{pp}$ and when every participant correctly follows the algorithms. Besides correctness, we require existential unforgeability under a chosen message attack against a single honest player.

Definition 22 (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*

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

where the environment maintains 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:*

$\mathsf{Sign}(\textsc{pk}, m)$: *This oracle obtains a public key set* $\textsc{pk}$ *and returns* $\perp$ *if* $\mathsf{pk}^\star \notin \textsc{pk}$*. Otherwise it simulates a new instance of* $\mathsf{Sign}(\textsc{pp}, \textsc{pk}, m, \mathsf{sk}^\star)$ *forwarding messages to and from* $\mathcal{A}$ *appropriately and sets* $\mathcal{Q}^{\mathsf{Sign}} \xleftarrow{\cup} m$*.*

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

Our Construction. Subsequently, we restrict ourselves to non-interactive Sign protocols, which basically means that every signer S_i locally computes a signatures σ_i and then broadcasts it to all other signers in $\textsc{pk}$. Furthermore, we

$\mathsf{PGen}(1^\kappa)$: 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:

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

Scheme 3: Black-Box Construction of Multisignatures

consider the signature scheme Σ to work with common parameters PP and in Scheme 3 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.

Theorem 3. *If Σ is correct,* EUF-CMA *secure, and publicly key-homomorphic, then Scheme 3 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 $\mathtt{KEY}$ 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. □

5 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 (levelled) fully homomorphic signatures supporting arbitrary functions [GVW15, BFS14]. However, all existing 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 [LTV12, CM15, MW16, PS16a], are known, such a feature has never been investigated in context of signatures so far.

In this section we close this gap and 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 and moreover discuss the usefulness of an additional homomorphic property on the key space for such schemes. Moreover, we discuss potential applications of such schemes and interesting open questions.

5.1 Multikey-Homomorphic Signatures

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 from Section 3, 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. Although we present a general definition of multikey-homomorphic schemes which, in analogy to the encryption case, i.e., [LTV12, CM15, MW16, PS16a, BP16], support the input of a set of public keys into the verification of a combined signature, we focus on schemes who use a *succinct* representation of a combined public key in the verification below.

Definition 23 (Multikey-Homomorphic Signatures). *A multikey-homomorphic signatures 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*).*

$\mathsf{Sign}(\mathsf{sk}, m, \tau)$: *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 24 (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 $\hat{m} = f(m_1, \ldots, m_n)$.

Definition 25 (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*

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

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

Observe that Definition 23 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 26 (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 2. *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, a major open question is whether it is also possible to come up with schemes which provide signature succinctness as defined below.

Definition 27 (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*

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

Finally, one could also define a notion in the vein of function privacy in the context of functional signatures [BGI14], i.e., although Combine takes a function f, the output of Combine would be required to be indistinguishable for any f' that evaluates to the same output on the same input. Ultimately, one could even ask for a stronger property requiring that the signatures output by Combine look identical to signatures produced by Sign.

5.2 Discussion

We consider it as an interesting open problem to find constructions of the various flavors of multikey-homomorphic signatures discussed above. It seems that using indistinguishability obfuscation in similar fashion as it is done in the context of universal signature aggregators [HKW15] is a viable direction to obtain signature succinctness. However, as the focus in this paper lies on key-homomorphisms, we leave a thorough investigation as future work. Another interesting question in this direction is whether one can achieve key and signature succinctness at the same time.

Subsequently, we informally discuss some further observations.

Related Concepts. Firstly, it seems that our notions are related to the properties one would expect from aggregate signatures [BGLS03] and the related notion of screening [BGR98]. Furthermore, they also seem to be related to batch verification of signatures [CHP12] and the recent notion of universal signature aggregators [HKW15].

Application to Delegation of Computation. Secondly, the concept of multi-key-homomorphic signatures seem to be a very interesting concept in the domain of verifiable delegation of computation on outsourced data.

Let us recall that homomorphic signatures for a class $\mathcal{F}$ can be used to certify computations on signed data for any $f \in \mathcal{F}$. Assume that some entity who holds a data set $(m_1, \ldots, m_n)$, is in possession of a secret key sk and produces signatures $(\sigma_1, \ldots, \sigma_n)$ for each respective message in the data set. Then, she can outsource the authenticated data set $(m_1, \sigma_1), \ldots, (m_n, \sigma_n)$ to some remote server (e.g., the cloud). Later, for any function $f \in \mathcal{F}$, the server can be asked to compute $\hat{m} = f(m_1, \ldots, m_n)$ and is able to deliver a succinct proof (signature) $\hat{\sigma}$ certifying the correctness of the computation. Anyone, given the public key pk of the data holder, the result $\hat{m}$, corresponding signature $\hat{\sigma}$ and the function f, can then verify whether the computation by the server has been performed correctly without needing to know the original data.

