# Secure Efficient History-Hiding Append-Only Signatures in the Standard Model

Benoît Libert, Marc Joye, Moti Yung, Thomas Peters

# Secure Efficient History-Hiding Append-Only Signatures in the Standard Model

Benoît Libert[1], Marc Joye[2], Moti Yung[3], and Thomas Peters[4]

[1] Ecole Normale Supérieure de Lyon, Laboratoire LIP (France)
[2] Technicolor (USA)
[3] Google Inc. and Columbia University (USA)
[4] Ecole Normale Supérieure (France)

**Abstract.** As formalized by Kiltz *et al.* (ICALP '05), append-only signatures (AOS) are digital signature schemes where anyone can publicly append extra message blocks to an already signed sequence of messages. This property is useful, *e.g.*, in secure routing, in collecting response lists, reputation lists, or petitions. Bethencourt, Boneh and Waters (NDSS '07) suggested an interesting variant, called *history-hiding* append-only signatures (HH-AOS), which handles messages as sets rather than ordered tuples. This HH-AOS primitive is useful when the exact order of signing needs to be hidden. When free of subliminal channels (i.e., channels that can tag elements in an undetectable fashion), it also finds applications in the storage of ballots on an electronic voting terminals or in other archival applications (such as the record of petitions, where we want to hide the influence among messages). However, the only subliminal-free HH-AOS to date only provides heuristic arguments in terms of security: Only a proof in the idealized (non-realizable) random oracle model is given. This paper provides the first HH-AOS construction secure in the standard model. Like the system of Bethencourt *et al.*, our HH-AOS features constant-size public keys, no matter how long messages to be signed are, which is atypical (we note that secure constructions often suffer from a space penalty when compared to their random-oracle-based counterpart). As a second result, we show that, even if we use it to sign ordered vectors as in an ordinary AOS (which is always possible with HH-AOS), our system provides considerable advantages over existing realizations. As a third result, we show that HH-AOS schemes provide improved identity-based ring signatures (i.e., in prime order groups and with a better efficiency than the state-of-the-art schemes).

**Keywords:** Homomorphic signatures, provable security, privacy, unlinkability, standard model, superset predicates, archive integrity.

## 1 Introduction

Append-only signatures (AOS), as introduced by Kiltz, Mityagin, Panjwani and Raghavan [37], are signature schemes where, given a signature on a multi-block message $(M_1, \ldots, M_n)$, anyone can publicly compute a signature on the message $(M_1, \ldots, M_n, M_{n+1})$, for any $M_{n+1}$. Kiltz *et al.* provided both generic constructions, based on any signature scheme, and concrete constructions based on specific assumptions. They further proved that AOS are equivalent to hierarchical identity-based signatures [46, 30]. Importantly, the schemes of [37] are inherently history-preserving in that signed messages are *ordered* tuples.

In [14], Bethencourt, Boneh and Waters (BBW) noted that certain important applications of incremental signature nature require, in fact, a kind of AOS system that allows authenticating sets (i.e., without divulging any order among elements) rather than ordered tuples. They suggested a primitive, called *History-Hiding Append-Only Signatures* (HH-AOS) that can be seen as a special case of homomorphic signatures. It allows one to sign a set of messages in such a way that anyone can subsequently derive a signature on arbitrary supersets of the initial set. Bethencourt *et al.*

used this primitive to design tamper-evident, history-hiding and subliminal-free mechanisms (by extending techniques due to Molnar *et al.* [41]) for storing ballots on e-voting terminals. To prevent anyone from injecting subliminal information (*e.g.*, by embedding some information in derived signatures), it is required that derived signatures be indistinguishable from original ones on the resulting superset. Independently, Moran, Naor and Segev [42] addressed the same problem using write-once memories rather than digital signatures. They described a deterministic vote-storage mechanism without relying on cryptographic techniques. Their solution fits within a line of work, initiated by Micciancio [40], on history-hiding data structures [40, 43], which recently has been extended to applied systems [8]. While secure against unbounded adversaries, the Moran *et al.* technique [42] is significantly more memory-demanding than [14] and this overhead was proved inherent to deterministic techniques [42]. The HH-AOS approach of Bethencourt *et al.* [14] thus appears to remain the most promising method to reliably store $n$ elements in a history-hiding, tamper-evident and scalable manner, namely, using only $O(n)$ memory.

It is worth noting that HH-AOS are a more powerful primitive than ordinary AOS: any HH-AOS can immediately be turned —by means of a hash-based order-embedding transformation— into an equally efficient regular append-only signature. HH-AOS schemes are thus more versatile as they can also be used in all the applications which append-only signatures were initially designed for.

Related Work. Homomorphic signatures were first suggested by Desmedt [24] as a new concept useful in the validation of computer operation. Johnson *et al.* [36] provided security definitions and examples of set homomorphic signatures. Several such constructions in [36, 5, 6] allow for subset derivation (i.e., a signature on a set allows deriving a signature on arbitrary subsets of that set) but none of these works considers the dual superset homomorphism case. The latter was investigated for the first time by Bethencourt *et al.* [14] who provided two HH-AOS realizations which both have some limitations pointed at by the original authors (in essence, demonstrating the associated difficulties with such a scheme). The first one is a generic construction, based on any signature, where the public key has linear size in the maximal size of sets to be signed. As a consequence, this construction requires the signer to determine an upper bound on the cardinality of sets when generating a key pair. Moreover, this generic construction is not free of subliminal channels. The reason is that it allows the party running the signature derivation algorithm to choose certain values pseudo-randomly (rather than truly randomly), which allows a distinguisher to infer some information on the derivation history of signatures.

The second construction of [14] is a subliminal-free system built upon the aggregate signature scheme of Boneh *et al.* [20]. It eliminates the disadvantages of the first scheme in that it provides constant-size public keys and removes the need for an *a priori* bound on the cardinality of authenticated sets. However, while practical, this second scheme is only shown secure in the random oracle model [11]. Recall that it is widely accepted that the random oracle methodology, while better than providing no proof whatsoever, is an idealization that may have no standard model instantiation. Indeed, at times, it is provably unrealizable, as was shown by a number of works (*e.g.*, [21]).

So far, the only apparent way to build a HH-AOS system in the standard model —let alone with constant-size public keys— is to take advantage of aggregate signatures [34, 35] in order to instantiate the BBW system system [14] outside the random oracle idealization. (As explained in Appendix C, sequential aggregate signatures like [39] do not suffice for this.) This requires standard model instantiations [25, 35] of Full Domain Hash [12]. As of now, this is only known under the recent "multi-linear maps" [27], which still have no practical realizations and serve as polynomial

plausibility only. Even the recent results of Hohenberger *et al.* [35] rely on indistinguishability obfuscation [28], known to exist from multi-linear maps only. Thus, such possible ideas cannot yield practical schemes based on simple standard assumptions (like Diffie-Hellman or Decision Linear [16]). In addition, multi-linear maps are quite new, and the state of their secure implementation remains unclear.

Our Contribution. We describe the first efficient history-hiding append-only signature with constant-size public keys in the standard model (by "constant" we mean that it only depends on the security parameter, and not on the cardinality of sets to be signed). This new scheme further provides perfectly re-randomizable signatures, which guarantees the absence of subliminal channels.

Our scheme also provably satisfies a definition of unlinkability stronger than that of [14]. We actually re-cast the syntax of HH-AOS schemes in the definitional framework of Ahn *et al.* [5] for homomorphic signatures. The privacy notion of [5] mandates that derived signatures be statistically indistinguishable from original signatures, *even* when these are given to the distinguisher. In [6], Attrapadung *et al.* further strengthened the latter privacy notion by considering all valid-looking original signatures and not only those in the range of the signing algorithm.

Our construction is asymptotically as efficient as the original BBW realization. Even if we ignore its history-hiding property, it favorably compares to existing append-only signatures [37] in that it appears to be the only known AOS realization that simultaneously provides the following properties: (i) full security (i.e., unforgeability in a model where the adversary can adaptively choose its target message); (ii) constant-size public keys; and (iii) privacy in the sense of the strongest definition considered in [6]. In comparison, the certificate-based generic AOS scheme of [37] is easily seen not to reach the latter level of privacy. As for other fully secure constructions with short public keys, they are all obtained by applying the Naor transformation [17] to unbounded hierarchical identity-based encryption systems [38], which build on Waters' dual system encryption technique [48]. Since the latter always involves at least two distinct distributions of valid signatures (or private keys), it seems inherently incompatible with the information-theoretic privacy notion used in [6].

Our scheme is motivated by ideas that were used in [6] to construct a subset homomorphic signature (namely, a signature on a set authenticates the entire powerset of that set). These ideas, in turn, are augmented by other novel techniques and ideas. Like [6], we rely on the randomizability of Groth-Sahai proofs [32] to render signatures perfectly randomizable. However, superset predicates seem harder to handle than their subset counterpart. Indeed, if we disregard privacy properties, simple constructions[5] readily solve the subset case whereas no such thing is known to work for superset predicates, even when privacy is not a concern. Like [6], our approach proceeds by generating a fresh ephemeral public key $X = g^x$ for each set to be signed. The underlying private key is split into $n$ additive shares $\{\omega_i\}_{i=1}^n$ such that $x = \sum_{i=1}^n \omega_i$, where $n$ is the cardinality of the set. Each of these is then used to sign a set element $m_i$ in the fashion of Boneh-Lynn-Shacham [19] signatures, by computing $H_{\mathbb{G}}(m_i)^{\omega_i}$ using a number theoretic hash function $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$. Although BLS signatures are only known to be secure in the random oracle model (at least in their original form), we, in contrast, can prove the security of the scheme in the standard model as long as $H_{\mathbb{G}}$ is programmable [33] in the same way as the hash function used in [47]. At the same time, we depart from the security proof of [47] in that the programmability of $H_{\mathbb{G}}$ is used in a different way which is closer to the security proofs of Hofheinz and Kiltz [33]. Recall that programmable

[5] For example, as mentioned in [6, Section 5], one can merely sign each set using a new ephemeral public key that is certified by the long-term key.

hash functions [33] are number theoretic hash functions where the hash value $H_{\mathbb{G}}(m)$ is linked to its representation $g^{a_m}h^{b_m}$ for given base elements $g, h \in \mathbb{G}$. While security proofs in the standard model often require $\log_g(H_{\mathbb{G}}(m))$ to be available in the forgery message and unavailable in signed messages, we proceed the other way around: at some crucial signing query $\mathsf{Msg} = \{m_1, \ldots, m_n\}$, we require $H_{\mathbb{G}}(m)$ not to depend on $h$ for exactly one set element $m_i \in \mathsf{Msg}$.

