A Zoo of Homomorphic Signatures
Multi-Key and Key-Homomorphism

Russell W. F. Lai, Raymond K. H. Tai, Harry W. H. Wong, Sherman S. M. Chow

Department of Information Engineering
The Chinese University of Hong Kong
Sha Tin, N.T., Hong Kong
{russell,raymondtai,whwong,sherman}@ie.cuhk.edu.hk

Abstract. Homomorphic signatures (HS) allow evaluation of signed messages by producing a signature on a function of messages signed by the same key. Motivated by the vast potential of applications, we initiate the study of multi-key HS (M-HS) which allows evaluation of signatures under different keys in the *insider corruption model*, where some or even all the signers are corrupt. We also study other multi-key extensions, namely, hierarchical HS (M-HiHS) for delegation of signing power over any supersets of a specified set of messages, and key-message-HS (M-KMHS) for evaluation of signatures under different keys with respect to both keys and messages. We thus also introduce the concept of key-homomorphism in signatures, which leads to the notion of multi-key key-HS (M-KHS) for evaluation of signatures with respect to keys only.

Notion-wise, our result shows that M-HS can act as a central notion since all its seemingly different extensions are almost all equivalent (except one which assumes collision-resistant hash functions). In particular, this suggests that key-homomorphism and message-homomorphism in signatures are identical in nature. As a sample application, we show that M-KHS implies decentralized attribute-based signatures (D-ABS). Our work also provides the first (leveled) fully KHS and the first (D-)ABS for circuits from standard assumptions.

Surprisingly, there is a huge gap between homomorphism in a single space and in two spaces. Indeed all existing (leveled) fully homomorphic signature schemes support only a single signer. In the multi-space setting, we construct M-HS from the adaptive zero-knowledge succinct non-interactive argument of knowledge (ZK-SNARK) (and other standard assumptions). We also show that two-key HS implies functional signatures. Our study equips the literature with a suite of signature schemes allowing different kinds of flexible evaluations.

1 Introduction

Homomorphic signatures (HS) allow evaluation of signed messages by producing a signature on a function of messages signed by the same key. HS has undergone great development, notably from supporting either only addition or multiplication [BFKW09, GKKR10, BF11b, CFW12, Fre12, LPJY13], to bounded-degree polynomials [BF11a, CFW14], and even to (leveled) fully homomorphic operations which allow evaluation of general circuits of a-priori bounded depth [GVW15, BFS14]. Beyond unforgeability, some works also consider privacy notions such as context-hiding [ABC$^+$12, ALP12, ALP13]. HS is a handy tool for applications which require computation on authenticated data, for example, verifiable computation. The state-of-the-art HS is restricted to the single-key setting until the recent concurrent work of Fiore *et al.* [FMNP16] and David and Daniel [DS16] who define multi-key homomorphic signatures with varying level of security. Independent from their work, in this paper, we initiate the study of HS under multi-key in the *insider corruption model*, as well as key-homomorphism in (message-)homomorphic signatures.

To illustrate the usefulness of multi-key HS that is homomorphic in both key space and message space, which we call multi-key key-message-homomorphic signatures (M-KMHS), consider a scenario where two managers Alice and Bob in a company are jointly making a decision. They sign their decisions m_1 and m_2 with their signing keys associated with attributes x_1 and x_2 respectively. Is it possible for a secretary Charlie to derive a signature on their joint decision, and prove that it is from combining those of the two managers?

Technically, we ask whether a public evaluator can derive a signature of m under attribute x, where $(x, m) = g((x_1, m_1), (x_2, m_2))$ for some functions g. In other words, such a signature scheme supports operations on both the key space and message space. In particular, if $x = x_1 = x_2$, $g((x_1, m_1), (x_2, m_2)) = (x, g_2(m_1, m_2))$, the above syntax captures (regular) homomorphic signatures. On the other hand, if $m = m_1 = m_2$ and $g((x_1, m_1), (x_2, m_2)) = (g_1(x_1, x_2), m)$, it becomes a signature scheme with homomorphism in the key space, or what we call key-homomorphic signatures (KHS), which itself is a new notion never considered explicitly in the literature. In the construction of a flexible credentials network, where nodes can, for example, combine and jointly issue/delegate credentials, multi-key key-message-homomorphic signatures (M-KMHS) and multi-key hierarchical homomorphic signatures (M-HiHS) are useful.

To formalize the above notion, we begin with the simpler case where homomorphism only occurs in the message space, namely multi-key homomorphic signatures (M-HS). In M-HS, users with different secret and public key pairs can sign on different messages, so that a public evaluator can perform homomorphic evaluations of a function g on all message-signature pairs to produce a signature which verifies under the union of all input public keys. A distinctive property of M-HS, which is absent in the single-key setting, is that we require M-HS to be secure against *insider attack*. That is, suppose signer A signs a message m_A while the evaluator colludes with another signer B, we require this coalition can only produce signatures on $(g, g(m_A, \cdot))$. In other words, signer B cannot falsely claim that m is computed from some value m'_A not contributed by signer A. This property is essential for various applications of M-HS to be realistic. For example, in signed majority vote, if the underlying M-HS is not secure against *insider attack*, any voter can manipulate the voting result by producing a signature on it, while in the case that it is secure against *insider attack*, a malicious signer can only control its own vote - no better than signing honestly.

As an immediate application, M-HS can be used along with multi-key homomorphic encryption [LTV12, CM15, MW16, PS16] to achieve verifiable secure multi-party computation. Apart from being a powerful primitive on its own, M-HS implies several of its seeming different extensions and variants, including multi-key hierarchical homomorphic signatures (M-HiHS), multi-key key-message-homomorphic signatures (M-KMHS), and multi-key key-homomorphic signatures (M-KHS), whose functionality will be explained below.

1.1 Our Results

Multi-key homomorphic signatures. We present multi-key homomorphic signatures (M-HS), a generalization of homomorphic signatures which allows a public evaluator to transform signatures of different sets of messages M signed under different public keys to a signature of $(g, g(\ldots, M, \ldots))$ signed under a combined public key. We define a strong security notion of M-HS called *unforgeability under corruption*, which requires that a coalition of the evaluator and a set of malicious signers cannot forge a signature of (g, m), where the resulting message m is not in the range of g restricted by the input of the honest signer. The coalition can also choose to corrupt the only honest signer. In this case, we require that it is infeasible to forge a signature of (g, m), where the resulting message m is not in the range of g. Interestingly, such a definition also makes sense in the single-key setting, where we require that even the signer itself cannot produce a signature on (g, m) where m is not in the range of g.

Implications to and from existing notions. Treated as a central hub of our work, we study how M-HS is related to other notions. First, we show that M-HS can be constructed from the adaptive zero-knowledge succinct non-interactive argument of knowledge (ZK-SNARK). There are some impossibility results regarding the security of SNARK in the presence of oracles (O-SNARK) [FN16]. In particular, they showed that there exists a secure signature scheme Σ such that every candidate construction of O-SNARK does not satisfy adaptive proof of knowledge with respect to the signing oracle of Σ. Fortunately, there are at least two ways to circumvent this impossibility result. The first approach is to use a ZK-SNARK with a "strong" adaptive proof of knowledge property [BGI14, FN16], where the extractor takes as input an additional trapdoor and ignores the random tape of the adversary. By a recursive witness extraction technique, we show that strong ZK-SNARK implies poly-hop H-HS. The second approach is to use the weaker (O-)SNARK and make some extra assumption about the SNARK or the underlying signature scheme [FN16, Section 5]. This approach yields a constant-hop M-HS.

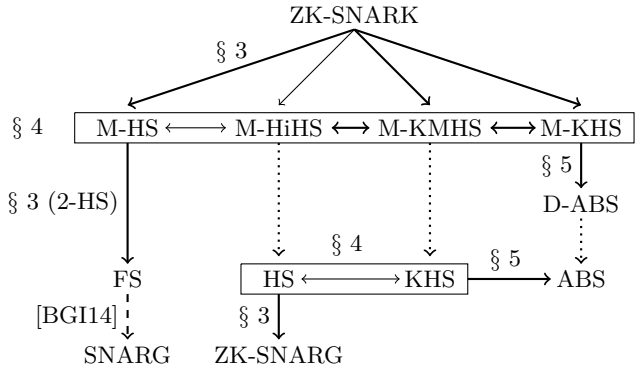

Fig. 1. Relations between all notions considered in this work: Solid arrows represent implications shown in this work. Dashed and dotted arrows represent existing and trivial implications respectively. Thin arrows represent the additional use of CRHF.

Then, we show that functional signatures (FS) [BGI14] can be constructed from a two-key M-HS (2-HS). Since the existence of succinct functional signatures implies the existence of succinct non-interactive argument (SNARG), we obtain as a corollary that the existence of 2-HS implies the existence of SNARG. Even stronger, we show that the existence of single-key HS which is unforgeable under corruption implies the existence of ZK-SNARG.

Equivalence of seemingly different variants. We then proceed to the relations between M-HS and its extensions and variants. Concretely, we propose several seemingly different notions listed as follows:

- Multi-Key Hierarchical Homomorphic Signatures (M-HiHS): extends M-HS such that a signer can delegate its signing power over any supersets of a specified set M.
- Multi-Key Key-Message-Homomorphic Signatures (M-KMHS): extends M-HS such that a public evaluator can evaluate signatures of a set of messages M signed by a key associated with a set of attributes X to a signature of M' signed by a key associated with the attribute X', where $(X', M') = g(X, M)$.
- Multi-Key Key-Homomorphic Signatures (M-KHS): allows a public evaluator to evaluate signatures of M signed by a key associated with attributes X to one signed by a key associated with the attribute $g(X)$.

Although these notions are seemingly stronger than or at least different from M-HS, we show that their existence are all equivalent to that of M-HS, up to the existence of collision-resistant hash function (CRHF) for M-HiHS. In particular, their corresponding special cases, HS and KHS, are also equivalent. Conceptually, this result bridges the gap between message- and key-homomorphism in the signatures setting. When instantiated by state-of-the-art (leveled) fully homomorphic signatures [GVW15], we obtain (leveled) fully key-homomorphic signatures (FKHS) which inherit all the nice properties, such as security based on (standard) lattices in the standard model.

Applications. Apart from grand applications such as multi-party verifiable computation, we are also interested in the implications of M-HS to more basic primitives. As an example, we show that (linkable) decentralized attribute-based signatures (D-ABS) can be constructed from M-KHS generically. When instantiated by the FKHS above, we obtain ABS for general circuits. Our scheme satisfies linkable anonymity. That is, while a signature does not leak any information about the attributes under which it is signed beyond whether they satisfy the policy of the verifier, different signatures from the same signer are publicly linkable. By a generic transformation using non-interactive witness-indistinguishable (NIWI) proofs, the resulting scheme achieves full anonymity.

To summarize, Figure 1 illustrates the relations between all notions studied in this work.

1.2 Related Work

Concurrent Work Fiore *et al.* [FMNP16] independently and concurrently propose the notion of multi-key homomorphic authenticators, which is a generalization of M-HS and multi-key homomorphic MAC. They extend HS by Gorbunov *et al.* [GVW15] to M-HS based on standard lattice assumptions, and introduce multi-key-homomorphic MAC based on pseudorandom functions. Their work adopts a weaker security model, in which the adversary must output a forgery that passes verification under non-corrupted keys. However, the inability to protect against insider attacks limits its application. It is claimed [FMNP16] that preventing insider attacks is impossible by the following argument: For general functions, controlling a few inputs implies controlling the function output. We find the claim inaccurate as there is a large class of functions for which this is not the case, *e.g.*, functions with AND gates, majority gates, and threshold gates. Our work, in contrast, constructs M-HS which prevent insider attacks, at the cost of heavier assumptions, *i.e.*, the existence of SNARKs. This property is essential for constructing various new notions related to multi-key-homomorphic signatures which allow wider applications.

Another independent and concurrent work by David and Daniel [DS16] proposes Φ-KHS, and constructs various simpler primitives based on different classes of key-homomorphisms Φ. The focus of our work is different in the sense that, we first define M-HS and its relation to other complex primitives, then extends it to M-KMHS and M-HiHS, and finally considers M-KHS as a special case. A major difference between their definition of KHS and ours is that, they define key-homomorphism directly over the key space while we consider the attribute space associated with the key. Their work also defines M-HS, with a security model which is stronger than that of Fiore *et al.* [FMNP16] but weaker than ours: They allow corruption of all but one signer, and the forgery must pass verification under a set of public keys including the non-corrupted one.

Key Homomorphism Key-homomorphism has been studied in some earlier works in the context of threshold fully homomorphic encryption [AJL+12] and pseudorandom functions [BLMR13]. The main inspiration of defining KHS in our work comes from the study of Boneh *et al.* [BGG+14] in the context of key-homomorphic encryption (KHE), who formulated KHE and constructed it based on the learning with errors problem. Furthermore, they used KHE to construct attribute-based encryption for general circuits with short secret keys. Inspired by their work, we study the corresponding notions in the signatures setting, namely KHS, and attribute-based signatures (ABS) for general circuits.

Unlike homomorphic encryption (HE) which allows homomorphic operations on the ciphertexts with respect to the plaintexts, KHE allows homomorphic operations on the ciphertexts with respect to the public keys. As the plaintexts are private while the public keys are public, KHE and HE are inherently different. For signature schemes, however, have both the messages and the public keys being public. It is natural to ask whether there is any connection between HS and KHS. Indeed, we show that the two notions are equivalent.

Following the formulation of KHE [BGG+14], we assume that public keys embed meaningful attributes, since performing operations on random looking keys may not be that meaningful intuitively which limits the application. A concurrent work by Derler and Slamanig [DS16] investigates KHS in the more literal setting, and uses it to construct more basic primitives such as ring signatures. They also define M-HS with a weaker unforgeability notion where no corruption is allowed.

Key-homomorphism in signatures is also considered in different extents in delegatable functional signatures (DFS) [BMS16] and operational signature scheme (OSS) [BDF+14]. In the former, the evaluator must use its secret key to derive signatures. The verification algorithm then takes as input both the public key of the original signature as well as the public key of the evaluator. In the latter, the evaluation algorithm takes as input tuples consisting of an identity, a message, and a signature. It outputs another tuple to a targeted identity. DFS is constructed generically from trapdoor permutations, while OSS is constructed from indistinguishability obfuscation and one-way functions. They thus serve as proof-of-concept without giving much intuition of how to achieve key-homomorphism in signatures.

Other related notions include policy-based signatures [BF14], in which a policy-dependent signing key can only sign messages satisfying the specified policy, and functional signatures [BGI14], in which a functional signing key can only sign messages in the range of the specified function.

Other Multi-Key Signatures The simplest variant of signatures which considers signatures in multi-users scenarios is arguable multi-signature, which allows a group of user to sign on a single message (without the powerful homomorphic property in M-HS). Aggregate signatures allow signatures from different users to be combined into a single signature, and the validity of it implies the validity of all underlying signatures. This property is similar to our proposed security under corruption of M-HS, which can be considered as generalization of aggregate signature that allows the collector of signatures to define the aggregate function from the set of admissible function.

2 Preliminaries

Let λ be the security parameter. We use $\mathsf{negl}(\lambda)$ to denote functions which are negligible in λ. Barred variables are vectors, *e.g.*, $\bar{x}$. If A is a probabilistic algorithm, $x \leftarrow \mathsf{A}(\cdot)$ denotes assigning the output from the execution of A to the variable x. For a set S, $x \leftarrow S$ denotes the sampling of a uniformly random $x \in S$. The empty string and set are denoted by ϵ and ϕ respectively. Let $M_1, \ldots, M_k$ be sets of possibly different sizes. We use $p_{k,j}$ to denote the projection function such that for each $m_{k,j} \in M_k$, we have $m_{k,j} \leftarrow p_{k,j}(M_1, \ldots, M_K)$. id and id_2 denote one- and two-dimensional identity functions respectively.

2.1 Settings and Notations

Let $N = \mathsf{poly}(\lambda)$ be an a-priori bounded maximum number of hops of homomorphic evaluation. That is, a freshly signed message m can at most pass through N functions. There is however no further restriction on the number of inputs and circuit depth of each function as long as it is efficiently computable.

We assume that any function obtained by *concatenation* $\circ$ keeps the concatenated representation (without simplification). Thus, given a function g, one can efficiently write down its evaluation process as a *evaluation tree*.

As a consequence, there exists a polynomial time algorithm Hop which, on input a function g, outputs the depth of the evaluation tree of g, which represents the *number of hops* $\mathsf{Hop}(g)$ passed through in the evaluation process of g.