Now, there are many scenarios with many different signers each of them holding a distinct secret key sk_i and each of them periodically authenticates some data item $m_{i,j}$ and sends it to a server. Then, the server could compute a function f over inputs authenticated by different secret keys. Think for instance of environmental sensors that periodically send authenticated measurements to a server and this server can then compute on these authenticated measurements. The result can then be verified under the respective public keys or in case of a scheme with key succinctness the results are verifiable for anyone under a compact public key $\hat{\mathsf{pk}}$ (which can be computed from all the single public keys once and pre-distributed). Consequently, the concept of multikey-homomorphic signatures seems to be an interesting and viable direction for extending the scope of verifiable delegation of computation on outsourced data based on signatures.

Acknowledgements. We thank the anonymous referees from TCC 2016-B 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 *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *LNCS*, pages 1–20. Springer, 2012.

[AHI11] Benny Applebaum, Danny Harnik, and Yuval Ishai. Semantic Security under Related-Key Attacks and Applications. In *Innovations in Computer Science - ICS 2011*, pages 45–60. Tsinghua University Press, 2011.

[ALP12] Nuttapong Attrapadung, Benoît Libert, and Thomas Peters. Computing on Authenticated Data: New Privacy Definitions and Constructions. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *LNCS*, pages 367–385. Springer, 2012.

[AO00] Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, pages 271–286, 2000.

[BCM11] Mihir Bellare, David Cash, and Rachel Miller. Cryptography Secure against Related-Key Attacks and Tampering. In *Advances in Cryptology - ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 486–503. Springer, 2011.

[BF11] Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. In *Advances in Cryptology–EUROCRYPT 2011*, pages 149–168. Springer, 2011.

[BFKW09] Dan Boneh, David Mandell Freeman, Jonathan Katz, and Brent Waters. Signing a Linear Subspace: Signature Schemes for Network Coding. In *Public Key Cryptography*, volume 5443 of *LNCS*, pages 68–87. Springer, 2009.

[BFP$^+$15] Abhishek Banerjee, Georg Fuchsbauer, Chris Peikert, Krzysztof Pietrzak, and Sophie Stevens. Key-Homomorphic Constrained Pseudorandom Functions. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015*, volume 9015 of *LNCS*, pages 31–60. Springer, 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 *Advances in Cryptology - EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, 2014.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings*, volume 8383 of *LNCS*, pages 501–519. Springer, 2014.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *LNCS*, pages 416–432. Springer, 2003.

[BGR98] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *LNCS*, pages 236–250. Springer, 1998.

[BJ08] Ali Bagherzandi and Stanislaw Jarecki. Multisignatures using proofs of secret key possession, as secure as the diffie-hellman problem. In *Security and Cryptography for Networks, 6th International Conference, SCN 2008,* *Amalfi, Italy, September 10-12, 2008. Proceedings*, volume 5229 of *LNCS*, pages 218–235. Springer, 2008.

[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*, 2010:86, 2010.

[BKM09] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology*, 22(1):114–138, 2009.

[BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key Homomorphic PRFs and Their Applications. In *Advances in Cryptology - CRYPTO 2013*, volume 8042 of *LNCS*, pages 410–428. Springer, 2013.

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In *Public Key Cryptography - PKC 2003, 6th International Workshop on Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6-8, 2003, Proceedings*, volume 2567 of *LNCS*, pages 31–46. Springer, 2003.

[BP14] Abhishek Banerjee and Chris Peikert. New and Improved Key-Homomorphic Pseudorandom Functions. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference*, volume 8616 of *LNCS*, pages 353–370. Springer, 2014.

[BP16] Zvika Brakerski and Renen Perlman. Lattice-based fully dynamic multi-key FHE with short ciphertexts. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 190–213, 2016.

[BPT12] Mihir Bellare, Kenneth G. Paterson, and Susan Thomson. RKA security beyond the linear barrier: Ibe, encryption and signatures. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *LNCS*, pages 331–348. Springer, 2012.

[Cat14] Dario Catalano. Homomorphic signatures and message authentication codes. In *Security and Cryptography for Networks - 9th International Conference, SCN 2014, Amalfi, Italy, September 3-5, 2014. Proceedings*, volume 8642 of *LNCS*, pages 514–519. Springer, 2014.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *LNCS*, pages 174–187. Springer, 1994.

[CFW14] Dario Catalano, Dario Fiore, and Bogdan Warinschi. Homomorphic signatures with efficient verification for polynomial functions. In *Advances in Cryptology–CRYPTO 2014*, pages 371–389. Springer, 2014.

[CGS07] Nishanth Chandran, Jens Groth, and Amit Sahai. Ring signatures of sublinear size without random oracles. In *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings*, pages 423–434, 2007.

[CHKM10] Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3):141–167, 2010.

[CHP12] Jan Camenisch, Susan Hohenberger, and Michael Østergaard Pedersen. Batch verification of short signatures. *J. Cryptology*, 25(4):723–747, 2012.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference*, volume 3152 of *LNCS*, pages 56–72. Springer, 2004.

[CM15] Michael Clear and Ciaran McGoldrick. Multi-identity and multi-key leveled FHE from learning with errors. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 630–656, 2015.

[DKNS04] Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup. Anonymous identification in ad hoc groups. In *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, pages 609–626, 2004.

[DMS16] Yevgeniy Dodis, Ilya Mironov, and Noah Stephens-Davidowitz. Message transmission with reverse firewalls - secure communication on corrupted machines. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 341–372, 2016.