Relation to Identity-Based Ring Signatures. Ring signatures, as introduced by Rivest, Shamir and Tauman [44] allow users to anonymously sign messages on behalf of any *ad hoc* set of users that includes them. Their typical application is to allow a source to anonymously reveal a sensitive information while providing guarantees of trustworthiness.

While ring signatures are known from 2001, rigorous security definitions remained lacking until the work of Bender *et al.* [13] and efficient constructions in the standard model were only given by Shacham and Waters [45] and by Chandran *et al.* [22]. In the identity-based setting, constructing ring signatures remains a non-trivial problem as generic constructions from ordinary ring signatures do not appear to work.

Identity-based ring signatures are extensions of ring signatures [44] to the identity-based setting [46]. They are signature schemes wherein users can employ a private key derived from their identity to sign messages on behalf of any set of identities that includes theirs. The verifier is convinced that a signature was created by a ring member but does not learn anything else. Recently, Au *et al.* [7] described a fully secure identity-based ring signature in the standard model using composite order groups. Their scheme seems amenable for constructing a HH-AOS system. However, due to the use of the dual system technique [29], it cannot achieve the same level of privacy as our scheme (as we discuss later on). Interestingly, any HH-AOS scheme, in fact, gives an identity-based ring signature as the private key of some identity $id$ can consist of a HH-AOS signature on the singleton $\{0\|id\}$ which allows the derivation of a signature on the set $\{0\|id, 0\|id_1, \ldots, 0\|id_n, 1\|M\|\mathcal{R}\}$, where $M$ is the message and $\mathcal{R} = \{id, id_1, \ldots, id_n\}$ is the ring. As detailed in Section 4, we obtain fully secure identity-based ring signatures based on simple assumptions in prime order groups, which allow for a much better efficiency and a stronger flavor of anonymity than [7].

## 2 Background

### 2.1 Definitions for History-Hiding Append-Only Signatures

We first recall the original syntactic definition of history-hiding append-only signatures.

**Definition 1 ([14]).** *An **History-Hiding Append-Only Signatures** (HH-AOS) is a tuple of algorithms* $(\mathsf{Keygen}, \mathsf{Append}, \mathsf{Verify})$ *with the following specifications.*

**$\mathsf{Keygen}(\lambda)$:** *takes as input a security parameter $\lambda \in \mathbb{N}$ and outputs a public key $PK$ and a private key $SK = \Phi$ which consists of an initial signature $\Phi$ on the empty set $\emptyset$.*

**$\mathsf{Append}(PK, \Phi, S, m)$:** *given a public key $PK$, a signature $\Phi$ for some set $S$ and a message $m \in \{0,1\}^*$, this algorithm outputs $\perp$ if $\Phi$ is not a valid signature on the set $S$ or if $m \in S$. Otherwise, it outputs a signature $\Phi'$ on the augmented set $S' = S \cup \{m\}$.*

**$\mathsf{Verify}(PK, S, \Phi)$:** *given a public key $PK$, and a presented signature $\Phi$ for a given set $S$, this algorithm outputs 1 if $\Phi$ is a valid signature for $S$ and 0 otherwise.*

CORRECTNESS. For any integers $\lambda \in \mathbb{N}$ and $n \in \mathsf{poly}(\lambda)$, all key pairs $(PK, SK) \leftarrow \mathsf{Keygen}(\lambda)$ and all sets $S = \{m_1, \ldots, m_n\}$, if $\Phi_0 = SK$, $S_0 = \emptyset$ and $\Phi_i \leftarrow \mathsf{Append}(PK, \Phi_{i-1}, S_i, m_i)$, where $S_i = S_{i-1} \cup \{m_i\}$, for $i = 1$ to $n$, then $\mathsf{Verify}(PK, S, \Phi_n) = 1$.

Bethencourt *et al.* [14] define two security properties of HH-AOS schemes which are called *append-only unforgeability* and *history-hiding*. These properties can be defined as follows.

**Definition 2.** *A HH-AOS scheme* $(\mathsf{Keygen}, \mathsf{Append}, \mathsf{Verify})$ *is* ***append-only unforgeable*** *if no PPT adversary has non-negligible advantage in the following game:*

1. *The challenger generates a key pair* $(PK, SK) \leftarrow \mathsf{Keygen}(\lambda)$ *and hands* $PK$ *to the adversary* $\mathcal{A}$.
2. *On polynomially occasions, the adversary* $\mathcal{A}$ *chooses a set* $S = \{m_1, \ldots, m_n\}$*, for some arbitrary* $n \in \mathsf{poly}(\lambda)$*. We assume w.l.o.g. that* $m_1, \ldots, m_n$ *are sorted in lexicographical order. For* $i = 1$ *to* $n$*, the challenger computes* $\Phi_i \leftarrow \mathsf{Append}(PK, \Phi_{i-1}, S_{i-1}, m_i)$*, where* $S_i = S_{i-1} \cup \{m_i\}$ *for each* $i \in \{1, \ldots, n\}$ *and with* $S_0 = \emptyset$*,* $\Phi_0 = SK$*. Then,* $\Phi_n$ *is returned to* $\mathcal{A}$.
3. $\mathcal{A}$ *outputs a pair* $(S^\star, \Phi^\star)$ *and wins if: (i)* $\mathsf{Verify}(PK, S^\star, \Phi^\star) = 1$*; (ii) If* $S_1, \ldots, S_q$ *denote the sets for which* $\mathcal{A}$ *obtained signatures at Step 2, then* $S_i \not\subseteq S^\star$ *for each* $i \in \{1, \ldots, q\}$*. The adversary's advantage is its probability of success, taken over all coin tosses.*

**Definition 3.** *A HH-AOS scheme* $(\mathsf{Keygen}, \mathsf{Append}, \mathsf{Verify})$ *is* ***history-hiding*** *if no PPT adversary has non-negligible advantage in the following game:*

1. *The challenger generates a key pair* $(PK, SK) \leftarrow \mathsf{Keygen}(\lambda)$ *and gives* $PK$ *to the adversary* $\mathcal{A}$.
2. *The adversary* $\mathcal{A}$ *chooses a set* $S = \{m_1, \ldots, m_n\}$*, for some* $n \in \mathsf{poly}(\lambda)$*, and two distinct permutations* $\pi_0, \pi_1 : \{1, \ldots, n\} \to \{1, \ldots, n\}$*. The challenger chooses a random bit* $b \xleftarrow{R} \{0, 1\}$ *and defines* $m'_i = \pi_b(m_i)$ *for each* $i \in \{1, \ldots, n\}$*. It computes* $\Phi_i \leftarrow \mathsf{Append}(PK, \Phi_{i-1}, S_{i-1}, m'_i)$*, where* $S_i = S_{i-1} \cup \{m'_i\}$ *for each* $i \in \{1, \ldots, n\}$ *and with* $S_0 = \emptyset$*,* $\Phi_0 = SK$*. It returns* $\Phi_n$ *to* $\mathcal{A}$.
3. $\mathcal{A}$ *outputs a bit* $b' \in \{0, 1\}$ *and wins if* $b' = b$*. The adversary's advantage is the distance* $\mathbf{Adv}(\mathcal{A}) := |\Pr[b' = b] - 1/2|$.

While the above definition is sufficient for applications like vote storage [14], it can be strengthened in a number of ways. For example, the adversary could be granted access to a signing oracle before and after Step 2. Alternatively, the adversary could be given the private key $SK$ at Step 1 of the game. Finally, we may also ask for security in the statistical (rather than computational) sense.

These stronger security properties will be naturally obtained by viewing HH-AOS schemes as a particular case of homomorphic signatures in the sense of the definitions of [5, 6].

### 2.2 Definitions for Homomorphic Signatures

**Definition 4 ([5]).** *Let* $\mathcal{M}$ *be a message space and* $2^{\mathcal{M}}$ *be its powerset. Let* $P : 2^{\mathcal{M}} \times \mathcal{M} \to \{0, 1\}$ *be a predicate. A message* $m'$ *is said* ***derivable*** *from* $M \subset \mathcal{M}$ *if* $P(M, m') = 1$*. As in [5],* $P^i(M)$ *is defined as the set of messages derivable from* $P^{i-1}(M)$*, where* $P^0(M) := \{m' \in \mathcal{M} \mid P(M, m') = 1\}$*. Finally,* $P^*(M) := \cup_{i=0}^{\infty} P^i(M)$ *denotes the set of messages iteratively derivable from* $M$.

**Definition 5 ([5]).** *A* ***P-homomorphic signature*** *for a predicate* $P : 2^{\mathcal{M}} \times \mathcal{M} \to \{0, 1\}$ *consists of a triple of algorithms* $(\mathsf{Keygen}, \mathsf{SignDerive}, \mathsf{Verify})$ *such that:*

**Keygen($\lambda$):** *takes in a security parameter* $\lambda \in \mathbb{N}$ *and outputs a key pair* $(\mathsf{sk}, \mathsf{pk})$. *As in [5], the private key* $\mathsf{sk}$ *is a signature on the empty tuple* $\varepsilon \in \mathcal{M}$.

**SignDerive$(\mathsf{pk}, (\{\sigma_m\}_{m\in M}, M), m')$:** *is a possibly probabilistic algorithm that inputs a public key* $\mathsf{pk}$, *a set of messages* $M \subset \mathcal{M}$, *a corresponding set of signatures* $\{\sigma_m\}_{m\in M}$ *and a derived message* $m' \in \mathcal{M}$. *If* $P(M, m') = 0$, *it outputs* $\perp$. *Otherwise, it outputs a derived signature* $\sigma'$.

**Verify$(\mathsf{pk}, m, \sigma)$:** *is a deterministic algorithm that takes as input a public key* $\mathsf{pk}$, *a signature* $\sigma$ *and a message* $m$. *It outputs* 0 *or* 1.

The empty tuple $\varepsilon \in \mathcal{M}$ satisfies $P(\varepsilon, m) = 1$ for each message $m \in \mathcal{M}$. Similarly to Ahn *et al.* [5], we define $\mathsf{Sign}(\mathsf{pk}, \mathsf{sk}, m)$ as the algorithm that runs[6] $\mathsf{SignDerive}(\mathsf{pk}, (\mathsf{sk}, \varepsilon), m)$ and outputs the result. For any $M = \{m_1, \ldots, m_k\} \subset \mathcal{M}$, we let $\mathsf{Sign}(\mathsf{sk}, M) := \{\mathsf{Sign}(\mathsf{sk}, m_1), \ldots, \mathsf{Sign}(\mathsf{sk}, m_k)\}$. Finally, we write $\mathsf{Verify}(\mathsf{pk}, M, \{\sigma_m\}_{m\in M}) = 1$ to say that $\mathsf{Verify}(\mathsf{pk}, m, \sigma_m) = 1$ for each $m \in M$.