We denote the combination of public keys and tags by $\mathsf{PK} := \mathsf{Comb}(\mathsf{pk}_1, \ldots, \mathsf{pk}_k)$ and $\mathcal{T} := \mathsf{Comb}(\tau_1, \ldots, \tau_k)$ respectively. In the rest of the paper, we define the combination function as union, that is, $\mathsf{Comb}(\mathsf{pk}_1, \ldots, \mathsf{pk}_k) := \cup_{k=1}^{K} \mathsf{pk}_k$ and $\mathsf{Comb}(\tau_1, \ldots, \tau_k) := \cup_{k=1}^{K} \tau_k$. Note that one can use other functions to combine public keys and tags.

2.2 Succinct Zero-Knowledge Non-Interactive Argument

Definition 1 (ZK-SNARG). *$\Pi = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Vf})$ is an* adaptive zero-knowledge succinct non-interactive argument *(ZK-SNARG) for a language $L \in \mathsf{NP}$ with the witness relation $\mathcal{R}$ if it satisfies the following properties:*

- ***Completeness**: For all x, w such that $\mathcal{R}(x, w) = 1$, and for all strings $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$, we have $\mathsf{Vf}(\mathsf{crs}, x, \mathsf{Prove}(x, w, \mathsf{crs})) = 1$.*
- ***Adaptive Soundness**: If $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$ is sampled uniformly at random, then for all PPT adversaries $\mathcal{A}$, the probability that $\mathcal{A}(\mathsf{crs})$ will output a pair (x, π) such that $x \notin L$ but $\mathsf{Vf}(\mathsf{crs}, x, \pi) = 1$, is at most $\mathsf{negl}(\lambda)$.*
- ***Succinctness**: For all x, w such that $\mathcal{R}(x, w) = 1$, if $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$ and $\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)$, then there exists a universal polynomial $p(\cdot)$ that does not depend on the relation $\mathcal{R}$, such that $|\pi| \leq p(k + \log t)$, where t denotes the runtime of the relation R associated with language L.*
- ***Adaptive Zero-Knowledge**: There exists a PPT algorithm $\mathcal{S} = (\mathcal{S}^{\mathsf{crs}}, \mathcal{S}^{\mathsf{Prove}})$ such that, for all PPT adversaries $\mathcal{A}$,*

$$\begin{aligned} &|\Pr[\mathcal{A}^{\mathsf{Prove}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) \to 1 : \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)] - \\ &\Pr[\mathcal{A}^{\mathcal{S}'(\mathsf{crs},\mathsf{td},\cdot,\cdot)}(\mathsf{crs}) \to 1 : (\mathsf{crs}, \mathsf{td}) \leftarrow \mathcal{S}^{\mathsf{crs}}(1^\lambda)]| = \mathsf{negl}(\lambda) \end{aligned}$$

where $\mathcal{S}'(\mathsf{crs}, \mathsf{td}, x, w) = \mathcal{S}^{\mathsf{Prove}}(\mathsf{crs}, \mathsf{td}, x)$.

Definition 2 ((Strong) ZK-SNARK [BGI14, FN16]). *A ZK-SNARG* $\Pi = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Vf})$ *is a (strong)* adaptive zero-knowledge succinct non-interactive argument of knowledge *(ZK-SNARK) for a language* L *in* NP *with witness relation* $\mathcal{R}$ *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that, for all PPT provers* P^**, there exists a PPT algorithm* $\mathsf{E}_{P^*} = (\mathsf{E}^1_{P^*}, \mathsf{E}^2_{P^*})$ *such that for* every *adversaries* $\mathcal{A}$*,*

$$\begin{aligned}&|\Pr[\mathcal{A}(\mathsf{crs}) \to 1 : \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)] - \\ &\Pr[\mathcal{A}(\mathsf{crs}) \to 1 : (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{E}^1_{P^*}(1^\lambda)]| = \mathsf{negl}(\lambda)\end{aligned}$$

and

$$\begin{aligned}&|\Pr[\mathsf{Vf}(\mathsf{crs}, x, \pi) = 1 \wedge (x, w^*) \notin \mathcal{R} : (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{E}^1_{P^*}(1^\lambda), \\ &(x, \pi) \leftarrow P^*(\mathsf{crs}), w^* \leftarrow \mathsf{E}^2_{P^*}(\mathsf{crs}, \mathsf{td}, x, \pi)]| = \mathsf{negl}(\lambda)\end{aligned}$$

where the probabilities are taken over $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{E}^1_{P^*}(1^\lambda)$*, and the random coin of the extractor* $\mathsf{E}^2_{P^*}$*.*

Definition 3 (O-SNARKs [FN16]). *A ZK-SNARG* $\Pi = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Vf})$ *is an* adaptive zero-knowledge succinct non-interactive arguments of knowledge in the presence of oracles for $\mathbb{O}$ *(O-SNARKs) for the oracle family* $\mathbb{O}$ *and a language* $L \in \mathsf{NP}$ *with the witness relation* $\mathcal{R}$ *if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that, for all PPT provers* P^**, there exists a PPT algorithm* E_{P^*} *such that*

$$\begin{aligned}&|\Pr[\mathsf{Vf}(\mathsf{crs}, x, \pi) = 1 \ \wedge \ (x, w^*) \notin \mathcal{R} : \\ &\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda), \mathcal{O} \leftarrow \mathbb{O}; (x, \pi) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{crs}), w \leftarrow E_{P^*}(\mathsf{crs}, \mathsf{qt})]| = \mathsf{negl}(\lambda)\end{aligned}$$

where $\mathsf{qt} = \{q_i, \mathcal{O}(q_i)\}$ *is the transcript of all oracle queries and answers made and received by* $\mathcal{A}$ *during its execution.*

3 Multi-Key Homomorphic Signatures (M-HS)

We define a new primitive called multi-key homomorphic signatures (M-HS). M-HS allows an arbitrary number of signers to generate keys and sign messages independently in a fully distributed manner. Suppose that each signer k signs a set of messages M_k with its secret key sk_k to produce a set of signatures Σ_k. An evaluator can then publicly evaluate a function g over the message-signature pairs (M_k, Σ_k) to derive a signature of (m, g) where $m = g(M_1, \ldots, M_K)$. Syntactically, M-HS generalizes the normal homomorphic signatures (HS) since it reduces to HS when all K secret keys are owned by the same party.

In the multi-signer setting, we must carefully analyze unforgeability when the adversary can corrupt some signers or even maliciously generate some key pairs. Such an insider attack is absent in HS since there is only one signer and hence one signing key involved with a signature. We formulate the unforgeability against insider corruption, which requires that such group of corrupt signers cannot produce signatures of (m, g), where the message m is outside the range of the function g restricted by the inputs of the uncorrupted signers. Security against insider attack is useful when the output of the function cannot be fully controlled by a few inputs, *e.g.*, functions with AND, majority, and threshold gates. Concretely, to illustrate the meaning of a successful forgery, consider the following specific configuration: Let g be the product function and $M_k \in \{0, 1\}$. As long as $M_k = 0$ for some uncorrupted signer k, the adversary is unable to produce a signature of $(1, g)$.

More interestingly, this requirement actually still makes sense even when there is only one signer who is also the adversary. In this case, unforgeability against insider corruption implies that even the signer cannot produce a signature of (m, g) if there does not exist M' such that $m = g(M')$. Furthermore, if the signature scheme is context-hiding, the signature of (m, g) can be regarded as an adaptive zero-knowledge succinct non-interactive argument (SNARG) of the NP language $\{(m, g) : \exists M' \ s.t. \ m = g(M')\}$ as long as g is efficiently computable.

3.1 Definitions

Syntax. A multi-key homomorphic signature scheme (M-HS) for a class $\mathcal{G}$ of admissible functions consists of the PPT algorithms $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$ defined as follows:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ inputs the security parameter λ. It outputs a public parameter pp which is an input to all algorithms implicitly. The public parameter also defines the message space $\mathcal{M}$ and the function family $\mathcal{G}$ which contains the identity function $\mathsf{id} : \mathcal{M} \to \mathcal{M}$.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$ inputs the public parameter. It outputs the public key pk and secret key sk. When an algorithm takes as input a secret key sk, we assume sk contains its corresponding public pk implicitly.
- $\Sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, \tau, M)$ inputs the secret key sk, a tag $\tau \in \{0,1\}^*$, and a set of messages $M \in \mathcal{M}^*$. It outputs a set of signatures Σ.
- $b \leftarrow \mathsf{Vf}(g, \mathsf{PK}, \mathcal{T}, m, \sigma)$ inputs a function $g \in \mathcal{G}$, a (possibly combined) public key PK, a (possibly combined) tag $\mathcal{T}$, a message $m \in \mathcal{M}$, and a signature σ. It outputs a bit $b = 0$ or $b = 1$, indicating whether m is the output of the function g over some signed data under tag $\mathcal{T}$.
- $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)_{k=1}^K)$ inputs a function $g \in \mathcal{G}$, and, from each contributor, a public key (possibly combined), a tag (possibly combined), a set of messages, and a set of signatures $(\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)$, where $k \in [K]$. It outputs a signature σ signing the evaluated data $m = g(M_1, \ldots, M_k)$ under the combined public key $\mathsf{PK} = \cup_{k=1}^K \mathsf{PK}_k$ and combined tags $\mathcal{T} = \cup_{k=1}^K \mathcal{T}_k$. We assume that the evaluator knows the history of the evaluated functions which produce M_k.

Correctness. For any $\mathsf{pp} \in \mathsf{Setup}(1^\lambda)$, any $K = \mathsf{poly}(\lambda)$ and $k \in [K]$, any $(\mathsf{pk}_k, \mathsf{sk}_k) \in \mathsf{KGen}(\mathsf{pp})$, any $M_k \in \mathcal{M}^*$, any $\tau_k \in \{0,1\}^*$, and any $g, h_{k,j} \in \mathcal{G}$ with appropriate dimensions, it holds that

- (Signing.) if $\Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_k, \tau_k, M_k)$, then $\mathsf{Vf}(\mathsf{id}, \mathsf{pk}_k, \tau_k, m_{k,j}, \sigma_{k,j}) = 1$ for each $\sigma_{k,j} \in \Sigma_k$ and $m_{k,j} \in M_k$; furthermore,
- (Evaluation.) for any Σ_k, if $\mathsf{Vf}(h_{k,j}, \mathsf{PK}_k, \mathcal{T}_k, m_{k,j}, \sigma_{k,j}) = 1$ for all $\sigma_{k,j} \in \Sigma_k$ and $m_{k,j} \in M_k$, $\mathsf{Hop}(h_{k,j}) \leq N - 1$, and $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)_{k=1}^K)$, then $\mathsf{Vf}(g \circ \bar{h}, \mathsf{PK}, \mathcal{T}, g(M_1, \ldots, M_k), \sigma) = 1$, where $\mathsf{PK} = \cup_{k=1}^K \mathsf{PK}_k$ and $\mathcal{T} = \cup_{k=1}^K \mathcal{T}_k$.

Unforgeability. Consider the following security game $\mathsf{cEUF\text{-}CMA}$ (*existential unforgeability under corruption and chosen message attack*) between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

- The challenger $\mathcal{C}$ runs $\mathsf{pp} \in \mathsf{Setup}(1^\lambda)$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$, and gives pp, pk to the adversary $\mathcal{A}$.
- The adversary $\mathcal{A}$ adaptively issues a polynomial number of signing queries. In each signing query $q \in [Q]$ where $Q = \mathsf{poly}(\lambda)$, $\mathcal{A}$ chooses a tag $\tau_q \in \{0,1\}^*$, and a set of data $M_q \in \mathcal{M}^*$. The challenger $\mathcal{C}$ responds with $\Sigma_q \leftarrow \mathsf{Sig}(\mathsf{sk}, \tau_q, M_q)$. It also decides whether to issue a corruption query which the challenger $\mathcal{C}$ responds with sk.
- The adversary $\mathcal{A}$ outputs a function $g^* \in \mathcal{G}$, a public key PK^*, a tag $\mathcal{T}^* \in \{0,1\}^*$, a message $m^* \in \mathcal{M}$, and a signature σ^*. The experiment outputs 1 if and only if $\mathsf{Vf}(g^*, \mathsf{PK}^*, \mathcal{T}^*, m^*, \sigma^*) = 1$, and $(g^*, \mathsf{PK}^*, \mathcal{T}^*, m^*, \sigma^*)$ is either a forgery such that
  - *Type-I*: the signer is not corrupted, $\mathsf{pk} \in \mathsf{PK}^*$ and $\forall\ \tau_q \in \mathcal{T}^*$, $m^* \notin g^*(\cdots, M_q, \cdots)$
  - *Type-II*: $m^* \notin R_{g^*}$, where R_{g^*} is the range of g^*.

We say that the scheme is unforgeable under corruption (cEUF-CMA-secure) if, for all PPT adversaries $\mathcal{A}$, we have $\Pr\{\mathsf{cEUF\text{-}CMA}_{\mathcal{HS},\mathcal{A}} = 1\} \leq \mathsf{negl}(\lambda)$. We say that the scheme is unforgeable (EUF-CMA-secure) if the adversary $\mathcal{A}$ is not allowed to issue the corruption query in the game.

We can define a game K-$\mathsf{cEUF\text{-}CMA}$ similar to above, except there are $K = \mathsf{poly}(\lambda)$ signers and $\mathcal{A}$ can corrupt any of the K signers and query signatures from any signers. The valid forgery is revised correspondingly as follows:

- *Type-I*: at least one signer $k \in K$ where $\mathsf{pk}_k \in \mathsf{PK}^*$ is not corrupted and $\forall\ \mathcal{T}_{q,k} \in \mathcal{T}^*$, $m^* \notin g^*(\cdots, M_{q,k}, \cdots)$
- *Type-II*: $m^* \notin R(g^*)$, where $R(g^*)$ is the range of g^*.

A scheme is said to be unforgeable under K-bounded corruption (K-cEUF-CMA-secure) if, for all PPT adversaries $\mathcal{A}$, $\Pr\{K\text{-}\mathsf{cEUF\text{-}CMA}_{\mathcal{HS},\mathcal{A}} = 1\} \leq \mathsf{negl}(\lambda)$. This is a generalization of the above, as the notions where one or no corruption is allowed can be obtained by letting $K = 1$ and $K = 0$ respectively. We have the following theorem showing cEUF-CMA-security and K-cEUF-CMA-security are equivalent. The proof can be found in the supplementary material.

Theorem 1. *cEUF-CMA-security and K-cEUF-CMA-security are equivalent.*

Lastly, one can also further restrict the adversary in the K-cEUF-CMA game such that PK^* in the forgery must be the union of a set of the public keys of *uncorrupted* signers. Schemes satisfying this property is said to be unforgeable for uncorrupted signers.

Context Hiding. There exist a simulator $\mathcal{S} = (\mathcal{S}^{\mathsf{Setup}}, \mathcal{S}^{\mathsf{Sig}})$ such that, for all $K = \mathsf{poly}(\lambda)$, $k \in [K]$, τ_k, M_k, and g, it holds that for any PPT adversaries $\mathcal{A}$,

$$\left| \Pr\left[\begin{array}{c} \mathcal{A}(g, (\mathsf{pk}_k, \mathsf{sk}_k, \tau_k, M_k, \Sigma_k)_{k=1}^K, \sigma) \to 1 : \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow \mathsf{KGen}(\mathsf{pp}) \\ \Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_k, \tau_k, M_k) \\ \sigma \leftarrow \mathsf{Eval}(g, (\mathsf{pk}_k, \tau_k, M_k, \Sigma_k)_{k=1}^K) \end{array}\right] - \Pr\left[\begin{array}{c} \mathcal{A}(g, (\mathsf{pk}_k, \mathsf{sk}_k, \tau_k, M_k, \Sigma_k)_{k=1}^K, \sigma) \to 1 : \\ (\mathsf{pp}, \mathsf{td}) \leftarrow \mathcal{S}^{\mathsf{Setup}}(1^\lambda) \\ (\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow \mathsf{KGen}(\mathsf{pp}) \\ \Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_k, \tau_k, M_k); \\ \sigma \leftarrow \mathcal{S}^{\mathsf{Sig}}(\mathsf{td}, g, \mathsf{PK}, \mathcal{T}, g(M_1, \ldots, M_K)) \end{array}\right] \right| = \mathsf{negl}(\lambda)\,.$$

The scheme is weakly context hiding if the above holds. The strong notion requires the fresh signature indistinguishable from the evaluated one.

Succinctness. There exist a polynomial $s(\cdot)$ such that for every $\lambda \in \mathbb{N}$, $K = \mathsf{poly}(\lambda)$, $k \in [K]$, $g \in \mathcal{G}$, $\tau_k \in \{0,1\}^*$, $M_k \in \mathcal{M}^*$, it holds with probability 1 over $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$; $(\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow \mathsf{KGen}(\mathsf{pp})$; $\Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_k, \tau_k, M_k)$; that the resulting signature $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)_{k=1}^K)$ on $m = g(M_1, \ldots, M_k)$ has size $|\sigma| \leq s(\lambda, |m|)$. In particular, the signature size is independent of the sizes $|M_k|$ of the inputs to the function, and the size $|g|$ of a description of the function g.