[FF13] Marc Fischlin and Nils Fleischhacker. Limitations of the meta-reduction technique: The case of schnorr signatures. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques*, volume 7881 of *LNCS*, pages 444–460. Springer, 2013.

[FHS15] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *LNCS*, pages 233–253. Springer, 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 *Public-Key Cryptography - PKC 2016 - 19th IACR International Conference on Practice and Theory in Public-Key Cryptography*, volume 9614 of *LNCS*, pages 301–330. Springer, 2016.

[FKMV12] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the fiat-shamir transform. In *Progress in Cryptology - INDOCRYPT 2012, 13th International Conference on Cryptology in India, Kolkata, India, December 9-12, 2012. Proceedings*, pages 60–79, 2012.

[FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 308–317, 1990.

[Fre12] David Mandell Freeman. Improved security for linearly homomorphic signatures: A generic framework. In *Public Key Cryptography - PKC 2012 - 15th*

International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings, pages 697–714, 2012.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 186–194, 1986.

[Gen09] Craig Gentry. Fully Homomorphic Encryption using Ideal Lattices. In *41st Annual ACM Symposium on Theory of Computing, STOC 2009*, pages 169–178. ACM, 2009.

[GK15] Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 253–280, 2015.

[GLW12] Shafi Goldwasser, Allison B. Lewko, and David A. Wilson. Bounded-Collusion IBE from Key Homomorphism. In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012*, volume 7194 of *LNCS*, pages 564–581. Springer, 2012.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 469–477. ACM, 2015.

[HKW15] Susan Hohenberger, Venkata Koppula, and Brent Waters. Universal signature aggregators. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 3–34, 2015.

[HS14] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *LNCS*, pages 491–511. Springer, 2014.

[IN83] K. Itakura and K. Nakamura. A public-key cryptosystem suitable for digital multisignatures. *NEC Research & Development*, 71:1–8, 1983.

[JMSW02] Robert Johnson, David Molnar, Dawn Xiaodong Song, and David Wagner. Homomorphic signature schemes. In *Topics in Cryptology - CT-RSA 2002, The Cryptographer's Track at the RSA Conference, 2002, San Jose, CA, USA, February 18-22, 2002, Proceedings*, volume 2271 of *LNCS*, pages 244–262. Springer, 2002.

[JSI96] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 143–154, 1996.

[KMP16] Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, pages 33–61, 2016.

[LOS+06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *LNCS*, pages 465–485. Springer, 2006.

[LTV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012*, pages 1219–1234. ACM, 2012.

[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 *Information Security and Cryptology - ICISC 2015 - 18th International Conference, Seoul, South Korea, November 25-27, 2015, Revised Selected Papers*, volume 9558 of *LNCS*, pages 20–35. Springer, 2015.

[MW16] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 735–763, 2016.

[PS16a] Chris Peikert and Sina Shiehian. Multi-key FHE from lwe, revisited. *IACR Cryptology ePrint Archive*, 2016:196, 2016.

[PS16b] David Pointcheval and Olivier Sanders. Short Randomizable Signatures. In *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016*, volume 9610 of *LNCS*, pages 111–126. Springer, 2016.

[Rot11] Ron Rothblum. Homomorphic Encryption: From Private-Key to Public-Key. In *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011*, volume 6597 of *LNCS*, pages 219–234. Springer, 2011.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *LNCS*, pages 552–565. Springer, 2001.

[RY07] Thomas Ristenpart and Scott Yilek. The Power of Proofs-of-Possession: Securing Multiparty Signatures against Rogue-Key Attacks. In *Advances in Cryptology - EUROCRYPT 2007*, volume 4515 of *LNCS*, pages 228–245. Springer, 2007.

[SBWP03] Ron Steinfeld, Laurence Bull, Huaxiong Wang, and Josef Pieprzyk. Universal designated-verifier signatures. In *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings*, pages 523–542, 2003.

[SS08] Siamak Fayyaz Shahandashti and Reihaneh Safavi-Naini. Construction of universal designated-verifier signatures and identity-based signatures from standard signatures. In *Public Key Cryptography - PKC 2008, 11th International Workshop on Practice and Theory in Public-Key Cryptography, Barcelona, Spain, March 9-12, 2008. Proceedings*, 2008.

[TW14] Stefano Tessaro and David A. Wilson. Bounded-collusion identity-based encryption from semantically-secure public-key encryption: Generic constructions with short ciphertexts. In *Public-Key Cryptography - PKC 2014*, volume 8383 of *LNCS*, pages 257–274. Springer, 2014.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *LNCS*, pages 114–127. Springer, 2005.

A Proof of Theorem 1

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

Lemma 3. *If Σ is correct, and Π is complete, then Scheme 1 is correct.*

Lemma 3 follows from inspection and the proof is therefore omitted.

Lemma 4. *If Σ is* $\mathsf{EUF\text{-}CMA}$ *secure, and provides adaptability of signatures, and Π admits proofs of knowledge, then Scheme 1 is unforgeable.*

Proof. In front of an adversary, we randomly guess it's strategy and either follow (1) or (2).