CORRECTNESS. For all key pairs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\lambda)$, for any message set $M \subset \mathcal{M}$ and any single message $m' \in \mathcal{M}$ such that $P(M, m') = 1$, the following conditions have to be satisfied:

(i) $\mathsf{SignDerive}(\mathsf{pk}, (\mathsf{Sign}(\mathsf{sk}, M), M), m') \neq \perp$;
(ii) $\mathsf{Verify}\big(\mathsf{pk}, m', \mathsf{SignDerive}(\mathsf{pk}, (\mathsf{Sign}(\mathsf{sk}, M), M), m')\big) = 1$.

**Definition 6 ([5]).** *A P-homomorphic signature scheme is* **unforgeable** *if no PPT adversary has noticeable advantage in the game below:*

1. *The challenger generates a key pair* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\lambda)$ *and gives* $\mathsf{pk}$ *to the adversary* $\mathcal{A}$. *It initializes two initially empty tables* $T$ *and* $Q$.
2. $\mathcal{A}$ *adaptively interleaves the following queries.*
   - *Signing queries:* $\mathcal{A}$ *chooses a message* $m \in \mathcal{M}$. *The challenger replies by choosing a handle* $\mathsf{h}$, *runs* $\sigma \leftarrow \mathsf{Sign}(sk, m)$ *and stores* $(\mathsf{h}, m, \sigma)$ *in a table* $T$. *The handle* $h$ *is returned to* $\mathcal{A}$.
   - *Derivation queries:* $\mathcal{A}$ *chooses a vector of handles* $\vec{\mathsf{h}} = (\mathsf{h}_1, \ldots, \mathsf{h}_k)$ *and a message* $m' \in \mathcal{M}$. *The challenger first retrieves the tuples* $\{(\mathsf{h}_i, m_i, \sigma_i)\}_{i=1}^k$ *from the table* $T$ *and returns* $\perp$ *if one of them is missing. Otherwise, it defines* $M := (m_1, \ldots, m_k)$ *and* $\{\sigma_m\}_{m\in M} = \{\sigma_1, \ldots, \sigma_k\}$. *If* $P(M, m') = 1$, *the challenger runs* $\sigma' \leftarrow \mathsf{SignDerive}\big(\mathsf{pk}, (\{\sigma_m\}_{m\in M}, M), m'\big)$, *chooses a handle* $\mathsf{h}'$, *stores* $(\mathsf{h}', m', \sigma')$ *in* $T$ *and returns* $\mathsf{h}'$ *to* $\mathcal{A}$.
   - *Reveal queries:* $\mathcal{A}$ *chooses a handle* $\mathsf{h}$. *If no entry of the form* $(\mathsf{h}, m', \sigma')$ *exists in* $T$, *the challenger returns* $\perp$. *Otherwise, it returns* $\sigma'$ *to* $\mathcal{A}$ *and adds* $(m', \sigma')$ *to the set* $Q$.
3. *The adversary* $\mathcal{A}$ *outputs a pair* $(\sigma', m')$ *and wins if: (i)* $\mathsf{Verify}(\mathsf{pk}, m', \sigma') = 1$; *(ii) If* $M \subset \mathcal{M}$ *is the set of messages in* $Q$, *then* $m' \notin P^*(M)$.

Ahn *et al.* [5] considered a strong notion of unconditional privacy that requires the inability of distinguishing derived signatures from original ones, *even* when these are given along with the private key. In [5], it was showed that, if a scheme is strongly context hiding, then Definition 6 can be simplified by only providing the adversary with an ordinary signing oracle.

As noted in [6], specific applications may require an even stronger definition. The following definition makes sense when homomorphic signatures are randomizable and/or the verifier accepts several distributions of valid signatures.

[6] The intuition here is that any message can be derived when the original signature contains the signing key.

**Definition 7 ([6]).** *An homomorphic signature* $(\mathsf{Keygen}, \mathsf{Sign}, \mathsf{SignDerive}, \mathsf{Verify})$ *is* ***completely context hiding*** *for the predicate $P$ if, for all key pairs* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\lambda)$*, for all message sets $M \subset \mathcal{M}^*$ and all messages $m' \in \mathcal{M}$ such that $P(M, m') = 1$, for all signatures $\{\sigma_m\}_{m \in M}$ such that* $\mathsf{Verify}(\mathsf{pk}, M, \{\sigma_m\}_{m \in M}) = 1$*, the distribution* $\big\{(\mathsf{sk},\ \{\sigma_m\}_{m \in M},\ \mathsf{Sign}(\mathsf{sk}, m'))\big\}_{sk, M, m'}$ *is statistically close to the distribution of* $\big\{\big(\mathsf{sk}, \{\sigma_m\}_{m \in M}, \mathsf{SignDerive}\big(\mathsf{pk}, (\{\sigma_m\}_{m \in M}, M), m'\big)\big)\big\}_{sk, M, m'}$.

We will be interested in HH-AOS systems, which can be seen as $P$-homomorphic signatures for superset predicates: namely, for any two messages $\mathsf{Msg}_1, \mathsf{Msg}_2 \in \mathcal{M}$, we have $P(\mathsf{Msg}_1, \mathsf{Msg}_2) = 1 \iff \mathsf{Msg}_1 \subseteq \mathsf{Msg}_2$. Note that a completely context-hiding homomorphic signature for superset predicates immediately implies a HH-AOS scheme satisfying a stronger privacy property than Definition 3.

In particular, our construction immediately implies an ordinary (i.e., non-history-hiding) AOS scheme that allows signing ordered tuples while enjoying a stronger form of privacy than in [37]. For example, if we consider the generic AOS [37], which builds on any digital signature, a signature on a vector $(m_1, \ldots, m_n)$ is a sequence $(\sigma_0, pk_1, \ldots, \sigma_n, pk_n, sk_n)$ where $\sigma_i = \mathsf{Sign}(sk_i, (m_{i+1} \| pk_{i+1}))$ for each $i \in \{0, \ldots, n-1\}$, $\{pk_i\}_{i=1}^n$ are fresh public keys generated by the signing algorithm and $(pk_0, sk_0)$ is the long term key pair of the scheme. This construction is clearly not completely context-hiding because auxiliary public keys $\{pk_i\}_{i=1}^n$ appear in an original signature and all its derivatives.

Non-generic AOS schemes can be derived from specific HIBE schemes like the one of Boneh, Boyen and Goh [15] but, in the standard model, the public parameters have length linear in the maximal length of signed messages. For the time being, the only known way to construct a fully secure AOS without having to fix a pre-determined maximal message length is to apply Naor's IBE-to-signature transformation [17] to an unbounded HIBE scheme [38]. Unfortunately, the security proof will probably rely on the dual system technique [48] (see also [29]) which is hardly compatible with the privacy notion of Definition 7. The reason is that this technique involves several computationally indistinguishable classes of signatures satisfying the same equations although they have different distributions. The difficulty is that there is usually no way to publicly modify the class that a given signature belongs to, so that a signature and its derivatives must be of the same class. Hence, for any original signatures $\{\sigma_m\}_{m \in M}$ outside the range of $\mathsf{Sign}(\mathsf{sk}, .)$, Definition 7 cannot be satisfied.

In contrast, using any completely context-hiding HH-AOS, we can obtain —seemingly for the first time— a completely context-hiding AOS scheme in the sense of Definition 7, which hides all information about the derivation history of a signature on an ordered tuple. The construction is detailed in Appendix E.

### 2.3 Programmable Hash Functions

A group hash function $H = (\mathsf{PHF.Gen}, \mathsf{PHF.Eval})$ is a pair of algorithms such that, for a security parameter $\lambda \in \mathbb{N}$, a key $\kappa \leftarrow \mathsf{PHF.Gen}(\lambda)$ is generated by the key generation algorithm. This key is used to evaluate the deterministic evaluation algorithm that, on input of a string $X \in \{0,1\}^L$, computes a hash value $H_{\kappa, \mathbb{G}}(X) = \mathsf{PHF.Eval}(\kappa, X) \in \mathbb{G}$, where $\mathbb{G}$ is a cyclic abelian group.

**Definition 8 ([33]).** *A group hash function $H_{\mathbb{G}} : \{0,1\}^* \to \mathbb{G}$ is* $(\boldsymbol{m, n, \gamma, \delta})$***-programmable*** *if there exist PPT algorithms* $(\mathsf{PHF.TrapGen}, \mathsf{PHF.TrapEval})$ *such that:*

– *For generators $g, h \in \mathbb{G}$, the trapdoor key generation algorithm $(\kappa', tk) \leftarrow \mathsf{PHF.TrapGen}(\lambda, g, h)$ outputs a key $\kappa'$ and a trapdoor $tk$ such that, for any $X \in \{0,1\}^L$, $(a_X, b_X) \leftarrow \mathsf{PHF.TrapEval}(tk, X)$ produces integers $a_X$, $b_X$ such that $H_{\kappa',\mathbb{G}}(X) = \mathsf{PHF.Eval}(\kappa', X) = g^{a_X} h^{b_X}$.*
– *For all $g, h \in \mathbb{G}$ and for $\kappa \leftarrow \mathsf{PHF.Gen}(\lambda)$, $(\kappa', tk) \leftarrow \mathsf{PHF.TrapGen}(\lambda, g, h)$, the distributions of $\kappa$ and $\kappa'$ are statistically $\gamma$-close to each other.*
– *For all $g, h \in \mathbb{G}$ and all keys $\kappa'$ produced by $\mathsf{PHF.TrapGen}$, for all $X_1, \ldots, X_m \in \{0,1\}^L$, $Z_1, \ldots, Z_n \in \{0,1\}^L$ such that $X_i \neq Z_j$, the corresponding $(a_{X_i}, b_{X_i}) \leftarrow \mathsf{PHF.TrapEval}(tk, X_i)$ and $(a_{Z_i}, b_{Z_i}) \leftarrow \mathsf{PHF.TrapEval}(tk, Z_i)$ are such that*

$$\Pr[b_{X_1} = \cdots = b_{X_m} = 0 \wedge b_{Z_1}, \ldots, b_{Z_n} \neq 0] \geq \delta \ ,$$

*where the probability is taken over the trapdoor $tk$ produced along with $\kappa'$.*

The hash function of [47] hashes $L$-bit strings $M = m_1 \cdots m_L \in \{0,1\}^L$ by mapping them to $H_{\kappa,\mathbb{G}}(M) = h_0 \cdot \prod_{i=1}^{L} h_i^{m_i}$ using public group elements $(h_0, \ldots, h_L)$. This function is known [47] to be a $(1, n, 0, \delta)$-programmable hash function where $\delta = 1/(8n(L+1))$, for any polynomial $n$. Using a different technique, Hofheinz and Kiltz [33] increased the probability $\delta$ to $O(1/(n\sqrt{L}))$.