3.2 Construction from ZK-SNARK

In the following, we provide a generic construction of M-HS from ordinary digital signatures and ZK-SNARKs, which is essentially a formalization of the multi-key generalization of the folklore construction of homomorphic signatures. The idea is straightforward. The signer signs fresh signatures using the ordinary signature scheme. To perform a homomorphic evaluation, the evaluator proves that it possesses a set of signatures on messages, and evaluation of a function on these messages produces the resulting message.

We use the proof system recursively by treating the proof as an evaluated signature, which can be further used in other proofs for further homomorphic evaluation. Similar techniques are used in the construction of homomorphic encryption with targeted malleability by Boneh *et al.* [BSW12].

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$

$\mathsf{crs}_0 := \phi$

$\mathsf{crs}_n \leftarrow \Pi_n.\mathsf{Gen}(1^\lambda)\ \forall n \in [N]$

return $\mathsf{pp} = (1^\lambda, \{\mathsf{crs}_n\}_{n=0}^N)$

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$

$(\mathsf{pk}_{\mathcal{DS}}, \mathsf{sk}_{\mathcal{DS}}) \leftarrow \mathcal{DS}.\mathsf{KGen}(1^\lambda)$

return $(\mathsf{pk}, \mathsf{sk}) := (\mathsf{pk}_{\mathcal{DS}}, \mathsf{sk}_{\mathcal{DS}})$

$b \leftarrow \mathsf{Vf}(g, \mathsf{PK}, \mathcal{T}, m, \sigma)$

parse $\sigma = (n, \sigma')$

if $n = 0 \ \wedge \ g = \mathsf{id}$ **then**

  $b \leftarrow \mathcal{DS}.\mathsf{Vf}(\mathsf{PK}, (\mathcal{T}, m), \sigma')$

  return b

elseif $n \in [N]$ **then**

  $x := (\mathsf{crs}_{n-1}, g, \mathsf{PK}, \mathcal{T}, m)$

  $b \leftarrow \Pi_n.\mathsf{Vf}(\mathsf{crs}_n, x, \sigma')$

  return b

else

  return 0

endif

$\Sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, \tau, M)$

$\sigma'_j \leftarrow \mathcal{DS}.\mathsf{Sig}(\mathsf{sk}_{\mathcal{DS}}, (\tau, m_j))\ \forall m_j \in M$

$\sigma_j := (0, \sigma'_j)$

return $\Sigma := \{\sigma_j\}_j$

$\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)_{k=1}^K)$

foreach $k \in [K], \sigma_{k,j} \in \Sigma_k$ **do**

  parse $\sigma_{k,j} = (n_{k,j}, \sigma'_{k,j})$

endfor

$n := \max_{k,j}(n_{k,j})$

$m \leftarrow g(M_1, \ldots, M_K)$

$\mathsf{PK} \leftarrow \cup_{k=1}^K \mathsf{PK}_k, \mathcal{T} \leftarrow \cup_{k=1}^K \mathcal{T}_k$

$x := (\{\mathsf{crs}_i\}_{i=1}^n, g \circ \bar{h}, \mathsf{PK}, \mathcal{T}, m)$

$w := (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)_{k=1}^K$

$\sigma' \leftarrow \Pi_{n+1}.\mathsf{Prove}(\mathsf{crs}_{n+1}, x, w)$

return $\sigma := (n+1, \sigma')$

Fig. 2. Construction of M-HS from ZK-SNARK

Concretely, we define for each hop of evaluation the languages for the proof system as follows. For each $n \in [N]$, let $\Pi_n.(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Vf})$ be a ZK-SNARK for the following recursively defined NP language L_n:

$$L_n = \begin{cases} \{(\phi, g, \mathsf{PK}, \mathcal{T}, m) : \exists (\mathsf{pk}_k, \tau_k, M_k, \Sigma_k)_{k=1}^K \ s.t. \\ \forall\ m_{k,j} \in M_k\ \forall\ \sigma_{k,j} = (0, \sigma'_{k,j}) \in \Sigma_k, \\ \mathcal{DS}.\mathsf{Vf}(\mathsf{pk}_k, (\tau_k, m_{k,j}), \sigma'_{k,j}) = 1\ \wedge \\ g(M_1, \ldots, M_K) = m\ \wedge\ \cup_{k=1}^K \mathsf{pk}_k = \mathsf{PK}\ \wedge\ \cup_{k=1}^K \tau_k = \mathcal{T}\} & n = 1 \\ \{(\{\mathsf{crs}_i\}_{i=1}^{n-1}, g \circ \bar{h}, \mathsf{PK}, \mathcal{T}, m) : \exists (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)_{k=1}^K \ s.t. \\ \forall\ m_{k,j} \in M_k\ \forall\ \sigma_{k,j} = (n_{k,j}, \sigma'_{k,j}) \in \Sigma_k, \\ \Pi_{n-1}.\mathsf{Vf}(\mathsf{crs}_{n_{k,j}}, (\{\mathsf{crs}_i\}_{i=1}^{n_{k,j}-1}, m_{k,j}, h_{k,j}, \mathsf{PK}_k, \mathcal{T}_k), \sigma'_{k,j}) = 1\ \wedge \\ g(M_1, \ldots, M_K) = m\ \wedge\ \cup_{k=1}^K \mathsf{PK}_k = \mathsf{PK}\ \wedge\ \cup_{k=1}^K \mathcal{T}_k = \mathcal{T}\} & n > 1 \end{cases}$$

where $h_{k,j}$ is a layer-2 (the layer just below the root) node of the trees $g \circ \bar{h}$ corresponding to the messages $m_{k,j}$.

Formally, we construct a multi-key homomorphic signature scheme $\mathcal{HS}$ as shown in Figure 2. Its correctness follows directly from the correctness of $\mathcal{DS}$ and Π. The context-hiding property follows from the zero-knowledge property of Π.

Next, we argue that it is unforgeable against insider corruption. The intuition is that, if the adversary outputs a signature (a proof) of a message outside the range of g restricted by the inputs of the honest signers, then either the proof is valid for a statement outside L, which breaks the soundness of Π, or it breaks the unforgeability of $\mathcal{DS}$. Specifically, one extractor among Π_n can extract signatures under some public key

pk_k of some message m not signed by the honest signer k. The proofs can be found in the supplementary material.

Theorem 2. *If one-way function exist and, for all $n \in [N]$ where $N = \mathsf{poly}(\lambda)$, Π_n is a sound strong ZK-SNARK (Definition 2), then $\mathcal{HS}$ is unforgeable under corruption.*

Theorem 3. *If one-way functions exist and, for all $n \in [N]$ where N is a constant, Π_n is a sound O-SNARK with respect to the signing oracle of $\mathcal{DS}$ (Definition 3), then $\mathcal{HS}$ is unforgeable under corruption. (Here, $\mathcal{HS}$ only supports constant-hop evaluation.)*

Theorem 4. *Suppose Π_n is adaptive zero knowledge for all $n \in [N]$, then $\mathcal{HS}$ is weakly context hiding.*

Theorem 5. *Suppose Π_n is succinct for all $n \in [N]$, then $\mathcal{HS}$ is succinct.*

3.3 Construction from Lattices

If we settle for unforgeability for uncorrupted signers only, a generalization of the (leveled) fully homomorphic signatures by Gorbunov *et al.* [GVW15] (GVW) gives an M-HS scheme. We only explain the core idea of this generalization. A concurrent work by Fiore *et al.* [FMNP16] has also formalized the idea.

The verification equation of the GVW scheme is of the form $\mathbf{V} = \mathbf{AU} + x\mathbf{G}$, where $x \in \{0, 1\}$ is the message signed, $\mathbf{A}$ and $\mathbf{V}$ are random matrices given in the public key, $\mathbf{U}$ is a random matrix of small norm acting as the signature of x, and $\mathbf{G}$ is a fixed, nicely-structured "gadget" matrix such that there exists an efficient algorithm (with abused notion as a matrix) $\mathbf{G}^{-1}$ which samples matrices of a small norm from the kernel of $\mathbf{G}$. The secret key is a "trapdoor" for $\mathbf{A}$ which allows efficient sampling of matrices of a small norm from the kernel of $\mathbf{A}$.

Suppose $\mathbf{U}_1$ and $\mathbf{U}_2$ are the signatures given by two different signers on the messages x_1 and x_2 respectively which satisfy the equations $\mathbf{V}_1 = \mathbf{A}_1\mathbf{U}_1 + x_1\mathbf{G}$ and $\mathbf{V}_2 = \mathbf{A}_2\mathbf{U}_2 + x_2\mathbf{G}$. Given the signatures, a public evaluator can compute the signatures $\mathbf{U}^+ = (\mathbf{U}_1, \mathbf{U}_2)^T$ and $\mathbf{U}^\times = (\mathbf{U}_1\mathbf{G}^{-1}\mathbf{V}_2, x_1\mathbf{U}_2)^T$ for the addition and multiplication function respectively. We can verify that these signatures satisfy $\mathbf{V}_1 + \mathbf{V}_2 = (\mathbf{A}_1, \mathbf{A}_2)\mathbf{U}^+ + (x_1 + x_2)\mathbf{G}$ and $\mathbf{V}_1\mathbf{G}^{-1}(\mathbf{V}_2) = (\mathbf{A}_1, \mathbf{A}_2)\mathbf{U}^\times + (x_1x_2)\mathbf{G}$. The treatments for adaptive security, multi-data support, and context-hiding property follow directly from the work of Gorbunov *et al.* [GVW15].

Unfortunately, such simple generalization only achieves unforgeability for uncorrupted signers, which means the forgery must only involve uncorrupted signers. This is because, if the adversary has knowledge of the trapdoor of $\mathbf{A}_k$ for some signer k, then it can sample matrices of a small norm from the kernel of $(\mathbf{A}_1, \ldots, \mathbf{A}_K)$ using standard trapdoor delegation technique. More disappointingly, even the original (single-key) scheme of GVW does not satisfy unforgeability under corruption. Crossing the gap from disallowing to allowing corruption based on standard assumptions should require substantially different techniques.

3.4 Functional Signatures from M-HS

We begin to show the power of M-HS by constructing functional signatures [BGI14] using a 2-key HS. As mentioned in the introduction, FS allows an authority with a master secret key to derive function-specific signing keys. Given a signing key for a function f, one can only sign messages in the range of f.

Definition. We recall the formal definition of functional signatures [BGI14].

Syntax. A functional signature scheme for a message space $\mathcal{M}$, and a function family $\mathcal{F} = \{f : \mathcal{D}_f \to \mathcal{M}\}$ consists of algorithms $\mathcal{FS}.(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf})$.

- $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathcal{FS}.\mathsf{Setup}(1^\lambda)$: inputs the security parameter λ; and outputs the master secret key msk and master public key mpk.
- $\mathsf{sk}_f \leftarrow \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f)$: inputs the master secret key msk and a function $f \in \mathcal{F}$; and outputs a secret key sk_f for f.

– $(f(m), \sigma) \leftarrow \mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_f, m)$: inputs the secret key sk_f for a function $f \in \mathcal{F}$, and message $m \in \mathcal{D}_f$; and outputs $f(m)$ and a signature of $f(m)$.
– $b \leftarrow \mathcal{FS}.\mathsf{Vf}(\mathsf{mpk}, m, \sigma)$: inputs the master public key mpk, a message m, and a signature σ; and outputs 1 if the signature is valid.

Correctness. We require that for any $\lambda \in \mathbb{N}$, any $(\mathsf{mpk}, \mathsf{msk}) \in \mathcal{FS}.\mathsf{Setup}(1^\lambda)$, any $f \in \mathcal{F}$, any $\mathsf{sk}_f \in \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f)$, any $m \in \mathcal{D}_f$, if $(m^*, \sigma) \leftarrow \mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_f, m)$, then $\mathcal{FS}.\mathsf{Vf}(\mathsf{mpk}, m^*, \sigma) = 1$.

Unforgeability. The scheme is unforgeable if the advantage of any PPT adversary $\mathcal{A}$ in the following game is negligible:

– The challenger generates $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathcal{FS}.\mathsf{Setup}(1^\lambda)$, and gives mpk to $\mathcal{A}$.
– $\mathcal{A}$ is allowed to query a key generation oracle $\mathcal{O}_{\mathsf{key}}$ and a signing oracle $\mathcal{O}_{\mathsf{sign}}$. These oracles share a dictionary indexed by tuples $(f, i) \in \mathcal{F} \times \mathbb{N}$, whose entries are signing keys: $\mathsf{sk}_f \leftarrow \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f)$. This dictionary keeps track of the keys that have been previously generated during the game. The oracles are defined as follows:
  - $\mathcal{O}_{\mathsf{key}}(f, i)$
    * If there exists an entry for the key (f, i) in the dictionary, output the corresponding value, sk_f^i.
    * Otherwise, sample a fresh key $\mathsf{sk}_f^i \leftarrow \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f)$, then add an entry $(f, i) \rightarrow \mathsf{sk}_f^i$ to the dictionary and output sk_f^i.
  - $\mathcal{O}_{\mathsf{sign}}(f, i, m)$
    * If there exists an entry for the key (f, i) in the dictionary, output $\sigma \leftarrow \mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_f^i, m)$.
    * Otherwise, sample a fresh key $\mathsf{sk}_f^i \leftarrow \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f)$, then add it to the entry (f, i) of the dictionary, and output $\sigma \leftarrow \mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_f^i, m)$.
  - $\mathcal{A}$ wins if it can produce (m^*, σ) such that:
    * $\mathcal{FS}.\mathsf{Vf}(\mathsf{mpk}, m^*, \sigma) = 1$;
    * There does not exist m such that $m^* = f(m)$ for any f which was sent as a query to the $\mathcal{O}_{\mathsf{key}}$ oracle;
    * There does not exist a query (f, m) to $\mathcal{O}_{\mathsf{sign}}$ where $m^* = f(m)$.

Function-Privacy. The scheme is function-private if the advantage of any PPT adversary $\mathcal{A}$ in the following game is negligible:

– The challenger honestly generates $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathcal{FS}.\mathsf{Setup}(1^\lambda)$, and gives mpk and msk (w.l.o.g. this includes the randomness used in Setup) to $\mathcal{A}$.
– $\mathcal{A}$ chooses a function f_0 and receives an (honestly generated) secret key $\mathsf{sk}_{f_0} \leftarrow \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f_0)$.
– $\mathcal{A}$ chooses a second function f_1 for which $|f_0| = |f_1|$ (where padding can be useful if there is a known upper bound) and receives an (honestly generated) secret key $\mathsf{sk}_{f_1} \leftarrow \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f_1)$.
– $\mathcal{A}$ chooses a pair of value m_0, m_1 *s.t.* $|m_0| = |m_1|$ and $f_0(m_0) = f_1(m_1)$.
– The challenger selects a random bit $b \leftarrow \{0, 1\}$ and generates a signature on the image message $m' = f_0(m_0) = f_1(m_1)$ using secret key sk_{f_b}, and gives the resulting signature $\sigma \leftarrow \mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_{f_b}, m_b)$ to $\mathcal{A}$.
– $\mathcal{A}$ outputs a bit b', and wins the game if $b' = b$.

Succinctness. There exist a polynomial $s(\cdot)$ such that for every $k \in \mathbb{N}$, $f \in \mathcal{F}$, $m \in \mathcal{D}_f$, it holds with probability 1 over $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathcal{FS}.\mathsf{Setup}(1^\lambda)$; $\mathsf{sk}_f \leftarrow \mathcal{FS}.\mathsf{KGen}(\mathsf{msk}, f)$; $(f(m), \sigma) \leftarrow \mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_f, m)$ that the resulting signature on $f(m)$ has size $|\sigma| \leq s(k, |f(m)|)$. In particular, its size is independent of the size $|m|$ of the input to the function, and the size $|f|$ of the description of f.