(1) We show that a Type-(1) adversary has negligible success probability.

Game 0: The original unforgeability 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. Furthermore, we also store csk.

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

Game 2: As Game 1, but inside the Sig oracle we execute the following modified Sign algorithm Sign' which additionally takes csk as input.

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

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

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

Game 3: As Game 2, but upon Setup we generate $\boxed{(\mathsf{crs}, \tau) \leftarrow \Pi.\mathsf{E}_1(1^\kappa)}$ and store the trapdoor τ.

Transition - Game 2 → Game 3: A distinguisher $\mathcal{D}^{2\to 3}$ distinguishes an honest CRS from an extraction CRS, i.e., $|\Pr[S_3] - \Pr[S_2]| \le \varepsilon_{\mathsf{e1}}(\kappa)$.

Game 4: As Game 3, but for every forgery $(m^\star, \sigma^\star, \mathcal{R}^\star)$ output by the adversary, we parse $\sigma^\star$ as $(\delta^\star, \mathsf{pk}^\star, \pi^\star)$, extract the witness $\mathsf{sk}' \leftarrow \Pi.\mathsf{E}_2(\mathsf{crs}, \tau, (\mathsf{pk}^\star, \mathcal{R}^\star), \pi^\star)$. If the extraction fails we abort.

Transition - Game 3 → Game 4: The success probability in Game 2 is the same as in Game 1, unless the extraction fails. That is, $\mathsf{Pr}[S_4] = (1-\varepsilon_{\mathsf{e2}}(\kappa))\cdot\mathsf{Pr}[S_3]$.

Game 5: As Game 4, but we abort if we have extracted sk' such that $\mathsf{cpk} = \mathsf{pk}^\star \cdot \mu(\mathsf{sk}')$.

Transition - Game 4 → Game 5: If we abort our guess regarding the adversarial strategy was wrong, i.e., $\mathsf{Pr}[S_5] = \frac{1}{2} \cdot \mathsf{Pr}[S_4]$.

Game 6: As Game 5, but we guess the index i the adversary will attack at the beginning of the game, and abort if our guess is wrong, i.e., $\mathsf{pk}_i \neq \mathsf{pk}^\star \cdot \mu(\mathsf{sk}')$.

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, we engage with an $\mathsf{EUF\text{-}CMA}$ challenger of Σ to obtain pk_i.

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 5 we compute $(\mathsf{pk}_i, \sigma_i) \leftarrow \mathsf{Adapt}(\mathsf{pk}^\star, m^\star, \sigma^\star, \mathsf{sk}')$ and return $(\sigma_i, m^\star)$ as a valid forgery for Σ. That is, $\mathsf{Pr}[S_7] \leq \varepsilon_{\mathsf{f}}(\kappa)$ and we obtain the following bound for the success probability of a Type-(1) adversary, i.e., $\mathsf{Pr}[S_0] \leq \frac{2\cdot\mathsf{poly}(\kappa)\cdot\varepsilon_{\mathsf{f}}(\kappa)}{1-\varepsilon_{\mathsf{e2}}(\kappa)} + \varepsilon_{\mathsf{e1}}(\kappa) + \varepsilon_{\mathsf{wi}}(\kappa)$ which is negligible.

(2) We show that a Type-(2) adversary has negligible success probability.

Game 0: The original unforgeability game.

Game 1: As Game 0, but upon Setup we generate $\boxed{(\mathsf{crs}, \tau) \leftarrow \Pi.\mathsf{E}_1(1^\kappa)}$ and store the trapdoor τ.

Transition - Game 0 → Game 1: A distinguisher $\mathcal{D}^{0\to 1}$ distinguishes an honest CRS from an extraction CRS, i.e., $|\mathsf{Pr}[S_1] - \mathsf{Pr}[S_0]| \leq \varepsilon_{\mathsf{e1}}(\kappa)$.

Game 2: As Game 1, but for every forgery $(m^\star, \sigma^\star, \mathcal{R}^\star)$ output by the adversary, we parse $\sigma^\star$ as $(\delta^\star, \mathsf{pk}^\star, \pi^\star)$, extract the witness $\mathsf{sk}' \leftarrow \Pi.\mathsf{E}_2(\mathsf{crs}, \tau, (\mathsf{pk}^\star, \mathcal{R}^\star), \pi^\star)$. If the extraction fails we abort.

Transition - Game 1 → Game 2: The success probability in Game 1 is the same as in Game 2, unless the extraction fails. That is, $\mathsf{Pr}[S_2] = (1-\varepsilon_{\mathsf{e2}}(\kappa))\cdot\mathsf{Pr}[S_1]$.

Game 3: As Game 2, but we abort if we have extracted sk' such that $\mathsf{cpk} \neq \mathsf{pk}^\star \cdot \mu(\mathsf{sk}')$.

Transition - Game 2 → Game 3: If we abort our guess regarding the adversarial strategy was wrong, i.e., $\mathsf{Pr}[S_3] = \frac{1}{2} \cdot \mathsf{Pr}[S_2]$.