### 2.4 Hardness Assumption

We consider bilinear maps $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ over groups of prime order $p$. In these groups, we assume the intractability of the following problem.

**Definition 9 ([16]).** *The **Decision Linear Problem** (DLIN) in a group $\mathbb{G}$ of prime order $p$ is, given $(g^a, g^b, g^{ac}, g^{bd}, \eta)$, with $a, b, c, d \xleftarrow{R} \mathbb{Z}_p$, to decide if $\eta = g^{c+d}$ or $\eta \in_R \mathbb{G}$.*

### 2.5 Structure-Preserving Signatures Secure Against Random Message Attacks

Structure-preserving signatures [1, 2] (SPS) are signature schemes where messages, signatures and public keys all consist of elements of an abelian group over which a bilinear map is efficiently computable. In addition, the verification algorithm proceeds by testing the validity of pairing product equations (as defined in Appendix A).

We use structure-preserving signatures satisfying a relaxed security notion, where the adversary obtains signatures on messages it has no control on. In the following syntax, a structure-preserving signature is a tuple of efficient algorithms $(\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Sign}, \mathsf{Verify})$ where, on input of a security parameter, $\mathsf{Setup}$ produces common public parameters $\mathsf{gk}$ (which typically specify the chosen bilinear groups) to be used by all other algorithms. As for algorithms $\mathsf{Keygen}$, $\mathsf{Sign}$ and $\mathsf{Verify}$, they operate as in an ordinary digital signatures.

In our construction, we need an SPS scheme that satisfies a notion of *extended* random-message security defined by Abe *et al.* [3]. In the definition hereunder, $\mathcal{M}$ denotes an efficient message sampler that takes as input common public parameters $\mathsf{gk}$ and outputs a message $m$ as well as the random coins $\tau$ used to sample it. In short, the definition requires the scheme to remain unforgeable even if the adversary obtains the random coins of $\mathcal{M}$.

**Definition 10 ([3]).** *A signature scheme $(\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Sign}, \mathsf{Verify})$ provides **extended random-message security** (or XRMA security) with respect to a message sampler $\mathcal{M}$ if, for any PPT adversary $\mathcal{A}$ and any polynomial $q \in \mathsf{poly}(\lambda)$, the adversary's advantage is negligible in the following game.*

1. *The challenger runs* $\mathsf{gk} \leftarrow \mathsf{Setup}(\lambda)$ *and* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Keygen}(\mathsf{gk})$. *For* $j = 1$ *to* $q$, *the challenger runs* $(m_j, \tau_j) \leftarrow \mathcal{M}(gk)$ *and computes* $\sigma_j \leftarrow \mathsf{Sign}(\mathsf{gk}, \mathsf{sk}, m_j)$. *The adversary is given* $(\mathsf{gk}, \mathsf{pk}, \{(m_j, \tau_j, \sigma_j)\}_{j=1}^{q})$
2. *The adversary* $\mathcal{A}$ *outputs a pair* $(m^\star, \sigma^\star)$. *It is declared successful if* $\mathsf{Verify}(\mathsf{gk}, \mathsf{pk}, m^\star, \sigma^\star) = 1$ *and* $m^\star \notin \{m_1, \ldots, m_q\}$. *As usual,* $\mathcal{A}$*'s advantage is its probability of success taken over all coin tosses.*

As in [3], we will need an XRMA-secure SPS scheme where $\tau$ contains the discrete logarithms of the group elements that $m$ is made of.

## 3 An Efficient HH-AOS Scheme

The scheme's design is motivated by [6] to construct a homomorphic subset signature, which is exactly the dual primitive of HH-AOS. Like the ring signature of [7], the scheme is also inspired by the Lewko-Waters unbounded HIBE system [38] in that the signature derivation algorithm implicitly transforms an $n$-out-of-$n$ additive secret sharing into a $(n+1)$-out-of-$(n+1)$ additive sharing of the same secret. This transformation actually takes place in the exponent as the shares themselves are not directly available to the derivation algorithm. Lewko and Waters [38] used a similar technique in the key delegation algorithm of their HIBE scheme. However, we depart from [38] in that the construction relies on the partitioning paradigm (i.e., the reduction is unable to sign certain messages that are used to solve a hard problem in the reduction) rather than the dual system approach. The reason is that, as pointed out in [6], the latter makes it harder to construct completely context-hiding schemes.

The construction relies on the properties of the hash function of [47]. A programmable hash function [33] maps a message $m$ to a group element so that the discrete logarithm of $H_{\mathbb{G}}(m) \in \mathbb{G}$ may be available with some probabilities. The hash function of [47] maps a $L$-bit string $m \in \{0,1\}^L$ to the group element $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{i=1}^{L} h_i^{m[i]}$, for uniformly distributed public group elements $(h_0, \ldots, h_L) \in_R \mathbb{G}^{L+1}$. For any $m \in \{0,1\}^L$, it is possible to relate $H_{\mathbb{G}}(m)$ to exponents $a_m, b_m \in \mathbb{Z}_p$ such that $H_{\mathbb{G}}(m) = g^{a_m} h^{b_m}$. As defined in [33], a $(m,n)$-programmable hash function is a hash function such that, for all $X_1, \ldots, X_m \in \{0,1\}^L$, $Z_1, \ldots, Z_n \in \{0,1\}^L$ with $X_i \neq Z_j$, the probability that $\bigwedge_{i=1}^{m} b_{X_i} = 0$ and $\bigwedge_{j=1}^{n} b_{Z_j} \neq 0$ is non-negligible.

It is known [47] that Waters' hash function is $(1,q)$-programmable with probability $1/8q(L+1)$. If this hash function is used to instantiate the Boneh *et al.* signatures [19] (for which a signature on $m$ consists of $H_{\mathbb{G}}(m)^{sk}$, where $sk$ is the private key), this allows proving its one-time security (i.e., its security in a game where the adversary is only allowed one signing query) in the standard model: the adversary's unique signing query $m$ is answered by computing $H_{\mathbb{G}}(m)^{sk} = (g^{sk})^{a_m}$ from the public key $g^{sk}$ if $b_m = 0$. If the adversary forges a signature on $m^\star$ such that $b_{m^\star} \neq 0$, the reduction can extract $h^{sk}$ and solve a Diffie-Hellman instance.

Our idea is to sign a set $\mathsf{Msg} = \{m_i\}_{i=1}^{n}$ by generating a fresh one-time key pair $(x, g^x) \in \mathbb{Z}_p \times \mathbb{G}$ for a BLS-type signature. The one-time public key $X = g^x$ is certified using the long-term key of a structure-preserving signature. Finally, $\mathsf{Msg} = \{m_i\}_{i=1}^{n}$ is signed by picking $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $\sum_{i=1}^{n} \omega_i = x$ and generating pairs $(\sigma_{i,1}, \sigma_{i,2}) = (H_{\mathbb{G}}(m_i)^{\omega_i}, g^{\omega_i})$, so that the verifier can check that $\prod_{i=1}^{n} \sigma_{i,2} = X$ and $e(\sigma_{i,1}, g) = e(H_{\mathbb{G}}(m_i), \sigma_{i,2})$ for each $i$. This allows anyone to publicly add new elements to the set by transforming the sharing $\{\omega_i\}_{i=1}^{n}$ into a new sharing $\{\omega'_i\}_{i=1}^{n+1}$ of the same value. At the same time, it will be infeasible to publicly remove elements from the signed set.

To guarantee the full context-hiding security, we refrain from letting $(\sigma_{i,1}, \sigma_{i,2})$ appear in clear and replace them by perfectly-hiding Groth-Sahai commitments to $(\sigma_{i,1}, \sigma_{i,2})$ along with NIWI randomizable proofs (which are recalled Appendix A) showing that committed values satisfy the appropriate relations.

In the notations hereunder, for any $h \in \mathbb{G}$ and any vector of group elements $\vec{g} = (g_1, g_2, g_3) \in \mathbb{G}^3$, the vector $\big(e(h, g_1), e(h, g_2), e(h, g_3)\big) \in \mathbb{G}_T^3$ is denoted by $E(h, \vec{g})$.

**Keygen($\lambda$):**

1. Choose a SPS scheme $\Pi^{\mathrm{SPS}} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Sign}, \mathsf{Verify})$ allowing to sign messages consisting of a single group element. We denote by $\ell_{\mathsf{sps}}$ and $v_{\mathsf{sps}}$ the number of group elements per signature and the number of verification equations, respectively, in this scheme. Generate common parameter $\mathsf{gk} \leftarrow \Pi^{\mathrm{SPS}}.\mathsf{Setup}(\lambda)$ and a key pair $(sk_{\mathsf{sps}}, pk_{\mathsf{sps}}) \leftarrow \Pi^{\mathrm{SPS}}.\mathsf{Keygen}(\mathsf{gk})$ for this scheme. We assume that $\mathsf{gk}$ includes the description of bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ or prime order $p > 2^\lambda$ with a generator $g \in_R \mathbb{G}$.
2. Generate a Groth-Sahai CRS $\mathbf{f} = (\vec{f}_1, \vec{f}_2, \vec{f}_3)$ for the perfect witness indistinguishability setting. Namely, choose $\vec{f}_1 = (f_1, 1, g)$, $\vec{f}_2 = (1, f_2, g)$, and $\vec{f}_3 = \vec{f}_1^{\,\xi_1} \cdot \vec{f}_2^{\,\xi_2} \cdot (1, 1, g)^{-1}$, with $f_1, f_2 \xleftarrow{R} \mathbb{G}$, $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p$.
3. Choose a vector $(h_0, h_1, \ldots, h_L) \xleftarrow{R} \mathbb{G}^{L+1}$ which defines the function $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ that maps any $m \in \{0,1\}^L$ to $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{i=1}^{L} h_i^{m[i]}$.

The public key is defined to be $\mathsf{pk} := \Big(\mathsf{gk},\ \mathbf{f},\ pk_{\mathsf{sps}},\ \{h_i\}_{i=0}^{L}\Big)$ and the private key is $\mathsf{sk} := sk_{\mathsf{sps}}$. The public key defines $\Sigma = \{0,1\}^L$.