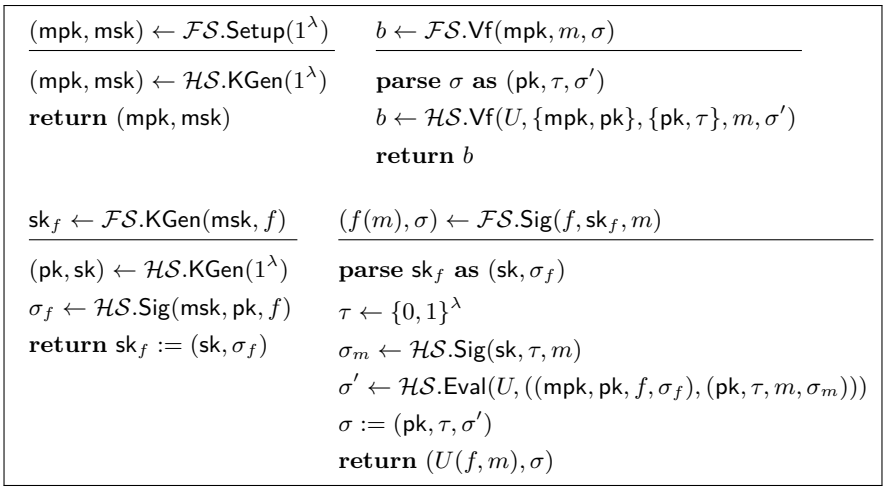

Fig. 3. Construction of FS from M-HS

Construction. We construct FS using an M-HS which supports 1-hop evaluation of signatures signed under two different keys. The functional signing key consists of a fresh M-HS secret key sk, and a signature σ_f of the function f signed under the master secret key. To sign a function output $f(m)$, the signer simply signs the input message m using sk, and evaluates the signatures σ_f and σ_m of the function and the message respectively using the universal circuit U, which is defined as $U(f, m) = f(m)$ for any function f and message m. The unforgeability under corruption of the M-HS scheme is crucial, for otherwise the signer might be able to produce a signature on any message (possibly outside the range of f) using sk.

Formally, let $\mathcal{HS}.(\mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$ be a 1-hop 2-HS scheme for a function family $\mathcal{G} = \{U : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^k\}$, where U is the universal circuit taking as input a circuit f with description size s and its n-bit input m, and computes $U(f, m) = f(m)$ of length k. Let $\mathcal{M} = \{0,1\}$. We construct a functional signature scheme $\mathcal{FS}.(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf})$ for the function family $\mathcal{F} = \{f : \{0,1\}^n \to \{0,1\}^k \ s.t. \ |f| = s\}$ as shown in Figure 3.

Theorem 6. *Suppose $\mathcal{HS}$ is correct, then $\mathcal{FS}$ is correct.*

Theorem 7. *Suppose $\mathcal{HS}$ is cEUF-CMA-secure, then $\mathcal{FS}$ is unforgeable.*

Theorem 8. *Suppose $\mathcal{HS}$ is weakly context-hiding, then $\mathcal{FS}$ is function-private.*

Theorem 9. *Suppose $\mathcal{HS}$ is succinct then $\mathcal{FS}$ is succinct.*

Since the existence of secure functional signatures implies that of SNARGs [BGI14], we have the following corollary.

Corollary 1. *Suppose cEUF-CMA-secure and weakly context-hiding 1-hop 2-HS exists, then SNARG for* NP *exists.*

3.5 ZK-SNARG from M-HS

In the previous subsection, we show that the existence of 2-HS implies that of FS, which in turn implies the existence of SNARGs. However, there are two limitations. First, the existence of FS only implies the existence of non-zero-knowledge SNARGs. Second, as constructing 2-HS might be significantly more difficult than constructing (1-)HS (both with unforgeability under corruption), it is desirable to construct (ZK-)SNARG directly from HS.

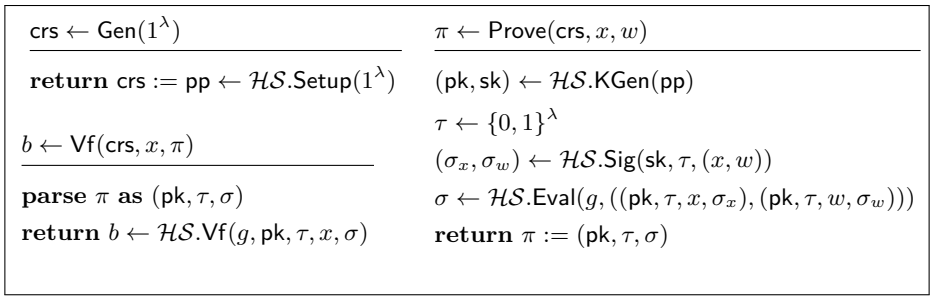

Fig. 4. Construction of SNARG from M-HS

The direct construction is as follows. Let the public parameters of M-HS be the common reference string. The prover generates a fresh M-HS key and signs both the statement x and the witness w. It then evaluates the signatures using a function g which, on input (x, w), outputs x if and only if w is a valid witness of x, and outputs the evaluated signature as the proof. We remark that, although the construction is quite different, the idea of using homomorphic signatures to construct proof systems is also considered by Libert *et al.* [LPJY14].

Formally, let $\mathcal{HS} = (\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$ be a 1-hop (1-)HS scheme. Let g be a function such that $g(x, w) = x$ if $R(x, w) = 1$, $\perp$ otherwise. Figure 4 shows our SNARG system $\Pi = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Vf})$ for NP language L with relation R.

Theorem 10. *Suppose $\mathcal{HS}$ is correct, then Π is complete.*

Theorem 11. *Suppose $\mathcal{HS}$ is cEUF-CMA-secure, then Π is sound.*

Theorem 12. *Suppose $\mathcal{HS}$ is weakly context-hiding, then Π is zero-knowledge.*

Theorem 13. *Suppose $\mathcal{HS}$ is succinct then Π is succinct.*

Note that if the underlying M-HS scheme is secure in the standard model (without a common reference string), *i.e.*, $\mathsf{pp} = \lambda$, then the above construction would yield a ZN-SNARG in the standard model, which is impossible. Therefore, we can also rule out the possibility of constructing standard model M-HS schemes which are unforgeable under corruption. Interestingly, the only existing M-HS scheme [FMNP16] is unforgeable (without corruption) in the standard model.

4 Extensions and Special Cases of Multi-Key HS

4.1 Multi-Key Hierarchical Homomorphic Signatures (M-HiHS)

Imagine the following scenario: Alice fills in the first two entries of a form, and passes the form to Bob. The latter fills the next two entries, and passes to Charlie, and so on. At the end, an evaluator collects a number of forms filled by multiple groups of people and processes them in batch. To support such functionality using some variant of M-HS, we require additionally that the scheme is delegatable. Roughly, a delegator is able to derive a signing key which is able to sign any set of messages that contains a subset specified by the delegator. test We define an extension of M-HS, namely, multi-key hierarchical homomorphic signatures (M-HiHS).

M-HiHS allows a signer to specify a set of messages M, and delegate the signing power of all supersets of M to another signer[1]. More concretely, let pk and sk be the public and secret key of the delegator. To delegate the signing power over a set M to another signer with public key pk', the delegator derives a delegated key dk'. A delegatee can use this delegated key to sign any set $M' \supseteq M$ to create a complete signature, so

[1] Similar ideas have been studied by Kiltz *et al.* [KMPR05] and Bethencourt *et al.* [BBW07] in the context of append only signatures.

that any public evaluator can derive a signature of $g(M')$ signed under the delegation chain $(\mathsf{pk}, \mathsf{pk}')$. This generalizes proxy signatures in the literature where M serves as a "smart warrant" with respect to function g.

More generally, a delegatee can further delegate its signing power to form a longer delegation chain. Lastly, similar to M-HS, the public evaluator can combine signatures signed under different chains.

History of Messages We first introduce a concept of "history". Consider a message which is obtained by evaluating a function g over multiple sets of messages. Each messages set is signed by a different delegation chain, within which messages are delegated and signed under different public keys and tags. The arrangement of these public keys and tags forms the *history of the message* Δ_m. More formally, we define Δ_m as a tree with the following properties:

1. Each node corresponds to either a delegator, a signer, or an evaluator.
2. Each node corresponding to a delegator contains the delegatee public key and the delegated tag. Its parent node corresponds to the delegatee.
3. Similarly, each node corresponding to a signer contains an empty string ϵ, indicating that the delegation chain is finalized, and the signed tag.
4. Each node corresponding to an evaluator is an empty parent node of the nodes corresponding to the signers of the input signatures.

Each leaf node of the tree thus corresponds to the first delegator in a delegation chain. The nodes corresponding to a delegation chain forms a linked list, and the linked lists are connected on top by a tree formed from the nodes corresponding to the evaluator. The *history of a set of messages*, denoted by Δ_M, is defined as a set of histories Δ_m for all m in M. When the context is clear, we drop the subscript and write Δ for conciseness.

Given a hash function $H : \{0,1\}^* \to \{0,1\}^\lambda$, one can compute the digest of the history, denoted by $H(\Delta)$, by evaluating the tree as follows: Consider an algorithm which rewrites each node in the history tree by intermediate values, and outputs the value written in the root node. The algorithm starts from the leaf nodes, which remain unchanged. For each parent node, if it is not a node corresponding to an evaluator, then it stores a public key pk (or an empty string ϵ) and a tag τ, and has only one children node with intermediate hash value h. In this case, the algorithm computes and stores $H(h\|pk\|\tau)$ in the node. Otherwise, the node is corresponding to an evaluator, and has children nodes with intermediate hash values $h_1, \ldots, h_K$. The algorithm thus stores $H(h_1, \ldots, h_K)$ in the node. When the algorithm reaches the root node, it outputs the hash value stored in it.

Definitions

Syntax. An M-HIHS scheme consists of the PPT algorithms $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Del}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$ defined as follows:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$ are the same as those of M-HS.
- $\mathsf{dk}' \leftarrow \mathsf{Del}(\mathsf{sk}, \mathsf{dk}, \mathsf{pk}', \tau, M', \Delta)$ is a new delegation algorithm which inputs the secret key sk of the delegator, an *optional* delegated key dk for some delegation chain ending at pk for the messages M, the public key pk' of the new delegatee, a tag $\tau \in \{0,1\}^*$, a set of messages $M' \in \mathcal{M}^*$ such that $M' \supseteq M$, and the history Δ of the messages M. It outputs a new delegated key dk'.
- $\Sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{dk}, \tau, M', \Delta)$ is the same as that of M-HS, except that it takes as input an *optional* delegated key dk for some delegation chain ending at pk for the messages M, and the history Δ of the messages M. It outputs the finalized signatures Σ on $M' \in \mathcal{M}^*$ such that $M' \supseteq M$.
- $b \leftarrow \mathsf{Vf}(g, \mathsf{PK}, \mathcal{T}, m, \sigma, \Delta)$ is the same as that of M-HS except that it additionally takes as input the history Δ of the message m.
- $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k, \Delta_k)_{k=1}^K)$ is the same as that of M-HS except that it additionally takes as input the history Δ_k of each set of messages M_k.

If the delegation chain length is limited to 1, all delegation chains become a singleton containing only the public key of the root delegator. The M-HiHS scheme thus becomes an M-HS scheme. The correctness, unforgeability, and context-hiding property are defined similarly to those of M-HS. A major difference is that, in the unforgeability game, the adversary is in addition allowed to query a delegation oracle, and the condition of valid forgery is adjusted accordingly to avoid trivial attacks.

Correctness. For any $\mathsf{pp} \in \mathsf{Setup}(1^\lambda)$, any $K, D_k = \mathsf{poly}(\lambda)$, $k \in [K]$, and $d_k \in [D_k]$, any $(\mathsf{pk}_{k,d_k}, \mathsf{sk}_{k,d_k}) \in \mathsf{KGen}(\mathsf{pp})$, any $M_{k,d_k} \in \mathcal{M}^*$, any $\tau_k \in \{0,1\}^*$, and any $g, h_{k,j} \in \mathcal{G}$ with appropriate dimensions, the following is true:

- (Delegation and Signing.) For any K delegation chains of lengths D_k, *i.e.*,

$$\mathsf{dk}_{k,d_k} \leftarrow \mathsf{Del}(\mathsf{sk}_{k,d_k}, \mathsf{dk}_{k,d_{k-1}}, \mathsf{pk}_{k,d_{k+1}}, \tau_{k,d_k}, M_{k,d_k}, \Delta_{k,d_{k-1}}),$$

  and any K finalizing signatures, *i.e.*,

$$\Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_{k,D_k}, \mathsf{dk}_{k,D_k-1}, \tau_{k,D_k}, M_{k,D_k}, \Delta_{k,D_k-1}),$$

  the verification of any individual signature passes, *i.e.*,

$$\mathsf{Vf}(p_{k,j}, \mathsf{PK}_k, \mathcal{T}_k, m_{k,j}, \sigma_{k,j}, \Delta_{k,j}) = 1$$

  for each $\sigma_{k,j} \in \Sigma_k$ and $m_{k,j} \in M_k$
- (Evaluation.) For any Σ_k, if $\mathsf{Vf}(h_{k,j}, \mathsf{PK}_k, \mathcal{T}_k, m_{k,j}, \sigma_{k,j}, \Delta_k) = 1$ for all $\sigma_{k,j} \in \Sigma_k$ and $m_{k,j} \in M_k$, $\mathsf{Hop}(h_{k,j}) \leq N-1$, and $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k, \Delta_k)_{k=1}^K)$, then $\mathsf{Vf}(g \circ \bar{h}, \mathsf{PK}, \mathcal{T}, g(M_1, \ldots, M_K), \sigma, \Delta) = 1$

Unforgeability. Consider the following security game cEUF-CMA between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

- The challenger $\mathcal{C}$ runs $\mathsf{pp} \in \mathsf{Setup}(1^\lambda)$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$, and gives pp, pk to $\mathcal{A}$.
- The adversary $\mathcal{A}$ adaptively issues a polynomial number of delegation queries and signing queries. It also decides whether to issue a corruption query.
  - Delegation queries: In each delegation query q_d, $\mathcal{A}$ chooses an *optional* delegated key dk_{q_d}, for some delegation chain ending at pk for the messages M_{q_d}, the public key pk'_{q_d} of the new delegatee, a fresh tag $\tau_{q_d} \in \{0,1\}^*$, a set of messages $M'_{q_d} \in \mathcal{M}^*$ such that $M'_{q_d} \supseteq M_{q_d}$, and the history Δ_{q_d} of the messages M_{q_d}. It outputs a new delegated key $\mathsf{dk}'_{q_d} \leftarrow \mathsf{Del}(\mathsf{sk}, \mathsf{dk}_{q_d}, \mathsf{pk}'_{q_d}, \tau_{q_d}, M'_{q_d}, \Delta_{q_d})$.
  - Signing queries: In each signing query q_s, $\mathcal{A}$ chooses an *optional* delegated key dk_{q_s}, for some delegation chain ending at pk for the messages M_{q_s}, a fresh tag $\tau_{q_s} \in \{0,1\}^*$, a set of messages $M'_{q_s} \in \mathcal{M}^*$ such that $M'_{q_s} \supseteq M_{q_s}$, and the history Δ_{q_s} of the messages M_{q_s}. It outputs a signature $\Sigma_{q_s} \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{dk}_{q_s}, \tau_{q_s}, M'_{q_s}, \Delta_{q_s})$.
  - Corruption query: The challenger $\mathcal{C}$ responds with sk.
- The adversary $\mathcal{A}$ outputs a public key PK^*, a tag $\mathcal{T}^* \in \{0,1\}^*$, a signature σ^*, a message $m^* \in \mathcal{M}$, a function $g^* \in \mathcal{G}$, and a history Δ^*. It wins the game if $\mathsf{Vf}(g^*, \mathsf{PK}^*, \mathcal{T}^*, m^*, \sigma^*, \Delta^*) = 1$, and $(g^*, \mathsf{PK}^*, \mathcal{T}^*, m^*, \sigma^*, \Delta^*)$ is either a forgery such that
  - *Type-I*: the signer is not corrupted, the signer is a delegator and $\forall\ (\tau_{q_d}, \mathsf{pk}^*) \in \Delta^*$, $\mathsf{pk}^* \neq \mathsf{pk}_{q_d}$
  - *Type-II*: the signer is not corrupted, $\mathsf{pk} \in \mathsf{PK}^*$ and
    - $\forall\ \tau_{q_d} \in \mathcal{T}^*$, $m^* \notin g^*(\cdots, M_{q_d}, \cdots)$
    - $\forall\ \tau_{q_s} \in \mathcal{T}^*$, $m^* \notin g^*(\cdots, M_{q_s}, \cdots)$
  - *Type-III*: $m^* \notin R_{g^*}$.

We say that the scheme is unforgeable under corruption (cEUF-CMA-secure) if, for all PPT adversaries $\mathcal{A}$, we have $\Pr\{\mathsf{cEUF\text{-}CMA}_{\mathcal{HS},\mathcal{A}} = 1\} \leq \mathsf{negl}(\lambda)$. We say that the scheme is unforgeable (EUF-CMA-secure) if the adversary $\mathcal{A}$ is not allowed to issue corrupt query in the game. We also define the K-cEUF-CMA in a similar way as in M-HS.