Game 4: As Game 3, but instead of honestly generating $(\mathsf{csk}, \mathsf{cpk})$ upon Setup we engage with an $\mathsf{EUF\text{-}CMA}$ challenger of Σ to obtain cpk and set $\mathsf{csk} \leftarrow \perp$.

Transition - Game 3 → Game 4: This change is conceptual, i.e., $\mathsf{Pr}[S_3] = \mathsf{Pr}[S_4]$.

If the adversary outputs a forgery $(m^\star, \sigma^\star, \mathcal{R}^\star)$ in Game 3 we compute $(\mathsf{cpk}, \sigma) \leftarrow \mathsf{Adapt}(\mathsf{pk}^\star, m^\star, \sigma^\star, \mathsf{sk}')$ and return $(\sigma, m^\star)$ as a valid forgery for Σ. Thus, we have that $\mathsf{Pr}[S_4] \leq \varepsilon_{\mathsf{f}}(\kappa)$ and we obtain the following bound for the success probability of a Type-(1) adversary, i.e., $\mathsf{Pr}[S_0] \leq \frac{2\cdot\varepsilon_{\mathsf{f}}(\kappa)}{1-\varepsilon_{\mathsf{e2}}(\kappa)} + \varepsilon_{\mathsf{e1}}(\kappa)$ which is negligible.

Overall Bound. The overall success probability is bounded by the maximum success probabilities in (1) and (2), which proves the lemma. □

Lemma 5. *If Σ provides adaptability of signatures and Π is witness indistinguishable, then Scheme 1 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 bot if $\mu(\mathsf{sk}_0) \notin \mathcal{R}$ $\boxed{\vee\ \mu(\mathsf{sk}_1) \notin \mathcal{R}}$. Otherwise, return $\sigma \leftarrow (\sigma, \mathsf{pk}, \pi)$, where

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

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

In Game 2, we have a simulation for $b = 1$; $|\mathsf{Pr}[S_2] - \mathsf{Pr}[S_0]| \leq \varepsilon_{\mathsf{wi}}(\kappa)$, which proves the lemma. □

B Proof of Theorem 2

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

Lemma 6. *If Σ is* EUF-CMA *secure and perfectly adapts signatures, f is a one-way function, and Π is witness indistinguishable and admits proofs of knowledge, then Scheme 2 is simulation-sound DV-unforgeable.*

Proof. We show that an adversary against DV-unforgeability is either (1) an EUF-CMA adversary for Σ, or (2) an adversary against the one-wayness of f. In front of an adversary we randomly guess it's strategy uniformly at random; taking both cases together then proves the lemma.

(1) We followingly bound the success probability for an EUF-CMA forger, 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 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 0 → Game 1: This change is conceptual, i.e., $\mathsf{Pr}[S_0] = \mathsf{Pr}[S_1]$.

Game 2: As Game 1, but inside the Sim oracle we execute the following modified Sim algorithm Sim', where $\mathcal{C}^{\mathsf{f}}$ denotes an $\mathsf{EUF\text{-}CMA}$ challenger.

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

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

Further, the environment keeps a mapping from public keys pk_{R} to challengers $\mathcal{C}^{\mathsf{f}}_{\kappa}$.

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 obtain crs upon Setup using $\boxed{(\mathsf{crs}, \tau) \leftarrow \Pi.\mathsf{E}_1(1^{\kappa})}$ and store the trapdoor τ.

Transition - Game 2 → Game 3: A distinguisher $\mathcal{D}^{2\rightarrow 3}$ distinguishes an honest CRS from an extraction CRS, i.e., $|\mathsf{Pr}[S_2] - \mathsf{Pr}[S_3]| \leq \varepsilon_{\mathsf{e1}}(\kappa)$.

Game 4: As Game 3, but for every forgery $(m^\star, \delta^\star)$ output by the adversary, we parse $\delta^\star$ as $(\mathsf{pk}'^\star, \sigma^\star_{\mathsf{R}}, \pi^\star)$ and extract the witness $(\mathsf{sk}', \mathsf{vsk}) \leftarrow \Pi.\mathsf{E}_2(\mathsf{crs}, \tau, (\mathsf{pk}'^\star, \mathsf{vpk}), \pi^\star)$.

Transition - Game 3 → Game 4: The success probability in Game 3 is the same as in Game 4, unless the extraction fails. That is, $\mathsf{Pr}[S_4] = (1 - \varepsilon_{\mathsf{e2}}(\kappa)) \cdot \mathsf{Pr}[S_3]$.

Game 5: As in Game 4, but whenever the adversary outputs a valid forgery, we check whether $\mathsf{pk} \cdot \mathsf{pk}'$ corresponds to a pk_{R} obtained from a challenger in the Sim oracle, or whether we have extracted sk' such that $\mu(\mathsf{sk}') = \mathsf{pk}'$. If not, we abort as we are in case (2).

Transition - Game 4 → Game 5: If we abort our guess regarding the adversarial strategy was wrong, i.e., $\mathsf{Pr}[S_5] = \frac{1}{2} \cdot \mathsf{Pr}[S_4]$.