**Sign(sk, Msg):** on input of a message $\mathsf{Msg} = \{m_i\}_{i=1}^n$, where $m_i \in \{0,1\}^L$ for each $i$, and the private key $\mathsf{sk} = sk_{\mathsf{sps}}$, do the following.

1. Generate a one-time public key $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$, and a Groth-Sahai commitment $\vec{C}_X = (1, 1, X) \cdot \vec{f}_1^{\,r_X} \cdot \vec{f}_2^{\,s_X} \cdot \vec{f}_3^{\,t_X}$, with $r_X, s_X, t_X \xleftarrow{R} \mathbb{Z}_p$.
2. Generate a structure-preserving signature $(\theta_1, \ldots, \theta_{\ell_{\mathsf{sps}}}) \in \mathbb{G}^{\ell_{\mathsf{sps}}}$ on the group element $X \in \mathbb{G}$. Then, for each $j \in \{1, \ldots, \ell_{\mathsf{sps}}\}$, generate commitments $\vec{C}_{\theta_j} = (1, 1, \theta_j) \cdot \vec{f}_1^{\,r_{\theta_j}} \cdot \vec{f}_2^{\,s_{\theta_j}} \cdot \vec{f}_3^{\,t_{\theta_j}}$. Finally, generate NIWI arguments $\{\vec{\pi}_{\mathsf{sps},j}\}_{j=1}^{v_{\mathsf{sps}}}$ showing that committed variables $(X, \{\theta_j\}_{j=1}^{\ell_{\mathsf{sps}}})$ satisfy the verification equations of the structure-preserving signature.
3. Choose $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ subject to the constraint $\sum_{i=1}^n \omega_i = x$. Then, for $i = 1$ to $n$, compute $(\sigma_{i,1}, \sigma_{i,2}) = \big(H_{\mathbb{G}}(m_i)^{\omega_i}, g^{\omega_i}\big)$, where the messages are indexed in some pre-determined lexicographical order.[7] Then, for each $i \in \{1, \ldots, n\}$, compute Groth-Sahai commitments

$$\vec{C}_{\sigma_{i,1}} = (1, 1, \sigma_{i,1}) \cdot \vec{f}_1^{\,r_{i,1}} \cdot \vec{f}_2^{\,s_{i,1}} \cdot \vec{f}_3^{\,t_{i,1}},$$
$$\vec{C}_{\sigma_{i,2}} = (1, 1, \sigma_{i,2}) \cdot \vec{f}_1^{\,r_{i,2}} \cdot \vec{f}_2^{\,s_{i,2}} \cdot \vec{f}_3^{\,t_{i,2}}$$

to $\{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^n$. Next, generate a NIWI argument $\vec{\pi}_i$ that $e(\sigma_{i,1}, g) = e(H_{\mathbb{G}}(m_i), \sigma_{i,2})$. This argument is

$$(\pi_{i,1}, \pi_{i,2}, \pi_{i,3}) = \big(g^{r_{i,1}} H_{\mathbb{G}}(m_i)^{-r_{i,2}}, g^{s_{i,1}} H_{\mathbb{G}}(m_i)^{-s_{i,2}}, g^{t_{i,1}} H_{\mathbb{G}}(m_i)^{-t_{i,2}}\big)$$

[7] This follows an observation by Naor and Teague [43] who used lexicographical ordering to make sure that the representation does not depend on the order of insertions.

and satisfies the equation

$$E\big(g, \vec{C}_{\sigma_{i,1}}\big) = E\big(H_{\mathbb{G}}(m_i), \vec{C}_{\sigma_{i,2}}\big) \cdot E\big(\pi_{i,1}, \vec{f}_1\big) \cdot E\big(\pi_{i,2}, \vec{f}_2\big) \cdot E\big(\pi_{i,3}, \vec{f}_3\big) \ . \quad (1)$$

4. Finally, generate a NIWI proof $\vec{\pi}_{sum}$ that $X = \prod_{i=1}^{n} \sigma_{i,2}$. This proof is

$$(\pi_{s,1}, \pi_{s,2}, \pi_{s,3}) = \big(g^{r_X - \sum_{i=1}^{n} r_{i,2}},\ g^{s_X - \sum_{i=1}^{n} s_{i,2}},\ g^{t_X - \sum_{i=1}^{n} t_{i,2}}\big) \quad (2)$$

which satisfies $E\big(g, \vec{C}_X \cdot \prod_{i=1}^{n} \vec{C}_{\sigma_{i,2}}^{-1}\big) = E\big(\pi_{s,1}, \vec{f}_1\big) \cdot E\big(\pi_{s,2}, \vec{f}_2\big) \cdot E\big(\pi_{s,3}, \vec{f}_3\big)$.

Return $\sigma = \big(\vec{C}_X, \{\vec{C}_{\theta_j}\}_{j=1}^{\ell_{\mathsf{sps}}}, \{\vec{\pi}_{\mathsf{sps},j}\}_{j=1}^{v_{\mathsf{sps}}}, \{(m_i, \vec{C}_{\sigma_{i,1}}, \vec{C}_{\sigma_{i,2}}, \vec{\pi}_i)\}_{i=1}^{n},\ \vec{\pi}_{sum}\big)$.

**SignDerive(pk, ($\sigma$, Msg), Msg′):** given the original message $\mathsf{Msg} = \{m_i\}_{i=1}^{n}$, return $\perp$ if $\mathsf{Msg}' \neq \mathsf{Msg} \cup \{m'\}$ for some $m' \in \Sigma$. Otherwise, parse $\sigma$ as above and do the following.

1. Choose $\omega'_1, \ldots, \omega'_{n+1} \xleftarrow{R} \mathbb{Z}_p$ subject to the constraint $\sum_{i=1}^{n+1} \omega'_i = 0$. For each index $i \in \{1, \ldots, n\}$, compute updated Groth-Sahai commitments $\vec{C}'_{\sigma_{i,1}} = (1, 1, H(m_i)^{\omega'_i}) \cdot \vec{C}_{\sigma_{i,1}}$ and $\vec{C}'_{\sigma_{i,2}} = (1, 1, g^{\omega'_i}) \cdot \vec{C}_{\sigma_{i,2}}$. Observe that the argument $\vec{\pi}_i = (\pi_{i,1}, \pi_{i,2}, \pi_{i,3})$ still satisfies the equation $E\big(g, \vec{C}'_{\sigma_{i,1}}\big) = E\big(H_{\mathbb{G}}(m_i), \vec{C}'_{\sigma_{i,2}}\big) \cdot E\big(\pi_{i,1}, \vec{f}_1\big) \cdot E\big(\pi_{i,2}, \vec{f}_2\big) \cdot E\big(\pi_{i,3}, \vec{f}_3\big)$ as it only depends on the randomness of commitments.
2. Set $\sigma_{n+1,1} = H_{\mathbb{G}}(m')^{\omega'_{n+1}}$ and $\sigma_{n+1,2} = g^{\omega'_{n+1}}$. Then, pick random $r_{n+1,1}, s_{n+1,1}, t_{n+1,1} \xleftarrow{R} \mathbb{Z}_p$, $r_{n+1,2}, s_{n+1,2}, t_{n+1,2} \xleftarrow{R} \mathbb{Z}_p$ and compute commitments

$$\vec{C}'_{\sigma_{n+1,1}} = (1, 1, \sigma_{n+1,1}) \cdot \vec{f}_1^{\,r_{n+1,1}} \cdot \vec{f}_2^{\,s_{n+1,1}} \cdot \vec{f}_3^{\,t_{n+1,1}}$$
$$\vec{C}'_{\sigma_{n+1,2}} = (1, 1, \sigma_{n+1,2}) \cdot \vec{f}_1^{\,r_{n+1,2}} \cdot \vec{f}_2^{\,s_{n+1,2}} \cdot \vec{f}_3^{\,t_{n+1,2}}$$

as well as a NIWI argument $\vec{\pi}_{n+1}$ showing that $e(\sigma_{n+1,1}, g) = e(H_{\mathbb{G}}(m'), \sigma_{n+1,2})$, which is obtained as

$$\big(g^{r_{n+1,1}} \cdot H_{\mathbb{G}}(m')^{-r_{n+1,2}}, g^{s_{n+1,1}} \cdot H_{\mathbb{G}}(m')^{-s_{n+1,2}}, g^{t_{n+1,1}} \cdot H_{\mathbb{G}}(m')^{-t_{n+1,2}}\big) \ .$$

3. Update $\vec{\pi}_{sum} = (\pi_{s,1}, \pi_{s,2}, \pi_{s,3})$ by computing

$$\begin{aligned} \vec{\pi}'_{sum} &= (\pi'_{s,1}, \pi'_{s,2}, \pi'_{s,3}) \\ &= \big(\pi_{s,1} \cdot g^{-r_{n+1,2}},\ \pi_{s,2} \cdot g^{-s_{n+1,2}},\ \pi_{s,3} \cdot g^{-t_{n+1,2}}\big) \ . \end{aligned}$$

Note that $\vec{\pi}'_{sum}$ is a valid proof that $X = \prod_{i=1}^{n+1} \sigma_{i,2}$ since $\vec{\pi}_{sum}$ only depends on the randomness of commitments $\vec{C}_X$, $\{\vec{C}_{\sigma_{i,2}}\}_{i=1}^{n}$, which have not been randomized at this point.
4. Re-randomize the commitments $\vec{C}_X$, $\{\vec{C}'_{\sigma_{i,1}}, \vec{C}'_{\sigma_{i,2}}\}_{i=1}^{n+1}$, $\{\vec{C}_{\theta_j}\}_{j=1}^{\ell_{\mathsf{sps}}}$ and the proofs $\{\vec{\pi}_{\mathsf{sps},j}\}_{j=1}^{v_{\mathsf{sps}}}$, $\{\vec{\pi}_i\}_{i=1}^{n+1}$, $\vec{\pi}'_{sum}$. Let $\vec{C}''_X$, $\{\vec{C}''_{\sigma_{i,1}}, \vec{C}''_{\sigma_{i,2}}\}_{i=1}^{n+1}$, $\{\vec{C}''_{\theta_j}\}_{j=1}^{\ell_{\mathsf{sps}}}$ and the proofs $\{\vec{\pi}''_{\mathsf{sps},j}\}_{j=1}^{v_{\mathsf{sps}}}$, $\{\vec{\pi}''_i\}_{i=1}^{n+1}$, $\vec{\pi}''_{sum}$ be the re-randomized commitment and proofs. Note that, in all of these commitments and proofs, the underlying exponents have been updated.