Theorem 14. *cEUF-CMA-security and K-cEUF-CMA-security are equivalent.*

Context-Hiding. There exist a simulator $\mathcal{S} = (\mathcal{S}^{\mathsf{Setup}}, \mathcal{S}^{\mathsf{Sig}})$ such that, for all $K, D_k = \mathsf{poly}(\lambda)$, $k \in [K]$, $d_k \in [D_k]$, τ_{k,d_k}, M_{k,d_k}, and $g \in \mathcal{G}$, it holds that for any PPT adversaries $\mathcal{A}$,

$$\left| \Pr \left[\begin{array}{c} \mathcal{A}(g, (\mathsf{pk}_{k,d_k}, \mathsf{sk}_{k,d_k}, \tau_{k,d_k}, M_{k,d_k}, \Sigma_{k,d_k})_{k=1,d_k=1}^{K,D_k}, \sigma, \Delta_{k,d_k}) \to 1 : \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{pk}_{k,d_k}, \mathsf{sk}_{k,d_k}) \leftarrow \mathsf{KGen}(\mathsf{pp}) \\ \mathsf{dk}_{k,d_k} \leftarrow \mathsf{Del}(\mathsf{sk}_{k,d_k}, \mathsf{dk}_{k,d_k-1}, \mathsf{pk}_{k,d_k+1}, \tau_{k,d_k}, M_{k,d_k}, \Delta_{k,d_k-1}) \\ \Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_{k,D_k}, \mathsf{dk}_{k,D_k-1}, \tau_{k,D_k}, M_{k,D_k}, \Delta_{k,D_k-1}) \\ \sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k, \Delta_k)_{k=1}^K) \end{array} \right] - \right.$$

$$\left. \Pr \left[\begin{array}{c} \mathcal{A}(g, (\mathsf{pk}_{k,d_k}, \mathsf{sk}_{k,d_k}, \tau_{k,d_k}, M_{k,d_k}, \Sigma_{k,d_k})_{k=1,d_k=1}^{K,D_k}, \sigma, \Delta_{k,d_k}) \to 1 : \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{pk}_{k,d_k}, \mathsf{sk}_{k,d_k}) \leftarrow \mathsf{KGen}(\mathsf{pp}) \\ \mathsf{dk}_{k,d_k} \leftarrow \mathsf{Del}(\mathsf{sk}_{k,d_k}, \mathsf{dk}_{k,d_k-1}, \mathsf{pk}_{k,d_k+1}, \tau_{k,d_k}, M_{k,d_k}, \Delta_{k,d_k-1}) \\ \Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_{k,D_k}, \mathsf{dk}_{k,D_k-1}, \tau_{k,D_k}, M_{k,D_k}, \Delta_{k,D_k-1}) \\ \sigma \leftarrow \mathcal{S}^{\mathsf{Sig}}(\mathsf{td}, g, \mathsf{PK}, \mathcal{T}, g(M_1, \ldots, M_K), \Delta) \end{array} \right] \right| = \mathsf{negl}(\lambda) .$$

The scheme is weakly context hiding if the above holds.

Succinctness. There exist polynomial $s(\cdot)$ such that for every $\lambda \in \mathbb{N}$, $K, D_k = \mathsf{poly}(\lambda)$, $k \in [K]$, $d_k \in [D_k]$, $g \in \mathcal{G}$, $\tau_{k,d_k} \in \{0,1\}^*$, $M_{k,d_k} \in \mathcal{M}^*$, it holds with probability 1 over $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$; $(\mathsf{pk}_{k,d_k}, \mathsf{sk}_{k,d_k}) \leftarrow \mathsf{KGen}(\mathsf{pp})$; $\mathsf{dk}_{k,d_k} \leftarrow \mathsf{Del}(\mathsf{sk}_{k,d_k}, \mathsf{dk}_{k,d_k-1}, \mathsf{pk}_{k,d_k+1}, \tau_{k,d_k}, M_{k,d_k}, \Delta_{k,d_k-1})$; $\Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_{k,D_k}, \mathsf{dk}_{k,D_k-1}, \tau_{k,D_k}, M_{k,D_k}, \Delta_{k,D_k-1})$; that the resulting signature $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k, \Delta_k)_{k=1}^K)$ on $m = g(M_1, \ldots, M_K)$ has size $|\sigma| \le s(\lambda, |m|)$. In particular, the signature size is independent of the sizes $|M_k|$ of the inputs to the function, and the size $|g|$ of a description of the function g. We also require that the signature size is independent of the delegation chain length.

Construction We next show that M-HiHS can be constructed from M-HS and collision-resistant hash functions (CRHF), which implies the equivalence of M-HS and M-HiHS up to the existence of CRHF. The idea is to use the delegation chain as part of the tag of a signature. To extend the delegation chain PK to pk' with the set of messages $M' \supseteq M$, the delegator simply signs M under a random tag using M-HS, and signs the hash value of all previous tags together with pk' by ordinary signatures. Likewise, to finalize the delegation chain PK with a set of messages M and tag τ, the delegatee signs M under the tag τ using M-HS, and signs all previous tags with ordinary signatures.

Formally, let $H : \{0,1\}^* \to \{0,1\}^\lambda$ be a collision-resistant hash function, and $\mathcal{HS}.(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$ be an $(N+1)$-hop M-HS scheme. We construct an N-hop M-HiHS scheme $\mathcal{HS}'$ as shown in Figure 5.

The correctness and context-hiding property follow straightforwardly from those of $\mathcal{HS}$. The proofs of the two theorems below can be found in the supplementary material.

Theorem 15. *Suppose $\mathcal{HS}$ is unforgeable under corruption and H is collision-resistant, then $\mathcal{HS}'$ is unforgeable under corruption.*

Theorem 16. *Suppose $\mathcal{HS}$ is succinct, then $\mathcal{HS}'$ is succinct.*

4.2 Multi-Key Key-Message Homomorphic Signatures (M-KMHS)

Another extension of M-HS is the multi-key key-message homomorphic signatures (M-KMHS). In M-KMHS, homomorphic operations are not only with respect to the message space, but also the key space. Computation over randomly generated public keys is however not always meaningful. To make computation over keys more meaningful, we adopt the terminology of Boneh *et al.* [BGG+14] where the public keys are actually attributes, whose corresponding secret keys are generated from their respective authorities. More concretely, suppose an authority authorizes the set of attributes X and X' of two signers respectively. The signer with attributes X then issues a signature on a set of messages M, and the signer attributed X' also signs M'. Given the signatures, a public evaluator can then derive a signature on the message $g_2((X, M), (X', M'))$ attributed to $g_1(X, X')$ for any functions g_1 and g_2.

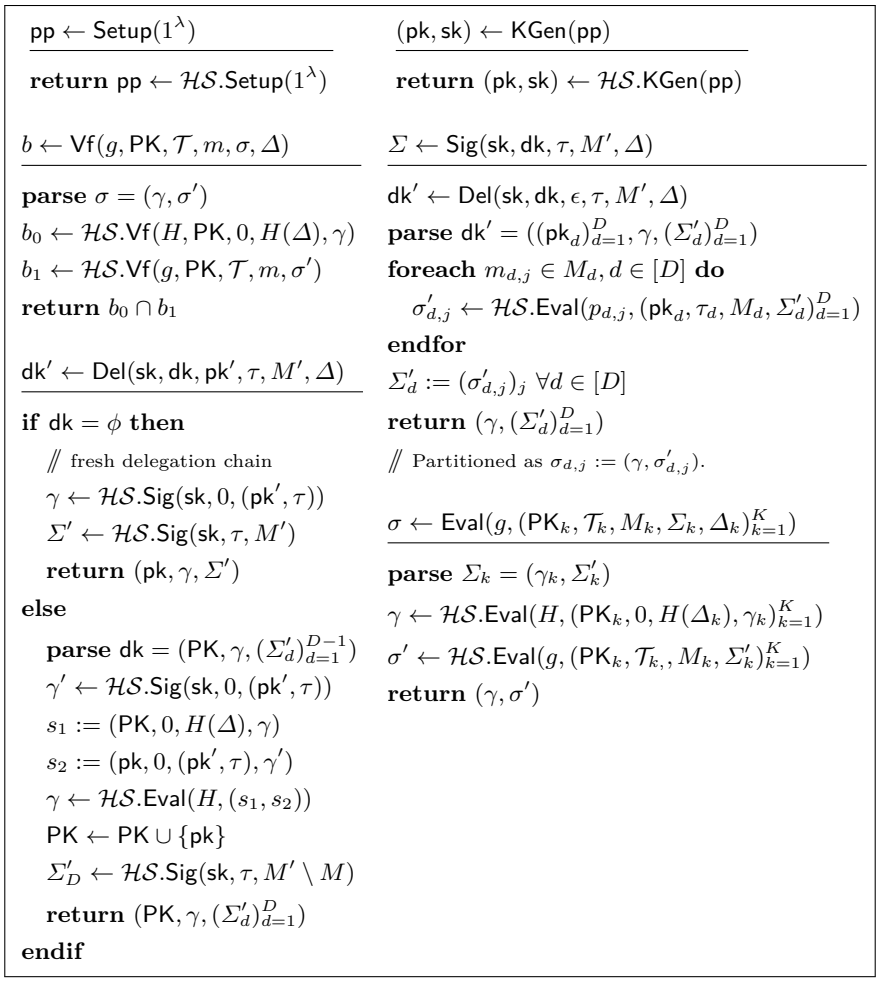

Fig. 5. Construction of M-HiHS from M-HS

Definitions

Syntax. A key-message homomorphic signature scheme consists of the following six PPT algorithms (Setup, $\mathsf{KGen}_{\mathrm{Auth}}, \mathsf{KGen}_{\mathrm{Sig}}, \mathsf{Auth}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval}$) defined as follows:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp})$ are the same as those of M-HS. The public parameter additionally defines the attribute space $\mathcal{X}$, and class of admissible functions $\mathcal{G} \subseteq \{g | g : (\mathcal{X}, \mathcal{M})^* \to (\mathcal{X}, \mathcal{M})\}$.
- $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp})$ is a new key generation algorithm for generating an authority public key apk and an authority secret key ask.
- $\Gamma \leftarrow \mathsf{Auth}(\mathsf{ask}, \mathsf{pk}, X)$ is a new authentication algorithm which inputs an authority secret key ask, the public key of the signer pk, and a set of attributes X. It outputs a set of credentials Γ.
- $\Sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, (\Gamma_k, X_k)_{k=1}^{K}, \tau, M)$ additionally takes attributes X_k and the corresponding credentials Γ_k signed by the K authorities.
- $b \leftarrow \mathsf{Vf}(g, \mathcal{APK}, \mathsf{PK}, \mathcal{T}, x, m, \sigma)$ takes as input an additional authority public key $\mathcal{APK}$ and an attribute $x \in \mathcal{X}$.
- $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{apk}_k, \mathsf{pk}_k, \tau_k, X_k, M_k, \Sigma_k)_{k=1}^{K})$ additionally evaluates the attributes. Concretely, for each input tuple corresponding to the K signers, it inputs an additional set of authority public keys $\mathcal{APK}_k$ and a set of attributes X_k. It outputs a new signature σ.

If the attribute space is defined as $\mathcal{X} = \{0, \perp\}$, and the admissible functions are of the form $g((x_1, m_1), \ldots, (x_K, m_K)) = (x, g_2(m_1, \ldots, m_K))$ if $x = x_1 = \ldots = x_K$, and $(\perp, \perp)$ otherwise, then the M-KMHS scheme essentially becomes an M-HS scheme. The correctness, unforgeability, and context-hiding property are defined similarly to those of M-HS. A major difference is that, in the unforgeability game, the adversary can also corrupt the authorities and query an authentication oracle, and the condition of valid forgery is adjusted accordingly to avoid trivial attacks. Furthermore, the context-hiding property now also hides the attributes input of the evaluation algorithm.

Correctness. A simple way to define the correctness of M-KMHS is to view $\mathsf{KGen}_{\mathrm{Auth}}$ and Auth as the key generation and signing algorithms of a separate M-HS scheme respectively. The evaluation correctness of M-KMHS requires that the evaluation correctness of the two separate M-HS holds simultaneously. Concretely, for any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, any $K, L = \mathsf{poly}(\lambda)$, $k \in [K]$, $\ell \in [L]$, any $(\mathsf{apk}_\ell, \mathsf{ask}_\ell) \in \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp})$ any $(\mathsf{pk}_k, \mathsf{sk}_k) \in \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp})$, any $X_{k,\ell} \in \mathcal{X}^*$, any $M_k \in \mathcal{M}^*$, any $\tau_k \in \{0,1\}^*$, any $h_{k,i,j}, g \in \mathcal{G}$ with appropriate dimensions, the following correctness requirements hold.

- (Authentication and Signing.) If $\Gamma_{k,\ell} \leftarrow \mathsf{Auth}(\mathsf{ask}_\ell, \mathsf{pk}_k, X_{k,\ell})$ and $\Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}_k, (X_{k,\ell}, \Gamma_{k,\ell})_{\ell=1}^L, \tau_k, M_k)$, then for all $(x_{k,\ell,i}, m_{k,j}) \in X_{k,\ell} \times M_k$ and $\sigma_{k,\ell,i,j} \in \Sigma_k$,

$$\mathsf{Vf}(\mathsf{id}_2, \mathcal{APK}_k \cup \mathsf{pk}_k, \mathsf{pk}_k \cup \tau_k, x_{k,\ell,i}, m_{k,j}, \sigma_{k,\ell,i,j}) = 1,$$

  where $\mathcal{APK}_k = \cup_{\ell=1}^L \mathsf{apk}_\ell$
- (Evaluation.) For any Σ_k, if for all $(x_{k,i}, m_{k,j}) \in X_k \times M_k$, $\sigma_{k,i,j} \in \Sigma_k$ and $\mathsf{Hop}(h_{k,i,j}) \leq N - 1$,

$$\mathsf{Vf}(h_{k,i,j}, \mathcal{APK}_k, \mathsf{PK}_k, \mathcal{T}_k, x_{k,i}, m_{k,j}, \sigma_{k,i,j}) = 1;$$

  and $\sigma \leftarrow \mathsf{Eval}(g, (\mathcal{APK}_k, \mathsf{PK}_k, \mathcal{T}_k, X_k, M_k, \Sigma_k)_{k=1}^K)$, it holds that

$$\mathsf{Vf}(g \circ \bar{h}, \mathcal{APK}, \mathsf{PK}, \mathcal{T}, g_1(X_1, \ldots, X_K), g_2((X_1, M_1), \ldots, (X_K, M_k)), \sigma) = 1.$$

Unforgeability. Consider the $\mathsf{cEUF\text{-}CMA}$ game played by an adversary below.

- The challenger $\mathcal{C}$ runs $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp})$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp})$, and gives apk and pk to $\mathcal{A}$.
- The adversary $\mathcal{A}$ makes four types of queries:
  - $\mathcal{O}_{\mathrm{CorrAuth}}()$ The challenger responds with the authority secret key ask.
  - $\mathcal{O}_{\mathrm{CorrSig}}()$ The challenger responds with the signer secret key sk.
  - $\mathcal{O}_{\mathsf{Auth}}(\mathsf{pk}_{q_a}, X_{q_a})$ The challenger returns $\Gamma_{q_a} \leftarrow \mathsf{Auth}(\mathsf{ask}, \mathsf{pk}_{q_a}, X_{q_a})$.
  - $\mathcal{O}_{\mathsf{Sig}}((\Gamma_{q_s,k}, X_{q_s,k})_{k=1}^K, \tau_{q_s}, M_{q_s})$ The challenger returns $\Sigma_{q_s} \leftarrow \mathsf{Sig}(\mathsf{sk}, (\Gamma_{q_s,k}, X_{q_s,k})_{k=1}^K, \tau_{q_s}, M_{q_s})$.
- Eventually the adversary $\mathcal{A}$ outputs a functions $g^* \in \mathcal{G}$, an authority public key $\mathcal{APK}^*$, a signer public key PK^*, a tag $\mathcal{T}^*$, an attribute x^*, a message m^*, and a signature σ^*. It wins the game if $\mathsf{Vf}(g^*, \mathcal{APK}^*, \mathsf{PK}^*, \mathcal{T}^*, x^*, m^*, \sigma^*) = 1$ and the forgery is either of:
  - *Type-I*: the authority is not corrupted, $\mathsf{apk} \in \mathcal{APK}^*$ and $\forall\ \mathsf{pk}_{q_a} \in \mathsf{PK}^*$, $x^* \notin g_1^*(\cdots, X_{q_a}, \cdots)$
  - *Type-II*: the signer is not corrupted, $\mathsf{pk} \in \mathsf{PK}^*$ and $\forall\ \tau_{q_s} \in \mathcal{T}^*$, $m^* \notin g_2^*(\cdots, M_{q_s}, \cdots)$
  - *Type-III*: $(x^*, m^*) \notin R_{g^*}$

We require that for all PPT adversaries $\mathcal{A}$, we have $\Pr\{\mathcal{A} \text{ wins}\} \leq \mathsf{negl}(\lambda)$ in the above game.