In Game 5, we can directly output $(m^\star, \sigma^\star_{\mathsf{R}})$ as a forgery for Σ if $\mathsf{pk} \cdot \mathsf{pk}'$ corresponds to a pk_{R} obtained from a challengers within Sim, or, if $\mu(\mathsf{sk}') = \mathsf{pk}'$, we can obtain $(\mathsf{pk}, \sigma) \leftarrow \Sigma.\mathsf{Adapt}(\mathsf{pk} \cdot \mathsf{pk}', m^\star, \sigma^\star_{\mathsf{R}}, -\mathsf{sk}')$ and output $(m^\star, \sigma)$ as a forgery for Σ. Taking the union bound yields $\mathsf{Pr}[S_5] \leq (q_{\mathsf{Sim}} + 1) \cdot \varepsilon_{\mathsf{f}}(\kappa)$, and we obtain $\mathsf{Pr}[S_0] \leq \frac{2 \cdot (q_{\mathsf{Sim}}+1) \cdot \varepsilon_{\mathsf{f}}(\kappa)}{1 - \varepsilon_{\mathsf{e2}}(\kappa)} + \varepsilon_{\mathsf{e1}}(\kappa)$ which is negligible.

(2) Subsequently we bound the success probability for a one-wayness adversary.

Game 0: The original DV-unforgeability game.

Game 1: As Game 0, but we simulate the Vrfy oracle by using the following modified $\mathsf{DVerify}$ algorithm $\mathsf{DVerify}'$ which takes vpk instead of vsk as input.

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

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

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 generating crs upon setup, we obtain crs from a witness indistinguishability challenger $\mathcal{C}_\kappa^{\mathsf{wi}}$ upon Setup.

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 inside the Sim oracle we execute the following modified Sim algorithm Sim' which additionally takes sk and vpk as input.

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

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

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

Game 4: As Game 3, but we further modify Sim', which now runs without vsk, as follows.

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

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

Transition - Game 3 → Game 4: A distinguisher between $\mathcal{D}^{3\to4}$ is a distinguisher for adaptive witness indistinguishability of Π, i.e., $|\mathsf{Pr}[S_4] - \mathsf{Pr}[S_3]| \leq \varepsilon_{\mathsf{wi}}(\kappa)$.

Game 5: As Game 4, but instead of generating $(\mathsf{vsk}, \mathsf{vpk}) \leftarrow \mathsf{DVGen}(\textsc{pp})$, we obtain vpk from an $\mathsf{EUF\text{-}CMA}$ challenger for Σ and set $\mathsf{vsk} \leftarrow \bot$.

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 crs upon Setup using $\boxed{(\mathsf{crs}, \tau) \leftarrow \Pi.\mathsf{E}_1(1^\kappa)}$ and store the trapdoor τ.

Transition - Game 5 → Game 6: A distinguisher $\mathcal{D}^{5\to6}$ distinguishes an honest CRS from an extraction CRS, i.e., $|\mathsf{Pr}[S_6] - \mathsf{Pr}[S_5]| \leq \varepsilon_{\mathsf{e1}}(\kappa)$.

Game 7: As Game 6, but for every forgery $(m^\star, \delta^\star)$ output by the adversary, we parse $\delta^\star$ as $(\mathsf{pk}'^\star, \sigma_{\mathsf{R}}^\star, \pi^\star)$, extract the witness $(\mathsf{sk}', \mathsf{vsk}) \leftarrow \Pi.\mathsf{E}_2(\mathsf{crs}, \tau, (\mathsf{pk}'^\star, \mathsf{vpk}), \pi^\star)$.

Transition - Game 6 → Game 7: The success probability in Game 6 is the same as in Game 7, unless the extraction fails. That is, $\mathsf{Pr}[S_7] = (1-\varepsilon_{\mathsf{e2}}(\kappa))\cdot\mathsf{Pr}[S_6]$.

Game 8: As Game 7, but whenever the adversary outputs a valid forgery, we check whether we have extracted vsk such that $f(\mathsf{vsk}) \neq \mathsf{vpk}$ and abort if so (as we are in the other case).

Transition - Game 7 → Game 8: If we abort our guess regarding the adversarial strategy was wrong, i.e., $\mathsf{Pr}[S_8] = \frac{1}{2} \cdot \mathsf{Pr}[S_7]$.

In Game 8, we output vsk and break the one-wayness of the one-way function. Thus, $\mathsf{Pr}[S_8] \leq \varepsilon_{\mathsf{ow}}(\kappa)$ and we obtain $\mathsf{Pr}[S_0] \leq \frac{2\cdot\varepsilon_{\mathsf{ow}}(\kappa)}{1-\varepsilon_{\mathsf{e2}}(\kappa)} + \varepsilon_{\mathsf{e1}}(\kappa) + \varepsilon_{\mathsf{wi}}(\kappa)$.