Return $\sigma' = \big(\vec{C}''_X, \{\vec{C}''_{\theta_j}\}_{j=1}^{\ell_{\mathsf{sps}}}, \{\vec{\pi}''_{\mathsf{sps},j}\}_{j=1}^{v_{\mathsf{sps}}}, \{(m_i, \vec{C}''_{\sigma_{i,1}}, \vec{C}''_{\sigma_{i,2}}, \vec{\pi}''_i)\}_{i=1}^{n+1},\ \vec{\pi}''_{sum}\big)$ after having re-organized the indexation of $\{(m_i, \vec{C}''_{\sigma_{i,1}}, \vec{C}''_{\sigma_{i,2}}, \vec{\pi}''_i)\}_{i=1}^{n+1}$ according to the lexicographical order for $\{m_i\}_{i=1}^{n+1}$.

**Verify(pk, Msg, $\sigma$):** given pk, and a message $\mathsf{Msg} = \{m_i\}_{i=1}^n$, where $m_i \in \Sigma$ for each $i$, parse $\sigma$ as above. Return 1 iff the following checks all succeed.

1. Return 0 if $\{\vec{\pi}_{\mathsf{sps},j}\}_{j=1}^{v_{\mathsf{sps}}}$ are not valid proofs that committed group elements $(X, \{\theta_j\}_{j=1}^{\ell_{\mathsf{sps}}})$ satisfy the verification equations of the structure-preserving signature.
2. Return 0 if, there exists $i \in \{1, \ldots, n\}$ such that $\vec{\pi}_i = (\pi_{i,1}, \pi_{i,2}, \pi_{i,3})$ does not satisfy (1).
3. Return 0 if $\vec{\pi}_{sum} = (\pi_{s,1}, \pi_{s,2}, \pi_{s,3})$ is not a valid proof.

Note that message elements $\{m_i\}_{i=1}^n$ can be omitted from the signature if the signature components $\{(\vec{C}_{\sigma_{i,1}}, \vec{C}_{\sigma_{i,2}}, \vec{\pi}_i)\}_{i=1}^n$ are organized according to the lexicographical order of $\{m_i\}_{i=1}^n$.

As in [14], one can finalize the set and prevent any further insertions by adding a special message of the form "$\mathsf{finalize}\|\#\mathsf{Msg}$" to the current message Msg, where #Msg denotes the cardinality of Msg. In this case, the verifier has to return 0 if Msg contains an element of the form "$\mathsf{finalize}\|x$", where $x \neq \#\mathsf{Msg}-1$. We also note that, as in [14], multi-sets can be supported by merely appending a nonce to each added message in order to ensure uniqueness.

The scheme is unconditionally completely context-hiding because, except $\{m_i\}_{i=1}^n$ (which are re-ordered to appear in lexicographical order at each derivation), signatures only consist of perfectly hiding commitments and NIWI proofs. Moreover, in the WI setting, these are uniformly distributed in the space of valid proofs (as stressed in [32][Section 10]). Since these proofs are also perfectly randomizable at each derivation, the complete context-hiding property follows.

The unforgeability is proved under the DLIN assumption and the assumption that the underlying SPS scheme is XRMA-secure. In one step, the proof of Theorem 1 relies on the programmability of the Waters hash function [47].

The security proof assumes a theoretical upper bound $n_{max}$ on the cardinality of sets to be signed. However, we emphasize that this bound does not affect the efficiency of the scheme whatsoever. In particular, the public key size is independent of $n_{max}$ and only depends on the security parameter.

**Theorem 1.** *The scheme is unforgeable assuming that the DLIN assumption holds in $\mathbb{G}$ and that the structure-preserving signature is secure against extended random message attacks.*

*Proof.* Since the scheme is completely context hiding, we can use a simplified definition where the adversary only interacts with a signing oracle. The proof uses a sequence of games where, for each $i \in \{0, 1, 2\}$, $S_i$ denotes the event that the adversary $\mathcal{A}$ wins in $\mathrm{Game}_i$.

**$\mathbf{Game}_0$:** This game is the real game. We denote by $S_0$ the event that the adversary $\mathcal{A}$ manages to output a successful forgery. By definition, $\mathcal{A}$'s advantage is $\Pr[S_0]$.

**$\mathbf{Game}_1$:** We change the generation of the public key and choose $\mathbf{f} = (\vec{f}_1, \vec{f}_2, \vec{f}_3)$ as a perfectly sound Groth-Sahai CRS, for which even an unbounded adversary cannot prove false statements. More precisely, the challenger $\mathcal{B}$ sets up $\vec{f}_1 = (f_1, 1, g)$, $\vec{f}_2 = (1, f_2, g)$ and $\vec{f}_3 = \vec{f}_1^{\,\xi_1} \cdot \vec{f}_2^{\,\xi_2}$, with $f_1 = g^{\phi_1}$ and $f_2 = g^{\phi_2}$, for randomly chosen $\phi_1, \phi_2, \xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p$. If this modification significantly increases the adversary's probability of success, we can build a distinguisher for the DLIN assumption (specifically, the DLIN distinguisher outputs 1 if the adversary is successful and a random bit otherwise). This implies that, under the DLIN assumption, this modification does not significantly affect $\mathcal{A}$'s behavior. We can thus write $|\Pr[S_1] - \Pr[S_0]| \leq \mathbf{Adv}^{\mathrm{DLIN}}(\mathcal{B})$.

**Game$_2$:** In this game, we can explicitly use the discrete logarithms $(\phi_1, \phi_2) = (\log_g(f_1), \log_g(f_2))$ that were defined in Game$_1$ since we are done with the DLIN assumption. When $\mathcal{A}$ outputs a forgery $\sigma^\star$, the challenger $\mathcal{B}$ uses $(\phi_1, \phi_2)$ to extract $X^\star$ from the Groth-Sahai commitment $\vec{C}_X^\star$ contained in $\sigma^\star$ (recall that, due to the modification introduced in Game$_1$, $\vec{C}_X^\star$ is a perfectly binding commitment). We raise a failure event, called $F_2$, and let the challenger $\mathcal{B}$ abort if the extracted $X^\star$ was never involved in any signing query. Clearly, any occurrence of $F_2$ immediately contradicts the extended random-message security of the SPS system as the adversary only gets to see structure-preserving signatures on uniformly distributed group elements $X$. The reduction is similar to that of [3, Theorem 3] and relies on the XRMA security of the underlying SPS scheme for the same reason.[8] We can thus write $|\Pr[S_2] - \Pr[S_1]| \leq \Pr[F_2] \leq \mathbf{Adv}^{\text{XRMA-SPS}}(\mathcal{B})$.

In Game$_2$, we will prove that, conditionally on $\neg F_2$, event $S_2$ can only occur with negligible probability if the Diffie-Hellman assumption holds. Let $(\sigma^\star, \mathsf{Msg}^\star = \{m_1^\star, \ldots, m_{n^\star}^\star\})$ denote $\mathcal{A}$'s forgery. If $F_2$ does not occur, the group element $X^\star$, which is extracted from the commitment $\vec{C}_X^\star$ contained in $\sigma^\star$, *was* used by $\mathcal{B}$ in some signing query. Letting $j \in \{1, \ldots, q\}$ denote the index of that query $\mathsf{Msg}_j = \{m_{j,1}, \ldots, m_{j,n_j}\}$, we know that that $\mathsf{Msg}_j \not\subseteq \mathsf{Msg}^\star$ since $\mathcal{A}$ would not be a successful forger otherwise. Consequently, there exists $\ell \in \{1, \ldots, n_j\}$ such that $m_{j,\ell} \notin \mathsf{Msg}^\star$. Assuming that signed messages $\mathsf{Msg}_1, \ldots, \mathsf{Msg}_q$ are sets of cardinality at most $n_{max}$, Lemma 1 constructs an algorithm $\mathcal{B}'$ breaking the Diffie-Hellman assumption with probability at least $\Pr[S_2|\neg F_2]/(16 \cdot q \cdot n_{max} \cdot (L+1))$. The probability of event $S_2|\neg F_2$ can thus be bounded by $\Pr[S_2|\neg F_2] \leq 16 \cdot q \cdot n_{max} \cdot (L+1) \cdot \mathbf{Adv}^{\text{CDH}}(\mathcal{B}')$.

Since $\Pr[S_2] = \Pr[S_2 \wedge F_2] + \Pr[S_2 \wedge \neg F_2] \leq \Pr[F_2] + \Pr[S_2|\neg F_2]$, we find

$$\Pr[S_0] \leq \mathbf{Adv}^{\text{DLIN}}(\mathcal{B}) + 2 \cdot \mathbf{Adv}^{\text{RMA-SPS}}(\mathcal{B}) + 16 \cdot q \cdot n_{max} \cdot (L+1) \cdot \mathbf{Adv}^{\text{CDH}}(\mathcal{B}')$$

which proves the announced result. □

The programmability properties of the Waters hash function are used in the proof of Lemma 1. In a nutshell, the reduction will have to guess upfront which one-time public key $X_{j^\star}$ will be recycled in the adversary's forgery among those involved in responses to signing queries. When answering this signing query, the reduction will implicitly use the $g^a$ part of its given Diffie-Hellman instance $(g, g^a, g^b)$ to form the one-time public key $X_{j^\star}$. In addition, if the input of the $j^\star$-th signing query is $\mathsf{Msg}_{j^\star} = \{m_{j^\star,i}\}_{i=1}^{n_{j^\star}}$, we know that at least one element of $\mathsf{Msg}_{j^\star}$ will be outside the set $\mathsf{Msg}^\star = \{m_i^\star\}_{i=1}^{n^\star}$ chosen by the adversary for its forgery. If we denote by $m_{j^\star,\ell}$ an arbitrary message in $\mathsf{Msg}_{j^\star} \backslash \mathsf{Msg}^\star$, the reduction will be successful if $H_{\mathbb{G}}(m_{j^\star,\ell}) = g^{a_{m_{j^\star,\ell}}}$, for some known $a_{m_{j^\star,\ell}} \in \mathbb{Z}_p$, and $H_{\mathbb{G}}(m_i^\star) = g^{a_{m_i^\star}} \cdot (g^b)^{b_{m_i^\star}}$ with $b_{m_i^\star} \neq 0$ for each $i \in \{1, \ldots, n^\star\}$. The results of [47, 33] guarantee that these conditions are met with non-negligible probability.