We say that the scheme is unforgeable under corruption (cEUF-CMA-secure) if, for all PPT adversaries $\mathcal{A}$, we have $\Pr\{\mathsf{cEUF\text{-}CMA}_{\mathcal{HS},\mathcal{A}} = 1\} \leq \mathsf{negl}(\lambda)$. We say that the scheme is unforgeable (EUF-CMA-secure) if the adversary $\mathcal{A}$ is not allowed to issue corrupt queries in the game. We also define the L-K-$\mathsf{cEUF\text{-}CMA}$ in a similar way as in M-HS where there are L authorities and K signers.

Theorem 17. *cEUF-CMA-security is equivalent to L-K-cEUF-CMA-security.*

Context-Hiding. There exist a simulator $\mathcal{S} = (\mathcal{S}^{\mathsf{Setup}}, \mathcal{S}^{\mathsf{Sig}})$ such that, for all $L, K = \mathsf{poly}(\lambda)$, $k \in [K]$, $\ell \in [L]$, τ_k, M_k, and $g \in \mathcal{G}$, it holds that for any PPT adversaries $\mathcal{A}$,

$$\left| \Pr \left[\begin{array}{c} \mathcal{A}(g, (\mathsf{apk}_k, \mathsf{pk}_k, \tau_k, X_k, M_k, \Sigma_k)_{k=1}^K, \sigma) \to 1 : \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{apk}_\ell, \mathsf{ask}_\ell) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp}) \\ (\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp}) \\ \Gamma_{\ell,k} \leftarrow \mathsf{Auth}(\mathsf{ask}_\ell, \mathsf{pk}_k, X_{\ell,k}) \\ \Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}, (\Gamma_{\ell,k}, X_{\ell,k})_{\ell=1}^L, \tau_k, M_k) \\ \sigma \leftarrow \mathsf{Eval}(g, (\mathsf{apk}_k, \mathsf{pk}_k, \tau_k, X_k, M_k, \Sigma_k)_{k=1}^K) \end{array} \right] - \right.$$

$$\left. \Pr \left[\begin{array}{c} \mathcal{A}(g, (\mathsf{apk}_k, \mathsf{pk}_k, \tau_k, X_k, M_k, \Sigma_k)_{k=1}^K, \sigma) \to 1 : \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{apk}_\ell, \mathsf{ask}_\ell) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp}) \\ (\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp}) \\ \Gamma_{\ell,k} \leftarrow \mathsf{Auth}(\mathsf{ask}_\ell, \mathsf{pk}_k, X_{\ell,k}) \\ \Sigma_k \leftarrow \mathsf{Sig}(\mathsf{sk}, (\Gamma_{\ell,k}, X_{\ell,k})_{\ell=1}^L, \tau_k, M_k) \\ \sigma \leftarrow \mathcal{S}^{\mathsf{Sig}}(\mathsf{td}, g, \mathcal{APK}, \mathsf{PK}, \mathcal{T}, g_1(X_1, \ldots, X_K), \\ g_2((X_1, M_1), \ldots, (X_K, M_K))) \end{array} \right] \right| = \mathsf{negl}(\lambda) .$$

The scheme is weakly context hiding if the above holds.

Construction In the previous part, we show that M-HiHS can be constructed from M-HS, hence showing the equivalence of their existences. We now construct M-KMHS using M-HS, which in turn shows the equivalence of all three notions. The idea is to issue M-HS signatures on attributes as M-KMHS credentials.

Let $\mathcal{HS}.(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$ be an M-HS scheme. We construct an M-KMHS scheme $\mathcal{HS}'$ as shown in Figure 6. In this construction, we uses projection functions p_1 and p_2 where $p_1(X_1, \cdots, X_k) = (X_1, \cdots, X_k)$ and $p_2((X_1, M_1), \cdots, (X_k, M_k))$ for $k \in \mathsf{poly}(\lambda)$. The correctness, unforgeability, and context-hiding property follow straightforwardly from those of $\mathcal{HS}$.

4.3 Multi-Key Key-Homomorphic Signatures (M-KHS)

Recall from the definition of M-KMHS in Section 4.2, consider a class of functions $g : (\mathcal{X}, \mathcal{M})^* \to (\mathcal{X}, \mathcal{M})$ such that $g((x_1, m_1), \ldots, (x_K, m_K)) = (g_1(x_1, \ldots, x_K), m)$ if $m = m_1 = \ldots = m_K$, and $g((x_1, m_1), \ldots, (x_K, m_K)) = (\perp, \perp)$ otherwise. We thus obtain a multi-key signature scheme which is only homomorphic in the attribute space, which we refer to as multi-key key-homomorphic signature scheme (M-KHS). In such configuration, the construction in Section 4.2 is actually a transformation from M-HS to M-KHS.

Formally, we define an M-KHS scheme as a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{KGen}_{\mathrm{Auth}}, \mathsf{KGen}_{\mathrm{Sig}}, \mathsf{Auth}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$ which are almost identical to those of M-KMHS, except that the class of admissible functions is restricted to $\mathcal{G} \subseteq \{g | g : \mathcal{X}^* \to \mathcal{X}\}$. The correctness requires that fresh signatures must pass verification with respect to the one-dimensional identity function $\mathsf{id} : \mathcal{X} \to \mathcal{X}$.

From the syntax, it is apparent that we can interpret the attribute space of M-KHS as the message space of M-HS, and obtain a construction of M-HS. In particular, we conclude that the existence of ordinary HS and single-authority KHS are equivalent.

4.4 Other Extensions of M-HS

In the previous subsections, we study two extensions of M-HS, namely M-HiHS and M-KMHS, and show that they can both be constructed from M-HS, hence proving the equivalence of their existences. Recall that M-HiHS extends M-HS vertically in the sense that it allows the delegation of signing power down the delegation chains. M-KMHS extends M-HS in another dimension as it introduces additional homomorphism

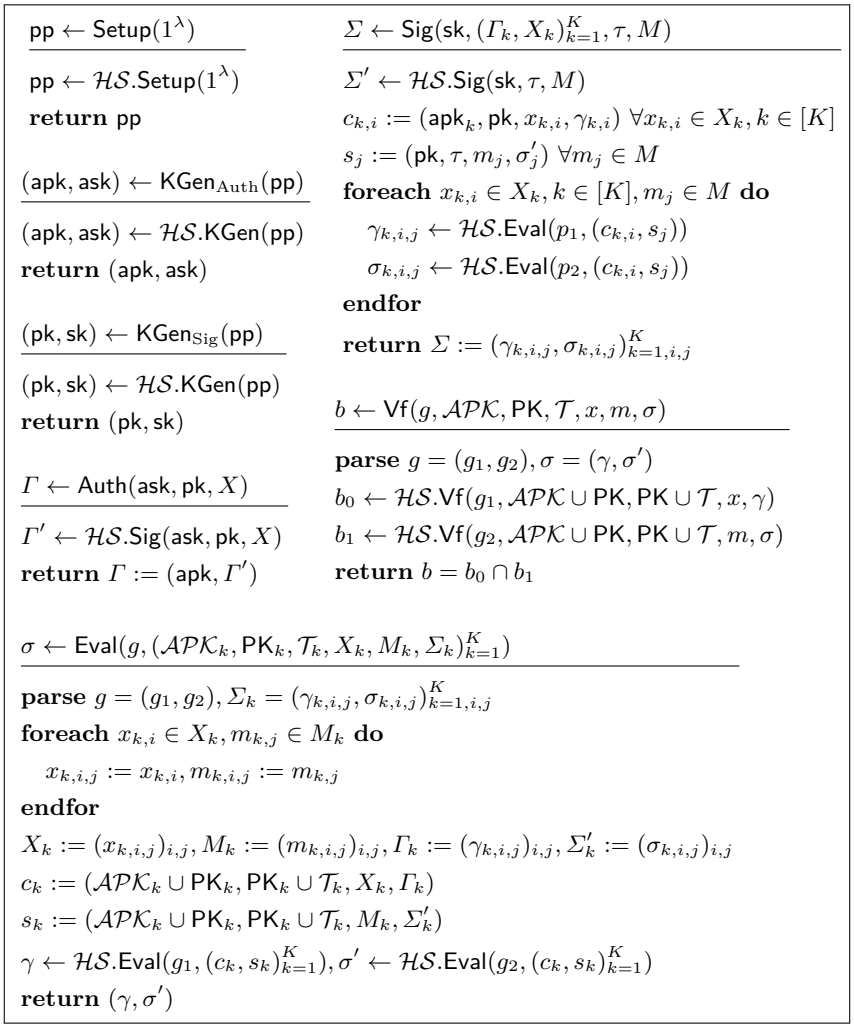

Fig. 6. Construction of M-KMHS from M-HS

to the key space. We note that other extensions, including but not limited to combing the hierarchy of M-HiHS and the key-homomorphism in M-KMHS, can likely be also constructed from M-HS. This illustrates the power of supporting homomorphism in more than one spaces, and the power of corruption-resistance in homomorphic signatures.

5 Decentralized Attribute-based Signatures

Apart from the natural application of allowing delegation of computation on data authenticated by multiple parties, we study the implications of M-HS to other primitives, specifically, decentralized attribute-based signatures (D-ABS) [OT13].

A D-ABS scheme allows multiple authorities to certify different sets of attributes of a signer in a completely distributed manner. After obtaining the certificates from the authorities, the signer can then issue signatures on messages, while at the same time show that its certified attributes satisfy certain access policy. We show how to construct from KHS a D-ABS scheme which supports access policies in the class of admissible functions $\mathcal{F}$ of the KHS scheme.

Due to the tag-based nature of KHS, this scheme can only achieve a weaker notion of anonymity, which we call linkable anonymity. It prevents the adversary from learning any information about the attributes associated with the signatures except those leaked from the access policies. On one hand, linkability is a useful feature to achieve strong accountability. For example, consider a simple membership system where a user can register by issuing a linkable attribute-based signature, so that the server can use the linkable part of the signature as the identity of the user. Indeed there is a branch of literature which incorporates various forms of linkability into signatures or credentials. On the other hand, one can generically transform this linkable scheme to an unlinkable one: Simply replace the signature by a non-interactive witness-indistinguishable (NIWI) proof of the knowledge of the tag in the KHS.

5.1 Definitions

Syntax. An attribute-based signature scheme consists of the PPT algorithms $(\mathsf{Setup}, \mathsf{KGen}_{\mathrm{Auth}}, \mathsf{KGen}_{\mathrm{Sig}}, \mathsf{Auth}, \mathsf{Sig}, \mathsf{Vf})$ defined as follows:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ inputs the security parameter λ and outputs the public parameter pp, which defines the attribute space $\mathcal{X}$ and the message space $\mathcal{M}$.
- $(\mathsf{apk}, \mathsf{ask}) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp})$ inputs the public parameter and outputs an authority public key apk and an authority secret key ask.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp})$ inputs the public parameter and outputs a signer public key pk and a signer secret key sk.
- $\Gamma \leftarrow \mathsf{Auth}(\mathsf{ask}, \mathsf{pk}, X)$ inputs an authority secret key ask, a signer public key pk, and a set of attributes $X \in \mathcal{X}^*$. It outputs a set of credentials Γ corresponding to the attributes X.
- $\sigma \leftarrow \mathsf{Sig}(f, \mathsf{sk}, (\Gamma_k, X_k)_{k=1}^K, m)$ inputs an access policy $f \in \mathcal{F}$, the signer secret key sk, a set of credentials Γ_k for the attributes X_k issued by authority k, and a message m. It outputs a signature σ signing m under the attributes $(X_1, \ldots, X_K)$ and policy f.
- $b \leftarrow \mathsf{Vf}(f, \mathcal{APK}, m, \sigma)$ inputs an access policy f, the authority public key apk_k of each of the authorities, a message m, and a signature σ. It outputs 1 if the signature is valid, 0 otherwise.

Correctness. For any $\mathsf{pp} \in \mathsf{Setup}(1^\lambda)$, any $K = \mathsf{poly}(\lambda)$ and $k \in [K]$, any $(\mathsf{apk}_k, \mathsf{ask}_k) \in \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp})$, any $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp})$, any $X_k \in \mathcal{X}^*$, any $\Gamma_k \in \mathsf{Auth}(\mathsf{ask}_k, \mathsf{pk}, X_k)$, any policy $f \in \mathcal{F}$, and any message $m \in \mathcal{M}$, it holds that if $\sigma \in \mathsf{Sig}(f, \mathsf{sk}, (\Gamma_k, X_k)_{k=1}^K, m)$, then $\mathsf{Vf}(f, \mathcal{APK}, m, \sigma) = f(X_1, \ldots, X_k)$.

Unforgeability. Consider the game below between an adversary and a challenger.

- Let $\mathfrak{K} = \mathsf{poly}(\lambda)$ be the maximum number of authorities in the system.

– Initialize an empty dictionary D.
– The challenger $\mathcal{C}$ runs $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $\{(\mathsf{apk}_k, \mathsf{ask}_k) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp})\}_{k=1}^{\mathfrak{K}}$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp})$, and gives $(\mathsf{apk}_k)_{k=1}^{\mathfrak{K}}$ and pk to $\mathcal{A}$.
– The adversary $\mathcal{A}$ makes four types of queries:
  • $\mathcal{O}_{\mathrm{Corr}}(k)$: The challenger responds with the authority secret key ask_k. Without loss of generality, suppose authority $1, \ldots, K$ are corrupt.
  • $\mathcal{O}_{\mathrm{Sig}}()$: The challenger responds with the signer secret key sk.
  • $\mathcal{O}_{\mathsf{Auth}}(k, X_k)$: If there exists a key for the entry k in the dictionary D, returns $\perp$. Otherwise, the challenger returns $\Gamma_k \leftarrow \mathsf{Auth}(\mathsf{ask}_k, \mathsf{pk}, X_k)$ and adds (Γ_k, X_k) to the entry k in the dictionary D.
  • $\mathcal{O}_{\mathsf{Sig}}(S, f, m)$: S is a set of authorities chosen by $\mathcal{A}$. If there exists $k \in S$ such that $D[k] = \perp$, the challenger returns $\perp$. Otherwise, for each $k \in S$, the challenger retrieves (Γ_k, X_k) from the entry k in the dictionary D. It returns $\sigma \in \mathsf{Sig}(f, \mathsf{sk}, (\Gamma_k, X_k)_{k \in S}, m)$.
– Eventually $\mathcal{A}$ outputs a policy $f^* \in \mathcal{F}$, a set of authorities S^*, a message m^*, and a signature σ^*. Let $\mathcal{APK}^* = \cup_{k \in S^*} \mathsf{apk}_k$. It wins the game if the following holds:
  • $\mathsf{Vf}(f^*, \mathcal{APK}^*, m^*, \sigma^*) = 1$, and
  • $\mathcal{APK}^* \subset \cup_{k=1}^{\mathfrak{K}} \mathsf{apk}_k$, and
  • (S^*, f^*, m^*) was not queried to the sign oracle before, and
  • $1 \notin f^*(\cdot, \ldots, \cdot, X_{K+1}, \ldots, X_{\mathfrak{K}})$ for all X_k queried to the $\mathcal{O}_{\mathsf{Auth}}$ oracle, where $k = K+1, \ldots, \mathfrak{K}$, or the signer is not corrupted.

We require that for all PPT adversaries $\mathcal{A}$, $\Pr\{\mathcal{A} \text{ wins}\} \leq \mathsf{negl}(\lambda)$.

Linkable Anonymity. We require that there exist a simulator $\mathcal{S} = (\mathcal{S}^{\mathsf{Setup}}, \mathcal{S}^{\mathsf{Sig}})$ such that, for all $K = \mathsf{poly}(\lambda)$, $k \in [K]$, m, X_k, and g, it holds that for any PPT adversaries $\mathcal{A}$,

$$\left| \Pr \left[\begin{array}{c} \mathcal{A}(\mathsf{pk}, \mathsf{sk}, (\mathsf{apk}_k, \mathsf{ask}_k, \mathsf{sk}_{X_k}, X_k)_{k=1}^K, \sigma, f, m) \to 1 : \\ \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{apk}_k, \mathsf{ask}_k) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp}) \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp}) \\ \Gamma_k \leftarrow \mathsf{Auth}(\mathsf{ask}_k, \mathsf{pk}, X_k) \\ \sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, (\Gamma_k, X_k)_{k=1}^K, f, m) \end{array} \right] - \Pr \left[\begin{array}{c} \mathcal{A}(\mathsf{pk}, \mathsf{sk}, (\mathsf{apk}_k, \mathsf{ask}_k, \mathsf{sk}_{X_k}, X_k)_{k=1}^K, \sigma, f, m) \to 1 : \\ (\mathsf{pp}, \mathsf{td}) \leftarrow \mathcal{S}^{\mathsf{Setup}}(1^\lambda) \\ (\mathsf{apk}_k, \mathsf{ask}_k) \leftarrow \mathsf{KGen}_{\mathrm{Auth}}(\mathsf{pp}) \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{Sig}}(\mathsf{pp}) \\ \Gamma_k \leftarrow \mathsf{Auth}(\mathsf{ask}_k, \mathsf{pk}, X_k) \\ \sigma \leftarrow \mathcal{S}^{\mathsf{Sig}}(\mathsf{td}, f, \mathsf{pk}, \mathcal{APK}, f(X_1, \ldots, X_K), m) \end{array} \right] \right| = \mathsf{negl}(\lambda) .$$

In particular, anonymity guarantees that only $f(X_1, \ldots, X_K)$ is revealed but not the individual values. Signates from thkable in the sense that the public key pk is revealed.