Overall Bound. The overall success probability is bounded by the maximum success probabilities in (1) and (2), which proves the lemma. □

Lemma 7. *If* Σ *perfectly adapts signatures, and* Π *is witness indistinguishable, then Scheme 2 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:

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

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

Transition - Game 1 → Game 2: A distinguisher between $\mathcal{D}^{1\rightarrow 2}$ 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:

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

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

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; SoD is simulated independently of b and $|\mathsf{Pr}[S_3] - \mathsf{Pr}[S_0]| \leq \varepsilon_{\mathsf{wi}}(\kappa)$, which proves the lemma. □

C Examples of Key-Homomorphic Signature Schemes

Subsequently we give some examples of signature schemes providing key-homomorphic properties. 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 $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, and ψ is an isomorphism $\mathbb{G}_2 \rightarrow \mathbb{G}_1$. If $\mathsf{t} = 3$ the isomorphism ψ is missing. If $\mathsf{t} = 1$ we have that $\mathbb{G}_1 = \mathbb{G}_2$ denoted as $\mathbb{G}$.

C.1 BLS Signatures [BLS04]

In Scheme 4 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 $H : \mathcal{M} \rightarrow \mathbb{G}_1$ uniformly at random from hash function family $\{H_k\}_k$, set $\textsc{pp} \leftarrow (\mathsf{BG}, H)$.
$\mathsf{KeyGen}(\textsc{pp}):$ Choose $x \xleftarrow{R} \mathbb{Z}_p$, set $\mathsf{pk} \leftarrow (\textsc{pp}, \tilde{g}^x)$, $\mathsf{sk} \leftarrow (\mathsf{pk}, x)$, and return $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{Sign}(\mathsf{sk}, m):$ Return $\sigma \leftarrow H(m)^x$.
$\mathsf{Verify}(\mathsf{pk}, m, \sigma):$ Verify whether $e(H(m), \tilde{g}^x) = e(\sigma, \tilde{g})$ and return 1 if so and 0 otherwise.

Scheme 4: Type 3 BLS Signatures

Lemma 8. *BLS signatures are perfectly adaptable according to Definition 13.*

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 $\mathsf{pk} = (\textsc{pp}, \tilde{g}^x)$. Return (pk', σ'), where $\mathsf{pk}' \leftarrow (\textsc{pp}, \tilde{g}^x \cdot \tilde{g}^\Delta)$ and $\sigma' \leftarrow \sigma \cdot H(m)^\Delta$.

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

Lemma 9. *BLS signatures are publicly key-homomorphic according to Definition 14.*

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

C.2 Waters Signatures [Wat05]

Below we recall Waters signatures with shared hashing parameters as presented in [CHKM10]. We stress that while perfect adaption equally applies to the original scheme in [Wat05], the public key-homomorphic property requires different public keys to share the same Water's hash parameters. Consequently, we only present the variant from [CHKM10], which is reasonable in a multi-user setting.

Lemma 10. *Waters signatures are perfectly adaptable according to Definition 13.*

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

$\mathsf{PGen}(1^\kappa):$ Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 3)$, choose $U = (u_0, \dots u_n) \xleftarrow{R} \mathbb{G}_1^k$, and define $H : \mathcal{M} \to \mathbb{G}_1$ as $H(m) \coloneqq u_0 \cdot \prod_{i=1}^n u_i^{m_i}$, where $\mathcal{M} = \{0,1\}^n$. Set $\textsf{PP} \leftarrow (\mathsf{BG}, U, H)$.
$\mathsf{KeyGen}(\textsf{PP}):$ Choose $x \xleftarrow{R} \mathbb{Z}_p$, set $\mathsf{pk} \leftarrow (\textsf{PP}, e(g^x, \tilde{g}))$, $\mathsf{sk} \leftarrow (\mathsf{pk}, g^x)$, and return $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{Sign}(\mathsf{sk}, m):$ Choose $r \xleftarrow{R} \mathbb{Z}_p$, set $\alpha \leftarrow g^x \cdot H(m)^r$, $\beta \leftarrow \tilde{g}^r, \gamma \leftarrow g^r$ and return (α, β, γ).
$\mathsf{Verify}(\mathsf{pk}, m, \sigma):$ Verify whether $e(\alpha, \tilde{g}) = e(g^x, \tilde{g}) \cdot e(H(m), \beta) \wedge e(\gamma, \tilde{g}) = e(g, \beta)$ and return 1 if it holds and 0 otherwise.

Scheme 5: Waters Signatures with Shared Hash Parameters

$\mathsf{Adapt}(\mathsf{pk}, m, \sigma, \Delta):$ Let $\Delta \in \mathbb{G}_1$, $\sigma = (\alpha, \beta, \gamma)$, and $\mathsf{pk} = (\textsf{PP}, e(g^x, \tilde{g}))$. Choose $r' \xleftarrow{R} \mathbb{Z}_p$, compute $\sigma' \leftarrow (\alpha \cdot \Delta \cdot H(m)^{r'}, \beta \cdot \tilde{g}^{r'}, \gamma \cdot g^{r'})$ and $\mathsf{pk}' \leftarrow (\textsf{PP}, e(g^x, \tilde{g}) \cdot e(\Delta, \tilde{g}))$.