**Lemma 1.** *In* Game$_2$*, if event $S_2|\neg F_2$ occurs with noticeable probability then there exists an algorithm $\mathcal{B}'$ solving the CDH problem with probability at least $\mathbf{Adv}^{\text{CDH}}(\mathcal{B}') \geq \Pr[S_2|\neg F_2]/(16 \cdot q \cdot n_{max} \cdot (L+1))$, where $n_{max}$ is the maximal cardinality of signed subsets.*

*Proof.* Algorithm $\mathcal{B}'$ takes as input $(g, g^a, g^b)$ and aims at computing $g^{ab}$ using its interaction with the adversary in Game$_2$.

[8] In short, for each message $X$ for which the XRMA challenger generates a signature, the reduction needs $x = \log_g(X)$ to properly run Step 3 of the signing algorithm.

To this end, $\mathcal{B}'$ begins by choosing $(h_0, h_1, \ldots, h_L) \in \mathbb{G}^{L+1}$ as in the security proof of Waters signatures [47]. Namely, for any string $m \in \{0,1\}^L$, the hash value $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{i=1}^{L} h_i^{m[i]}$ can be written as $H_{\mathbb{G}}(m) = (g^b)^{J(m)} \cdot g^{K(m)}$ for certain integer-valued functions $J, K : \{0,1\}^L \to \mathbb{Z}_p$ that remain internal to the simulation. In the terminology of programmable hash functions [33], $H_{\mathbb{G}}$ will have to be $(1, 2n_{max} - 1)$-programmable with non-negligible probability $\delta$. Concretely, using the technique of [47], the functions $J$ and $K$ are chosen so that, for any pairwise distinct inputs $m, m_1, \ldots, m_{2n_{max}-1}$, we have $J(m) = 0 \bmod p$ and $J(m_i) \neq 0 \bmod p$ for each $i \in \{1, \ldots, 2n_{max}-1\}$ with non-negligible probability $\delta = 1/(16 \cdot n_{max} \cdot (L+1))$.

Algorithm $\mathcal{B}'$ begins by drawing $j^\star \xleftarrow{R} \{1, \ldots, q\}$ and starts interacting with the forger $\mathcal{A}$.

**Signing queries:** For $j \in \{1, \ldots, q\}$, we let $\mathsf{Msg}_j = \{m_{j,1}, \ldots, m_{j,n_j}\}$, with $n_j \leq n_{max}$, be the $j$-th signing query made by $\mathcal{A}$. These queries are handled by considering two cases:

- If $j \neq j^\star$, $\mathcal{B}'$ chooses a fresh $x_j \xleftarrow{R} \mathbb{Z}_p$, computes $X_j = g^{x_j}$ and answers the query by generating $\omega_1, \ldots, \omega_{n_j} \xleftarrow{R} \mathbb{Z}_p$ such that $\sum_{i=1}^{n_j} \omega_i = x_j$. This allows answering the query faithfully, by generating commitments and proofs according to the specification of the signing algorithm.
- If $j = j^\star$, $\mathcal{B}'$ implicitly defines $X_{j^\star} = g^a$. At this point, $\mathcal{B}'$ considers each message $m_{j^\star,i} \in \mathsf{Msg}_{j^\star}$ and evaluates $J(m_{j^\star,i})$ for each $i \in \{1, \ldots, n_{j^\star}\}$. If $J(m_{j^\star,i}) \neq 0$ for all $i$, $\mathcal{B}'$ halts and declares failure. It also aborts if $\mathsf{Msg}_{j^\star}$ contains more than *one* message $m_{j^\star,i}$ such that $J(m_{j^\star,i}) = 0$. (A lower bound on the probability for $\mathcal{B}'$ not to abort will be determined later on). Otherwise, there exists a unique index $\ell \in \{1, \ldots, n_{j^\star}\}$ such that $J(m_{j^\star,\ell}) = 0$. In this case, we have $H_{\mathbb{G}}(m_{j^\star,\ell}) = g^{K(m_{j^\star,\ell})}$, so that $\mathcal{B}'$ can pick $\omega_1, \ldots, \omega_{\ell-1}, \omega_{\ell+1}, \ldots, \omega_{n_{j^\star}} \xleftarrow{R} \mathbb{Z}_p$ and set

$$\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i} \qquad \sigma_{i,2} = g^{\omega_i} \qquad \text{for } i \in \{1, \ldots, n_{j^\star}\} \backslash \{\ell\},$$

as well as

$$\sigma_{\ell,1} = \left((g^a) \cdot g^{-\sum_{i=1, i\neq\ell}^{n_{j^\star}} \omega_i}\right)^{K(m_{j^\star,\ell})} \qquad \sigma_{\ell,2} = (g^a) \cdot g^{-\sum_{i=1,i\neq\ell}^{n_{j^\star}} \omega_i} .$$

Note that $\{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n_{j^\star}}$ have the correct distribution as they implicitly share $a = \log_g(X_{j^\star})$ in the exponent. Next, $\mathcal{B}'$ generates commitments and NIWI proofs as in the real signing algorithm.

**Forgery:** When $\mathcal{A}$ terminates, it outputs a set $\mathsf{Msg}^\star = \{m_i^\star\}_{i=1}^{n^\star}$ with a valid signature

$$\sigma^\star = \left(\vec{C}_X^\star, \{\vec{C}_{\theta_j}^\star\}_{j=1}^{\ell_{\mathsf{sps}}}, \{\vec{\pi}_{\mathsf{sps},j}^\star\}_{j=1}^{v_{\mathsf{sps}}}, \{(m_i^\star, \vec{C}_{\sigma_{i,1}}^\star, \vec{C}_{\sigma_{i,2}}^\star, \vec{\pi}_i^\star)\}_{i=1}^{n^\star}, \ \vec{\pi}_{sum}^\star\right) .$$

At this point, $\mathcal{B}'$ uses the extraction trapdoor $(\phi_1, \phi_2) = (\log_g(f_1), \log_g(f_2))$ of the commitment to obtain $X^\star$ and $\{\sigma_{i,1}^\star, \sigma_{i,2}^\star\}_{i=1}^{n^\star}$ from $\vec{C}_X^\star$ and $\{C_{\sigma_{i,1}}^\star, C_{\sigma_{i,2}}^\star\}_{i=1}^{n^\star}$, respectively. If one of the following events occurs, $\mathcal{B}'$ aborts and declares failure:

E.1 $X^\star \neq g^a$: This is the event that $\mathcal{B}'$ fails to correctly predict which one-time public key $X_j$ would be re-used in $\mathcal{A}$'s forgery among those involved in signing queries.

E.2 $m_{j^\star,\ell} \in \mathsf{Msg}^\star$.

E.3 There exists $i \in \{1, \ldots, n^\star\}$ such that $J(m_i^\star) = 0$.

If none of these events occurs, the perfect soundness of the proof $\vec{\pi}^{\star}_{sum}$ guarantees that $\mathcal{B}'$ can compute

$$g^{ab} = \prod_{i=1}^{n^\star} \Big( \frac{\sigma^\star_{i,1}}{{\sigma^\star_{i,2}}^{K(m^\star_i)}} \Big)^{\frac{1}{J(m^\star_i)}} \ .$$

We are thus left with assessing the probability for $\mathcal{B}'$ to avoid the failure state during the game.

Since the choice of $j^\star$ is independent of $\mathcal{A}$'s view, we do have $X^\star = g^a$ with probability at least $\Pr[\neg E_1] \geq 1/q$. Regarding $E_2$ and $E_3$, since $\mathsf{Msg}_{j^\star} \not\subseteq \mathsf{Msg}^\star$, we know that there exists $k \in \{1, \ldots, n_{j^\star}\}$ such that $m_{j^\star,k} \in \mathsf{Msg}_{j^\star} \backslash \mathsf{Msg}^\star$. If we define the set $\overline{\mathsf{Msg}} = (\mathsf{Msg}_{j^\star} \cup \mathsf{Msg}^\star) \backslash \{m_{j^\star,k}\}$, a sufficient condition for the desirable event $\neg E_2 \wedge \neg E_3$ to come about is to have

$$J(m_{j^\star,k}) = 0 \quad \text{and} \quad J(m) \neq 0 \ \ \forall m \in \overline{\mathsf{Msg}} \ . \tag{3}$$

Since the cardinality of $\overline{\mathsf{Msg}}$ is at most $2n_{max} - 1$, the results of [47, 33] imply that condition (3) is satisfied with probability at least $1/(16 \cdot n_{max} \cdot (L+1))$. A lower bound on the probability that $\neg E_2 \wedge \neg E_3$ and that $\mathcal{B}'$ does not abort at the $j^\star$-th signing query is thus given by $1/(16 \cdot n_{max} \cdot (L+1))$. Taking into account the probability $\Pr[\neg E_1] \geq 1/q$, it comes that $\mathcal{B}'$ never aborts with probability at least $1/(16 \cdot q \cdot n_{max} \cdot (L+1))$. □

For the time being, the most efficient XRMA-secure structure-preserving signature based on simple assumptions is the construction of Abe *et al.* [4], where signatures consist of 8 group elements and the verifier has to compute one quadratic equation and three linear equations. Also, each one-time public key $X \in \mathbb{G}$ must be encoded as a triple $(g^x, g_1^x, g_2^x)$, for public elements $(g_1, g_2) \in \mathbb{G}^2$ and where $x = \log_g(X)$. Hence, the commitment $\vec{C}_X$ must come along with two other similar commitments. If the SPS scheme of [4] is plugged into our HH-AOS construction, a set $\{m_i\}_{i=1}^n$ of cardinality $n$ can be signed using $9n + 54$ group elements under the DLIN assumption (which implies the CDH assumption). Then, the bit-size of the signature amounts to $4608 \cdot n + 27648$ if each element of $\mathbb{G}$ has a 512-bit representation. In comparison with [14], our scheme only inflates signatures by a constant factor.

In Section 4 and Appendix E, we discuss further implications of the above result to the setting of ring signatures and ordinary (i.e., history-preserving) append-only signatures, where it implies constructions for arbitrarily long rings or sets.

## 4 Generic Identity-Based Ring Signatures

An identity-based ring signature is a tuple of efficient algorithms $(\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Sign}, \mathsf{Verify})$ with the following syntax.