5.2 Construction

From an M-KHS scheme $\mathcal{HS}.(\mathsf{Setup}, \mathsf{KGen}_{\mathrm{Auth}}, \mathsf{KGen}_{\mathrm{Sig}}, \mathsf{Auth}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Eval})$, we get a generic construction of decentralized attribute-based signatures (D-ABS) immediately. Figure 7 presents this construction.

The correctness of ABS follows from the correctness of $\mathcal{HS}$ directly. Suppose $\mathcal{HS}$ is unforgeable, then ABS is unforgeable. Finally, suppose $\mathcal{HS}$ is context-hiding, then ABS is linkably anonymous.

Drawing the connections from Section 4, it is easy to see that the above D-ABS scheme can be extended to support homomorphism in the message space (by M-KMHS). It can also be extended (similar to M-HiHS) so that there is a hierarchy of authorities certifying multiple layers of attributes.

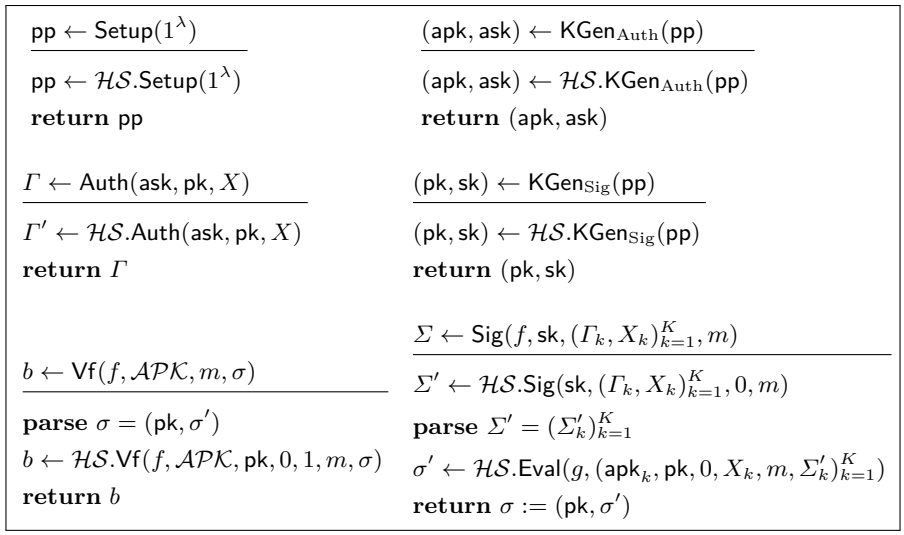

Fig. 7. Construction of D-ABS from M-KHS

5.3 Instantiations

We can instantiate the above generic construction from the multi-key generalization of the fully homomorphic signature scheme by Gorbunov *et al.* [GVW15] (GVW). The following discussions are in order.

- We obtain a D-ABS scheme for general circuits based on lattice assumptions in the standard model.
- In our terminology, the generalized GVW is an M-HS scheme unforgeable for uncorrupted signers. This implies a weak unforgeability of the resulting D-ABS scheme in the sense that the adversary is not allowed to corrupt either the signer or any subset of the authorities.
- In (generalized) GVW, normal verification takes as long as computing the function g. To improve efficiency, part of the verification can be pre-computed so that the amortized verification cost with the fixed function g can be made constant (as explained in details [GVW15]). This is suitable for our purpose as the access policies of a verifier in D-ABS typically remain relatively stable.
- The transformation to a fully anonymous scheme can be instantiated by the recent proof system for linear congruences proposed by Libert *et al.* [LLM+16].

6 Concluding Remark

The study of homomorphic signatures (HS) is in a single-key setting concentrating on the homomorphism in the message space. In this work, we have introduced the notion of multi-key homomorphic signatures with a strong security property known as unforgeability under corruption.

Despite of having a relatively simple syntax, multi-key HS turns out to be a central-hub of various seemingly more powerful or at least different variants of homomorphic signatures. Specifically, it implies multi-key hierarchical HS, multi-key key-message-HS, and multi-key key-HS. The equivalence of multi-key HS and multi-key key-HS suggests that message-homomorphism and key-homomorphism in signatures are identical in nature. Thus, existing fully homomorphic signature schemes readily give fully key-homomorphic signature schemes. We also show that it further implies attribute-based signatures for general circuits from standard assumptions. It is unknown that whether there exist some flavors of HS which are not covered by multi-key HS.

We have constructed a multi-key HS scheme from ZK-SNARK, and shown that the existence of corruption-resistant HS implies the existence of ZK-SNARG. It will be interesting to design a corruption-resistant HS scheme from different primitives or assumptions.

Acknowledgments

We thank some of the anonymous reviewers of Asiacrypt 2016 for their detailed and helpful comments. We also thank Yvo Desmedt and Daniel Wichs for inspiring discussions.

References

[ABC+12] Jae Hyun Ahn, Dan Boneh, Jan Camenisch, Susan Hohenberger, abhi shelat, and Brent Waters. Computing on authenticated data. In Ronald Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 1–20, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Heidelberg, Germany.

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany.

[ALP12] Nuttapong Attrapadung, Benoît Libert, and Thomas Peters. Computing on authenticated data: New privacy definitions and constructions. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 367–385, Beijing, China, December 2–6, 2012. Springer, Heidelberg, Germany.

[ALP13] Nuttapong Attrapadung, Benoît Libert, and Thomas Peters. Efficient completely context-hiding quotable and linearly homomorphic signatures. In Kurosawa and Hanaoka [KH13], pages 386–404.

[BBW07] John Bethencourt, Dan Boneh, and Brent Waters. Cryptographic methods for storing ballots on a voting machine. In *ISOC Network and Distributed System Security Symposium – NDSS 2007*, San Diego, California, USA, February 28 – March 2, 2007. The Internet Society.

[BDF+14] Michael Backes, Özgür Dagdelen, Marc Fischlin, Sebastian Gajek, Sebastian Meiser, and Dominique Schröder. Operational signature schemes. *IACR Cryptology ePrint Archive*, 2014:820, 2014.

[BF11a] Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. In Paterson [Pat11], pages 149–168.

[BF11b] Dan Boneh and David Mandell Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011: 14th International Conference on Theory and Practice of Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 1–16, Taormina, Italy, March 6–9, 2011. Springer, Heidelberg, Germany.

[BF14] Mihir Bellare and Georg Fuchsbauer. Policy-based signatures. In Krawczyk [Kra14], pages 520–537.

[BFKW09] Dan Boneh, David Freeman, Jonathan Katz, and Brent Waters. Signing a linear subspace: Signature schemes for network coding. In Stanislaw Jarecki and Gene Tsudik, editors, *PKC 2009: 12th International Conference on Theory and Practice of Public Key Cryptography*, volume 5443 of *Lecture Notes in Computer Science*, pages 68–87, Irvine, CA, USA, March 18–20, 2009. Springer, Heidelberg, Germany.

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

[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 Nguyen and Oswald [NO14], pages 533–556.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Krawczyk [Kra14], pages 501–519.

[BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 410–428, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[BMS16] Michael Backes, Sebastian Meiser, and Dominique Schröder. Delegatable functional signatures. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 9614 of *Lecture Notes in Computer Science*, pages 357–386, Taipei, Taiwan, March 6–9, 2016. Springer, Heidelberg, Germany.

[BSW12] Dan Boneh, Gil Segev, and Brent Waters. Targeted malleability: homomorphic encryption for restricted computations. In Shafi Goldwasser, editor, *ITCS 2012: 3rd Innovations in Theoretical Computer Science*, pages 350–366, Cambridge, Massachusetts, USA, January 8–10, 2012. Association for Computing Machinery.

[CFW12] Dario Catalano, Dario Fiore, and Bogdan Warinschi. Efficient network coding signatures in the standard model. In Fischlin et al. [FBM12], pages 680–696.

[CFW14] Dario Catalano, Dario Fiore, and Bogdan Warinschi. Homomorphic signatures with efficient verification for polynomial functions. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 371–389, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

[CM15] Michael Clear and Ciaran McGoldrick. Multi-identity and multi-key leveled FHE from learning with errors. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 630–656, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany.

[DS16] David Derler and Daniel Slamanig. Key-homomorphic signatures and applications to multiparty signatures. Cryptology ePrint Archive, Report 2016/792, 2016. http://eprint.iacr.org/2016/792.

[FBM12] Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors. *PKC 2012: 15th International Conference on Theory and Practice of Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, Darmstadt, Germany, May 21–23, 2012. Springer, Heidelberg, Germany.

[FMNP16] Dario Fiore, Aikaterini Mitrokotsa, Luca Nizzardo, and Elena Pagnin. Multi-key homomorphic authenticators. In *Advances in Cryptology - ASIACRYPT 2016*, pages 499–530, 2016.

[FN16] Dario Fiore and Anca Nitulescu. On the (in)security of SNARKs in the presence of oracles. In *TCC 2016: 13th Theory of Cryptography Conference*, 2016. To appear.

[Fre12] David Mandell Freeman. Improved security for linearly homomorphic signatures: A generic framework. In Fischlin et al. [FBM12], pages 697–714.

[GKKR10] Rosario Gennaro, Jonathan Katz, Hugo Krawczyk, and Tal Rabin. Secure network coding over the integers. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010: 13th International Conference on Theory and Practice of Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 142–160, Paris, France, May 26–28, 2010. Springer, Heidelberg, Germany.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th Annual ACM Symposium on Theory of Computing*, pages 469–477, Portland, OR, USA, June 14–17, 2015. ACM Press.

[KH13] Kaoru Kurosawa and Goichiro Hanaoka, editors. *PKC 2013: 16th International Conference on Theory and Practice of Public Key Cryptography*, volume 7778 of *Lecture Notes in Computer Science*, Nara, Japan, February 26 – March 1, 2013. Springer, Heidelberg, Germany.

[KMPR05] Eike Kiltz, Anton Mityagin, Saurabh Panjwani, and Barath Raghavan. Append-only signatures. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP 2005: 32nd International Colloquium on Automata, Languages and Programming*, volume 3580 of *Lecture Notes in Computer Science*, pages 434–445, Lisbon, Portugal, July 11–15, 2005. Springer, Heidelberg, Germany.

[Kra14] Hugo Krawczyk, editor. *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany.

[Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, October 1979.

[LLM+16] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. Cryptology ePrint Archive, Report 2016/101, 2016.

[LPJY13] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Linearly homomorphic structure-preserving signatures and their applications. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 289–307, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[LPJY14] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive NIZK proofs and CCA2-secure encryption from homomorphic signatures. In Nguyen and Oswald [NO14], pages 514–532.

[LTV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, *44th*

Annual ACM Symposium on Theory of Computing, pages 1219–1234, New York, NY, USA, May 19–22, 2012. ACM Press.

[MW16] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In *Advances in Cryptology - EUROCRYPT 2016*, pages 735–763, 2016.

[NO14] Phong Q. Nguyen and Elisabeth Oswald, editors. *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.

[OT13] Tatsuaki Okamoto and Katsuyuki Takashima. Decentralized attribute-based signatures. In Kurosawa and Hanaoka [KH13], pages 125–142.

[Pat11] Kenneth G. Paterson, editor. *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, Tallinn, Estonia, May 15–19, 2011. Springer, Heidelberg, Germany.

[PS16] Chris Peikert and Sina Shiehian. Multi-key FHE from LWE, revisited. Cryptology ePrint Archive, Report 2016/196, 2016.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press.

A Supplementary Material

A.1 Digital Signatures

Definition 4 (Signatures). *A signature scheme is a tuple of PPT algorithms* $\mathcal{DS}.(\mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf})$ *defined as follows:*

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$*: The key generation algorithm takes as input the security parameter* λ *and generates a key pair* $(\mathsf{pk}, \mathsf{sk})$*.*
- $\sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, m)$*: The signing algorithm takes as input a secret key* sk *and a message* $m \in \{0,1\}^*$*. It outputs a signature* σ*.*
- $b \leftarrow \mathsf{Vf}(\mathsf{pk}, m, \sigma)$*: The verification algorithm takes as input a public key* pk*, a message* m*, and a signature* σ*. It outputs a bit* b*.*

Correctness. The scheme is correct if, for all $\lambda \in \mathbb{N}$*, all key pairs* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$*, all messages* $m \in \{0,1\}^*$*, and all signatures* $\sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, m)$*, it holds that* $\mathsf{Vf}(\mathsf{pk}, m, \sigma) = 1$*.*

Definition 5 (Existential Unforgeability). *A signature scheme* $\mathcal{DS}$ *is* existentially unforgeable under chosen message attacks (EUF-CMA-secure) *if, for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that* $\Pr[\mathsf{EUF\text{-}CMA}_{\mathcal{DS},\mathcal{A}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$ *where* $\mathsf{EUF\text{-}CMA}_{\mathcal{DS},\mathcal{A}}$ *is an experiment defined as follows:*

- *The challenger* $\mathcal{C}$ *generates* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ *and gives* pk *to* $\mathcal{A}$*.*
- *The adversary* $\mathcal{A}$ *is given access to a signing oracle* $\mathcal{O}_{\mathsf{Sig}}(\mathsf{sk}, \cdot)$*.*
- *Eventually,* $\mathcal{A}$ *outputs a forgery* (m^*, σ^*)*.*
- *If* m^* *is not queried to the signing oracle, outputs* $\mathsf{Vf}(\mathsf{pk}, m^*, \sigma^*)$*.*
- *Otherwise, the experiment outputs* 0*.*

A.2 Proofs in the Main Text

Proof (Theorem 1). It is easy to see that K-cEUF-CMA-security implies cEUF-CMA-security. It remains to show that cEUF-CMA-security implies K-cEUF-CMA-security. Suppose there exists an adversary $\mathcal{A}_K$ that wins the K-$\mathsf{cEUF\text{-}CMA}$ game with non-negligible probability. We show how to construct an adversary $\mathcal{A}_1$ that uses $\mathcal{A}_K$ to win the $\mathsf{cEUF\text{-}CMA}$ game with non-negligible probability.

$\mathcal{A}_1$ receives from its challenger $(\mathsf{pp}, \mathsf{pk})$. It sets $\mathsf{pk}_K = \mathsf{pk}$ and generates other public keys honestly by $\{(\mathsf{pk}_k, \mathsf{sk}_k) \leftarrow \mathsf{KGen}(\mathsf{pp})\}_{k=1}^{K-1}$. Then $\mathcal{A}_1$ forwards pp and (in a random order) $(\mathsf{pk}_1, \ldots, \mathsf{pk}_K)$ to $\mathcal{A}_K$.

$\mathcal{A}_K$ makes two types of queries:

- Signing queries q with signer k, tag $\tau_{q,k} \in \{0,1\}^*$ and a set of messages $M_{q,k} \in \mathcal{M}^*$: If $k \neq K$, $\mathcal{A}_1$ signs the messages honestly by $\Sigma_{q,k} \leftarrow \mathsf{Sig}(\mathsf{sk}_k, \tau_{q,k}, M_{q,k})$. Else if $k = K$, $\mathcal{A}_1$ forwards $(\tau_{q,K}, M_{q,K})$ to its signing oracle, and receives $\Sigma_{q,K}$. In either case, it outputs $\Sigma_{q,k}$.
- Corruption queries with index k: If $k \neq K$, $\mathcal{A}_1$ outputs sk_k. Otherwise, $\mathcal{A}_1$ queries its corruption oracle to get sk and outputs $\mathsf{sk}_K = \mathsf{sk}$.

Eventually, $\mathcal{A}_K$ outputs a forgery $(g^*, \mathsf{PK}^*, \mathcal{T}^*, m^*, \sigma^*)$ and $\mathcal{A}_1$ just outputs whatever $\mathcal{A}_K$ outputs.

If $\mathcal{A}_K$ corrupts all signers, the forgery must be a Type-II forgery. Then $\mathcal{A}_1$ wins if $\mathcal{A}_K$ wins. Otherwise, with at least $\frac{1}{K}$ probability, signer K is not corrupted, $\mathsf{pk}_K \in \mathsf{PK}^*$ and $\forall\, \mathcal{T}_{q,K} \in \mathcal{T}^*$, $m^* \notin g^*(\cdots, M_{q,K}, \cdots)$. In such case, $\mathcal{A}_1$ also wins if $\mathcal{A}_K$ wins. □

Proof (Theorem 2). It is well known that EUF-CMA-secure digital signatures can be constructed from one-way functions [Lam79, Rom90]. Thus, suppose that $\mathcal{DS}$ is EUF-CMA-secure.