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

Lemma 11. *Waters signatures are publicly key-homomorphic according to Definition 14.*

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 $\mathsf{pk}_i = (\textsf{PP}, e(g^{x_i}, \tilde{g}))$. Run $\hat{\mathsf{pk}} \leftarrow (\textsf{PP}, \prod_{i=1}^n e(g^{x_i}, \tilde{g}))$ and $\hat{\sigma} \leftarrow (\prod_{i=1}^n \alpha_i, \prod_{i=1}^n \beta_i, \prod_{i=1}^n \gamma_i)$ and return $\hat{\mathsf{pk}}$ and $\hat{\sigma}$. □

C.3 PS Signatures [PS16b]

In Scheme 6 we recall a recent signature scheme from [PS16b], which provides perfect adaption, but is not publicly key-homomorphic.

$\mathsf{PGen}(1^\kappa):$ Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 3)$ set $\textsf{PP} \leftarrow \mathsf{BG}$.
$\mathsf{KeyGen}(\textsf{PP}):$ Choose $x, y \xleftarrow{R} \mathbb{Z}_p$, compute $\tilde{X} \leftarrow \tilde{g}^x$, $\tilde{Y} \leftarrow \tilde{g}^y$ and set $\mathsf{pk} \leftarrow (\textsf{PP}, \tilde{X}, \tilde{Y})$, $\mathsf{sk} \leftarrow (\mathsf{pk}, x, y)$, and return $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{Sign}(\mathsf{sk}, m):$ Choose $h \xleftarrow{R} \mathbb{G}_1^*$ and return $\sigma \leftarrow (h, h^{(x+y\cdot m)})$.
$\mathsf{Verify}(\mathsf{pk}, m, \sigma):$ Parse σ as (σ_1, σ_2) and 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 12. *PS signatures are perfectly adaptable according to Definition 13.*

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 $(\textsf{PP}, \tilde{X}, \tilde{Y})$, 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 $\mathsf{pk}' \leftarrow (\textsf{PP}, \tilde{X} \cdot \tilde{g}^{\Delta_1}, \tilde{Y} \cdot \tilde{g}^{\Delta_2})$ 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'. $\square$

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.

C.4 CL Signature Variant [CHP12]

While the original pairing-based CL signature scheme [CL04] does not satisfy any of the key-homomorphic properties discussed in this paper, we recall a CL signature variant from [CHP12] in Scheme 7 which does.

$\mathsf{PGen}(1^\kappa)$: Run $\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa, 1)$, choose some polynomially bound set Ψ and hash functions $H_1 : \Psi \to \mathbb{G}$, $H_2 : \Psi \to \mathbb{G}$, $H_3 : \mathcal{M} \times \Psi \to \mathbb{Z}_p$ uniformly at random from suitable hash function families. Set $\mathsf{PP} \leftarrow (\mathsf{BG}, H_1, H_2, H_3)$.
$\mathsf{KeyGen}(\mathsf{PP})$: Choose $x \xleftarrow{R} \mathbb{Z}_p$ and set $\mathsf{pk} \leftarrow (\mathsf{PP}, g^x)$, $\mathsf{sk} \leftarrow (\mathsf{pk}, x)$, and return $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{Sign}(\mathsf{sk}, (m, \psi))$: If it is the first call to Sign during time period $\psi \in \Psi$, then compute $w \leftarrow H_3(m, \psi)$, $a \leftarrow H_1(\psi)$, $b \leftarrow H_2(\psi)$ and return $\sigma \leftarrow a^x b^{xw}$. Otherwise abort.
$\mathsf{Verify}(\mathsf{pk}, (m, \psi), \sigma)$: Compute $w \leftarrow H_3(m, \psi)$, $a \leftarrow H_1(\psi)$, $b \leftarrow H_2(\psi)$ and check whether $e(\sigma, g) = e(a, X) \cdot e(b, X)^w$ holds. If so return 1 and 0 otherwise.

Scheme 7: CL Signature Variant

Lemma 13. *Adapted CL signatures are perfectly adaptable according to Definition 13.*

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

$\mathsf{Adapt}(\mathsf{pk}, (m, \psi), \sigma, \Delta)$: Parse pk as (PP, X) and compute $w \leftarrow H_3(m, \psi)$, $a \leftarrow H_1(\psi)$, $b \leftarrow H_2(\psi)$. Compute $\mathsf{pk}' \leftarrow (\mathsf{PP}, X \cdot g^\Delta)$ and $\sigma' \leftarrow \sigma \cdot a^\Delta \cdot b^{\Delta \cdot w}$ and return (pk', σ').

It is easy to see that adapted signatures are identical to fresh signatures under $\mathsf{pk}' = (\mathsf{PP}, X \cdot g^\Delta)$. $\square$

Lemma 14. *Adapted CL signatures are publicly key-homomorphic according to Definition 14.*

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 $\mathsf{pk}_i = (\mathsf{PP}, g^{x_i})$. Run $\hat{\mathsf{pk}} \leftarrow (\mathsf{PP}, \prod_{i=1}^n g^{x_i}$ and $\hat{\sigma} \leftarrow (\prod_{i=1}^n \sigma_i)$ and return $\hat{\mathsf{pk}}$ and $\hat{\sigma}$. $\square$