Suppose there exists an adversary $\mathcal{A}_{\mathcal{HS}}$ that produces a forgery in $\mathcal{HS}$ with non-negligible probability. Suppose Π_n is sound for all $n \in [N]$, we show how to construct an adversary $\mathcal{A}_{\mathcal{DS}}$ that uses $\mathcal{A}_{\mathcal{HS}}$ to produce a forgery of $\mathcal{DS}$. $\mathcal{A}_{\mathcal{DS}}$ acts as a challenger in the $\mathsf{cEUF\text{-}CMA}$ game of $\mathcal{HS}$.

$\mathcal{A}_{\mathcal{DS}}$ obtains from its challenger the public key $\mathsf{pk}_{\mathcal{DS}}$. It generates for each n $(\mathsf{crs}_n, \mathsf{td}_n) \leftarrow \Pi_n.\mathsf{E}^1(1^\lambda)$, a simulated crs_n for Π_n, together with a trapdoor td_n, and forwards the public parameters $\mathsf{pp} = (1^\lambda, \mathsf{crs}_0, \ldots, \mathsf{crs}_N)$ and the public key $\mathsf{pk} = \mathsf{pk}_{\mathcal{DS}}$ to $\mathcal{A}_{\mathcal{HS}}$, where $\mathsf{crs}_0 := \phi$. $\mathcal{A}_{\mathcal{HS}}$ makes two types of queries:

– Signing queries of tag $\tau_q \in \{0,1\}^*$ and a messages set $M_q \in \mathcal{M}^*$: $\mathcal{A}_{\mathcal{DS}}$ forwards $\{(\tau_q, m_{q,j})\}$ to its signing oracle to receive $\sigma_{q,j}$, and outputs $\Sigma_q = \{\sigma_{q,j}\}_j$.
– Corruption query: $\mathcal{A}_{\mathcal{DS}}$ just aborts. This happens with negligible probability, the reason is as follows. If $\mathcal{A}_{\mathcal{HS}}$ issues corruption query, then the only valid forgery it can produce is a type-II forgery. It is easy to see that a type-II forgery breaks the soundness of Π_n for some $n \in [N]$. By the soundness of Π_n, $\mathcal{A}_{\mathcal{HS}}$ can produce such forgery with at most negligible probability.

After querying the oracles, $\mathcal{A}_{\mathcal{HS}}$ will output an alleged forgery of $\mathcal{HS}$, a public key PK^*, a tag $\mathcal{T}^* \in \{0,1\}^*$, a function $g^* \in \mathcal{G}$, some data $m^* \in \mathcal{M}$, and a signature $\sigma^* = (n^*, \sigma')$ such that $\mathsf{Vf}(\mathsf{PK}^*, \mathcal{T}^*, \sigma^*, m^*, g^*) = 1$, $\mathsf{pk} \in \mathsf{PK}^*$ and $\forall\ \mathcal{T}_q \in \mathcal{T}^*$, $m^* \notin g^*(\cdots, M_q, \cdots)$. $\mathcal{A}_{\mathcal{DS}}$ runs $\Pi_n.\mathsf{E}^2$, the extractor of ZK-SNARK for L_n, recursively from $n = n^*$ to $n = 1$, so that it recovers a set of key-message-signature tuples $\{(\mathsf{pk}_k^*, (\tau_k^*, m_k^*), \sigma_k^*)\}$, all pass the verification of $\mathcal{DS}$. Since Π_n is sound for all $n \in [N]$, than with non-negligible probability, there exist a tuple $(\mathsf{pk}, (\tau', m'), \sigma') \in \{(\mathsf{pk}_k^*, (\tau_k^*, m_k^*), \sigma_k^*)\}$ such that (τ', m') is not queried before. Suppose that is the case, then $(\mathsf{pk}, (\tau', m'), \sigma')$ is a valid forgery to $\mathcal{DS}$. We remark that if we have $p \in \mathsf{poly}(\lambda)$ number of recursive extractions and each extraction success with overwhelming probability $1 - u$, where $u \in \mathsf{negl}(\lambda)$, the success probability of the final extraction is:

$$(1-u)^p \geq 1 - \sum_{i=2k+1}^{p} \binom{p}{i} u^i \geq 1 - \sum_{i=2k+1}^{p} (\frac{\mathrm{e}}{i} pu)^i \in 1 - \mathsf{negl}(\lambda)$$

Note that we have used the inequality $\binom{p}{i} \leq (\frac{ep}{i})^i$ where e is the base of natural logarithm.

Proof (Theorem 3). The proof of the above theorem is exactly the same as the proof of Theorem 2. Note that for all $n \in [N]$, the same signing oracle for $\mathcal{DS}$ is required. Therefore, with the transcript of signing oracle queries, the set of extractors $\Pi_n.\mathsf{E}^2$ for the recursive language is able to extract the witnesses. We remark that if N is super-logarithmic, then the total runtime of recursively running the set of extractors $\Pi_n.\mathsf{E}^2$ might become exponential. We thus restrict N to be a constant.

Proof (Theorem 4). Π_n is adaptive zero-knowledge, so there exists a simulator $\mathcal{S}_{\Pi_n} = (\mathcal{S}_{\Pi_n}^{\mathsf{crs}}, \mathcal{S}_{\Pi_n}^{\mathsf{Prove}})$ which simulates a proof π_n for any instance in L_n. To construct a simulator $\mathcal{S}_{\mathcal{HS}}$ for $\mathcal{HS}$, we define $\mathcal{S}_{\mathcal{HS}}^{\mathsf{Setup}}$ which simulates the common reference strings crs_n using $\mathcal{S}_{\Pi_n}^{\mathsf{crs}}$, and $\mathcal{S}_{\mathcal{HS}}^{\mathsf{Sig}}$ which simulates the signatures using $\mathcal{S}_{\Pi_n}^{\mathsf{Prove}}$. The proofs simulated from $\mathcal{S}_{\Pi_n}$ are indistinguishable from the real proofs, so the simulated signatures from $\mathcal{S}_{\mathcal{HS}}$ are indistinguishable from the real signatures.

Proof (Theorem 5). The size of a signature produced by $\mathsf{Eval}(g, (\mathsf{PK}_k, \mathcal{T}_k, M_k, \Sigma_k)_{k=1}^K)$ is the sum of proof length of Π_n for some n and the length of the binary representation of N, which is logarithmic in the security parameter. The succinctness of $\mathcal{HS}$ follows directly from the succinctness property of Π.

Proof (Theorem 7). Suppose there exists an adversary $\mathcal{A}_{\mathcal{FS}}$ that produces a forgery in $\mathcal{FS}$ with non-negligible probability. We show how to construct an adversary $\mathcal{A}_{\mathcal{HS}}$ that uses $\mathcal{A}_{\mathcal{FS}}$ to produce a forgery of $\mathcal{HS}$. $\mathcal{A}_{\mathcal{HS}}$ acts as a challenger in the unforgeability game of $\mathcal{FS}$.

$\mathcal{A}_{\mathcal{HS}}$ receives mpk from the challenger of the $\mathsf{EUF\text{-}CMA}$ game of $\mathcal{HS}$. It forwards public key mpk to $\mathcal{A}_{\mathcal{FS}}$. $\mathcal{A}_{\mathcal{FS}}$ makes two types of queries:

– $\mathcal{O}_{\mathsf{key}}(f, i)$
  • If there exists an entry for (f, i) in the dictionary, output the corresponding value, sk_f^i.
  • Otherwise, generate a public key honestly by $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{HS}.\mathsf{KGen}(1^\lambda)$ and query the signing oracle of $\mathcal{HS}$ to get $\sigma_f^i \leftarrow \mathcal{HS}.\mathsf{Sig}(\mathsf{msk}, \mathsf{pk}, f)$. Then add $\mathsf{sk}_f^i = (\mathsf{sk}, \sigma_f^i)$ to the dictionary entry (f, i) and output sk_f^i.
– $\mathcal{O}_{\mathsf{sign}}(f, i, m)$
  • If there exists an entry for (f, i) in the dictionary, retrieve $\mathsf{sk}_f^i = (\mathsf{sk}, \sigma_f^i)$.
  • Otherwise, generate a public key honestly by $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{HS}.\mathsf{KGen}(1^\lambda)$ and query the signing oracle of $\mathcal{HS}$ to get $\sigma_f^i \leftarrow \mathcal{HS}.\mathsf{Sig}(\mathsf{msk}, \mathsf{pk}, f)$. Then add an entry $(f, i) \rightarrow \mathsf{sk}_f^i = (\mathsf{sk}, \sigma_f^i)$.

- In either case, sample $\tau \leftarrow \{0,1\}^\lambda$, and compute $\sigma_m \leftarrow \mathcal{HS}.\mathsf{Sig}(\mathsf{sk}, \tau, m)$ and $\sigma' \leftarrow \mathcal{HS}.\mathsf{Eval}(U, ((\mathsf{mpk}, \mathsf{pk}, f, \sigma_f^i), (\mathsf{pk}, \tau, m, \sigma_m)))$, where U is the universal circuit. Set $\sigma = (\mathsf{pk}, \tau, \sigma')$ and output $(U(f, m), \sigma)$.

After querying the oracles, $\mathcal{A}_{\mathcal{FS}}$ responds with forgery (m^*, σ^*), where $\sigma^* = (\mathsf{pk}^*, \tau^*, \sigma'^*)$. $\mathcal{A}_{\mathcal{HS}}$ then answer $(U, \{\mathsf{mpk}, \mathsf{pk}^*\}, \{\mathsf{pk}^*, \tau^*\}, m^*, \sigma'^*)$ to its EUF-CMA game. The answer of $\mathcal{A}_{\mathcal{HS}}$ is a valid forgery since, by the definition of the unforgeability game of functional signatures, m^* is not in the range of any f queried to the $\mathcal{O}_{\mathsf{key}}$ oracle, and $m^* \neq f(m)$ for any (f, m) queried to the $\mathcal{O}_{\mathsf{sign}}$ oracle.

Proof (Theorem 8). Let $\mathcal{A}_{\mathcal{FS}}$ be an adversary playing the function-privacy game with a challenger. Since $\mathcal{HS}$ is weakly context-hiding, there exists a simulator $\mathcal{S}_{\mathcal{HS}}$ which, on input $(U, \{\mathsf{mpk}, \mathsf{pk}\}, \{\mathsf{pk}, \tau\}, f(m))$, outputs a signature of $f(m)$ which is indistinguishable from that produced by $\mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_f, m)$. We can thus replace the challenger with the simulator $\mathcal{S}_{\mathcal{HS}}$, which is indistinguishable in the view of $\mathcal{A}_{\mathcal{FS}}$ except with negligible probability. The simulated signatures contain no information about the function f and input message m except for $f(m)$. The probability that the adversary $\mathcal{A}_{\mathcal{FS}}$ guesses correctly in the simulated game is 0.

Proof (Theorem 9). The size of a signature produced by $\mathcal{FS}.\mathsf{Sig}(f, \mathsf{sk}_f, m)$ is the signature length of $\mathcal{HS}$. The succinctness of $\mathcal{FS}$ follows directly from that of $\mathcal{HS}$.

Proof (Theorem 11). Suppose there exists an adversary $\mathcal{A}_{\Pi}$ that breaks the soundness of Π with non-negligible probability. We show how to construct an adversary $\mathcal{A}_{\mathcal{HS}}$ that uses $\mathcal{A}_{\Pi}$ to produce a forgery of $\mathcal{HS}$. $\mathcal{A}_{\mathcal{HS}}$ acts as a challenger in the soundness game of Π.

$\mathcal{A}_{\mathcal{HS}}$ receives pp from the challenger of EUF-CMA game of $\mathcal{HS}$, and forwards the common reference string $\mathsf{crs} := \mathsf{pp}$ to $\mathcal{A}_{\Pi}$. Eventually, $\mathcal{A}_{\Pi}$ responds with (x^*, π^*) such that $\mathsf{Vf}(\mathsf{crs}, x^*, \pi^*) = 1$ but $x^* \notin L$. $\mathcal{A}_{\mathcal{HS}}$ then parses $\pi^* = (\mathsf{pk}^*, \tau^*, \sigma^*)$ and answers $(g, \mathsf{pk}^*, \tau^*, x^*, \sigma^*)$ to its EUF-CMA game. Since $x^* \notin L$, we have $x^* \neq g(x, w)$ for all $(x, w) \in \mathcal{M}^2$.

Proof (Theorem 12). Since $\mathcal{HS}$ is weakly context-hiding, there exists a simulator $\mathcal{S}_{\mathcal{HS}} = (\mathcal{S}_{\mathcal{HS}}^{\mathsf{Setup}}, \mathcal{S}_{\mathcal{HS}}^{\mathsf{Sig}})$ such that, $\mathcal{S}_{\mathcal{HS}}^{\mathsf{Setup}}$ simulates the public parameter, and $\mathcal{S}_{\mathcal{HS}}^{\mathsf{Sig}}$ simulates on input $(R, \mathsf{pk}, \tau, x)$ a signature on x which is statistically close to the real signatures. We can thus construct $\mathcal{S}_{\Pi}^{\mathsf{crs}}$ using $\mathcal{S}_{\mathcal{HS}}^{\mathsf{Setup}}$ and $\mathcal{S}_{\Pi}^{\mathsf{Prove}}$ using $\mathcal{S}_{\mathcal{HS}}^{\mathsf{Sig}}$, and conclude that Π is zero-knowledge.

Proof (Theorem 13). The proof produced by $\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)$ consists of a λ-bit string and a signature of $\mathcal{HS}$. The succinctness of Π follows directly from that of $\mathcal{HS}$.

Proof (Theorem 15). Suppose there exists an adversary $\mathcal{A}_{\mathcal{HS}'}$ that produces a forgery in $\mathcal{HS}'$ with non-negligible probability. Suppose H is collision-resistant, we show how to construct an adversary $\mathcal{A}_{\mathcal{HS}}$ that uses $\mathcal{A}_{\mathcal{HS}'}$ to produce a forgery of $\mathcal{HS}$. $\mathcal{A}_{\mathcal{HS}}$ acts as a challenger in the EUF-CMA game of $\mathcal{HS}'$.

$\mathcal{A}_{\mathcal{HS}}$ received pp and $\mathsf{pk}_{\mathcal{HS}}$ from its challenger. It forwards pp and the public key $\mathsf{pk} = \mathsf{pk}_{\mathcal{HS}}$ to $\mathcal{A}_{\mathcal{HS}'}$.

$\mathcal{A}_{\mathcal{HS}'}$ makes three types of queries. For delegation and signing queries, $\mathcal{A}_{\mathcal{HS}}$ queries the challenger of $\mathcal{HS}$ for the signatures γ and σ'. For corruption query, it queries the challenger of $\mathcal{HS}$ for the secret key $\mathsf{sk} = \mathsf{sk}_{\mathcal{HS}}$. After querying the oracles, $\mathcal{A}_{\mathcal{HS}'}$ responds with forgery, a function $g^* \in \mathcal{G}$, a public key PK^*, a tag $\mathcal{T}^* \in \{0,1\}^*$, a message $m^* \in \mathcal{M}$, a signature $\sigma^* = (\gamma^*, \sigma'^*)$, and a history Δ^*. The only ways for the forgery to be valid is to either forge the underlying $\mathcal{HS}$ signature or find a collision in H. Since the latter is infeasible, for otherwise we can construct adversaries for breaking H. $\mathcal{A}_{\mathcal{HS}}$ guess whether σ'^* or γ^* is a forgery and sends the corresponding signature to its cEUF-CMA game.

Proof (Theorem 16). The succinctness of the σ' part of a signature is trivial. For the γ part, succinctness follows since there is one $\mathcal{DS}$ signature on a λ-bit message for each member of each delegation chain.

The size of signatures produced by Del, Sig, and Eval, namely, $(\sigma_1, \ldots, \sigma_{\mathfrak{N}}) \leftarrow \mathsf{Del}(\mathsf{sk}, \mathsf{PK}, \mathsf{pk}', (m_1, \ldots, m_{\mathfrak{N}}))$, $(\sigma_1, \ldots, \sigma_{\mathfrak{N}}) \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{PK}, \tau, (m_1, \ldots, m_{\mathfrak{N}}))$ and $\sigma \leftarrow \mathsf{Eval}(g, (\mathsf{pk}_{k,d}, \mathsf{PK}_{k,d}, \tau_{k,d}, (m_{k,d,j}, \sigma_{k,d,j})_{j=1}^{N_{k,d}})_{k=1,d=1}^{K,D_k})$ are the same as those in $\mathcal{HS}$. The succinctness of $\mathcal{HS}'$ follows directly from the succinctness of $\mathcal{HS}$.