$\mathsf{Setup}$ is a randomized algorithm that takes as input a security parameter $\lambda \in \mathbb{N}$ and outputs a master key pair $(\mathsf{msk}, \mathsf{mpk})$. $\mathsf{Keygen}$ is a possibly randomized algorithm that takes as input an identity $id$ and returns a private key $d_{id}$. Algorithm $\mathsf{Sign}$ takes as input a list of identities $\mathcal{R} = \{id_1, \ldots, id_r\}$, a private key $d_{id}$ for an identity such that $id \in \mathcal{R}$ and a message $M$ to output a signature $\sigma \leftarrow \mathsf{Sign}(\mathsf{mpk}, d_{id}, \mathcal{R}, M)$. Algorithm $\mathsf{Verify}$ inputs $\mathsf{mpk}$, a message $M$, a list of identities $\mathcal{R} = \{id_1, \ldots, id_r\}$ and a signature $\sigma$. It outputs 1 if $\sigma$ is deemed valid for the message $M$ and the ring $\mathcal{R}$ and 0 otherwise.

Identity-based ring signatures should satisfy two notions called *unforgeability* and *anonymity*, which can be formalized as below.

Bellare, Namprempre, and Neven [10] showed how to construct identity-based signatures from any signatures. Galindo, Herranz, and Kiltz [26] extended the generic construction of [10] to several kinds of identity-based signatures with special properties but their results do not carry over to the ring signature case. Boneh and Hamburg [18] gave a generic way to build short identity-based ring signatures from their spatial encryption primitive. However, their instantiations require to choose a maximal ring size when the system is set up. It thus remains interesting to provide a generic construction allowing for full security and rings of arbitrary size.

UNFORGEABILITY. This notion is formalized by a game where the challenger generates a master key pair $(\mathsf{mpk}, \mathsf{msk})$, where $\mathsf{mpk}$ is given to the adversary. Throughout the game, the adversary $\mathcal{A}$ is allowed to make private key queries: it chooses an identity $id$ and obtains a private key $d_{id} \leftarrow \mathsf{Keygen}(\mathsf{msk}, id)$. The adversary is also granted access to a signing oracle: at each query, it chooses a triple $(id, M, \mathcal{R})$ and the challenger returns $\perp$ if $id \notin \mathcal{R}$ and $\sigma \leftarrow \mathsf{Sign}(d_{id}, M, \mathcal{R})$ otherwise. Eventually, the adversary outputs a triple $(\sigma^\star, M^\star, R^\star)$ and wins if: (i) $\mathsf{Verify}(\mathsf{mpk}, M^\star, R^\star, \sigma^\star) = 1$; (ii) $\mathcal{A}$ did not invoke the signing oracle on a tuple $(id, M^\star, \mathcal{R}^\star)$ for any identity $id \in \mathcal{R}^\star$; (iii) No private key query was made for any $id \in \mathcal{R}^\star$. Note that this model allows the adversary to adaptively choose the ring $\mathcal{R}^\star$ of identities involved in the forgery. In the weaker model of selective-ring security, the adversary would be forced to declare $\mathcal{R}^\star$ at the very beginning of the game, before seeing $\mathsf{mpk}$.

FULL ANONYMITY. This property is defined via the following game. Initially, the challenger generates a pair $(\mathsf{mpk}, \mathsf{msk})$ and gives $\mathsf{mpk}$ and $\mathsf{msk}$ to the adversary $\mathcal{A}$. The adversary chooses a message $M$, a list of identities $\mathcal{R} = \{id_1, \ldots, id_r\}$, a pair of identities $(id_0, id_1)$ and two private keys $d_{id_0}$, $d_{id_1}$. If $\{id_0, id_1\} \not\subseteq \mathcal{R}$ or if $d_{id_0}, d_{id_1}$ are not valid private keys for the identities $id_0$ and $id_1$, respectively, the challenger returns $\perp$. Otherwise, it challenger flips a fair coin $d \stackrel{R}{\leftarrow} \{0,1\}$ and returns $\sigma \leftarrow \mathsf{Sign}(\mathsf{mpk}, d_{id_d}, M, \mathcal{R})$. The adversary eventually outputs a bit $d' \in \{0,1\}$ and wins if $d' = d$. As usual, the adversary's advantage is measured by the distance $\mathbf{Adv}(\mathcal{A}) := |\Pr[d' = d] - 1/2|$.

The above definition of anonymity could be strengthened (as done in [7]) by allowing the adversary to choose the random coins used by the challenger to generate $(\mathsf{mpk}, \mathsf{msk})$. Although the generic construction hereunder does not guarantee anonymity in the sense of this stronger definition, the specific instantiations obtained from our HH-AOS schemes can be proved secure in that sense.

We also remark that the above definition allows the adversary to come up with private keys $d_{id_0}$, $d_{id_1}$ of its own in the challenge phase. The anonymity definition of [7] is different and rather allows the adversary to choose the random coins used in the generation of $d_{id_0}$ and $d_{id_1}$. However, the definition of [7] still forces the challenger to generate $d_{id_0}$ and $d_{id_1}$ by running the legal key generation algorithm. In this aspect, our definition is stronger since it allows the adversary to choose any identity-based private keys $d_{id_0}$, $d_{id_1}$ that satisfy the key sanity check (we assume w.l.o.g. that valid private keys are recognizable) without necessarily being in the range of the private key generation algorithm. It is easy to verify that the scheme of [7] does not provide unconditional anonymity in the sense of the above definition. The reason is its use of groups $\mathbb{G}$ of composite order $N = p_1 p_2 p_3$ and the fact that signatures and private keys live in the subgroup of order $p_1$: if an unbounded adversary chooses $d_{id_0}$, $d_{id_1}$ so that $d_{id_0}$ does not have a component of order $p_2$ but $d_{id_1}$ does, this adversary can infer the challenger's bit by testing if the signature $\sigma$ has a component of order $p_2$.

Our generic construction thus provides the first fully secure schemes allowing for rings of arbitrary size while satisfying our definition of anonymity. Let $\Pi = (\mathsf{Keygen}, \mathsf{Sign}, \mathsf{SignDerive}, \mathsf{Verify})$ be a completely context-hiding and unforgeable HH-AOS scheme. Using $\Pi$, we can generically construct an identity-based ring signature as follows.

**Setup($\lambda$):** run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Pi.\mathsf{Keygen}(\lambda)$ and output $(\mathsf{msk}, \mathsf{mpk}) = (\mathsf{sk}, \mathsf{pk})$.

**Keygen(msk, $id$):** given $\mathsf{msk} = \mathsf{sk}$, compute and return $d_{id} \leftarrow \Pi.\mathsf{Sign}(\mathsf{sk}, \{0\|id\})$.

**Sign(mpk, $d_{id}$, $M$, $\mathcal{R}$):** return $\perp$ if $id \notin \mathcal{R}$. Otherwise, encode $\mathcal{R} = \{id_1, \ldots, id_r\}$ and the message $M$ as a set $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$ of cardinality $r + 1$. Then, use $d_{id}$ to compute $\sigma \leftarrow \Pi.\mathsf{SignDerive}(\mathsf{pk}, \{(d_{id}, \{0\|id\})\}, L)$, which is possible since $L$ is a superset of the singleton $\{0\|id\}$ by construction.

**Verify(mpk, $M$, $R$, $\sigma$):** given $\mathsf{mpk} = \mathsf{pk}$, the ring of identities $\mathcal{R} = \{id_1, \ldots, id_r\}$ and the message $M$, define the set $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$. Return 1 if $\Pi.\mathsf{Verify}(\mathsf{pk}, L, \sigma) = 1$ and 0 otherwise.

Note that, in order to guarantee the unforgeability of the scheme, the ring of identities $R$ must be appended to the actual message in the last element of $L$. Otherwise, the adversary would be able to introduce extra identities in the ring associated with any given signature.

**Theorem 2.** *The above identity-based ring signature scheme provides unforgeability against adaptive-ring attacks assuming that $\Pi$ is an unforgeable HH-AOS. Moreover, it provides full anonymity against unbounded adversaries if $\Pi$ is a completely context hiding HH-AOS scheme.*

*Proof.* The proof of unforgeability is straightforward as, given a ring signature forger, one can clearly construct a forger against the underlying HH-AOS. We thus focus on the anonymity property.

The proof of anonymity is also immediate. We consider a first game, called $\mathrm{Game}_0$, which is the actual attack game. We define $\mathrm{Game}_1$ to be identical to $\mathrm{Game}_0$ except that we modify the way to compute the challenge signature in the challenge phase. Namely, instead of computing the challenge signature as $\sigma \leftarrow \Pi.\mathsf{SignDerive}(\mathsf{pk}, \{(d_{id_d}, id_d)\}, L)$, the challenger computes a new signature $\sigma \leftarrow \Pi.\mathsf{Sign}(\mathsf{sk}, L)$ on the set $L$. The complete context-hiding property guarantees that $\mathcal{A}$'s view will not be affected by this change since $\sigma$ has exactly the same distribution in both games. However, in $\mathrm{Game}_1$, the challenge signature $\sigma$ does not depend on the adversary's secret bit $d \in_R \{0, 1\}$, which is thus independent of the adversary's view. □

## Acknowledgments

We thank the anonymous reviewers for useful comments. The first author's work was supported in part by the "Programme Avenir Lyon Saint-Etienne de l'Université de Lyon" in the framework of the programme "Inverstissements d'Avenir" (ANR-11-IDEX-0007). The last author's work was supported by the ERC grant CryptoCloud.

## References

1. M. Abe, K. Haralambiev, M. Ohkubo. Signing on elements in bilinear groups for modular protocol design. Cryptology ePrint Archive: Report 2010/133, 2010.
2. M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. O. Structure-preserving signatures and commitments to group elements. In *CRYPTO '10*, LNCS 6223, Springer, 2010.

3. M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, M. Ohkubo. Constant-Size Structure-Preserving Signatures: Generic Constructions and Simple Assumptions. In *Asiacrypt'12*, *LNCS* 7658, pp. 4–24, 2012.
4. M. Abe, B. David, M. Kohlweiss, R. Nishimaki, M. Ohkubo. Tagged one-time signatures: Tight security and optimal tag size. In *PKC '13*, LNCS 7778, Springer, 2013.
5. J.-H. Ahn, D. Boneh, J. Camenisch, S. H, a. shelat, B. Waters. Computing on authenticated data. In *TCC '12*, LNCS 7194, Springer, 2012.
6. N. Attrapadung, B. Libert, T. Peters. Computing on authenticated data: New privacy definitions and constructions. In *ASIACRYPT '12*, LNCS 7658, Springer, 2012.
7. M.-H. Au, J. Liu, W. Susilo, J. Zhou. Realizing fully secure unrestricted ID-based ring signature in the standard model from HIBE. *IEEE Trans. Information Forensics and Security* **8**(12), 2013.
8. S. Bajaj, R. Sion. HIFS: History independence for file systems. In *ACM-CCS '13*, ACM Press, 2013.
9. M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, H. Shacham. Randomizable proofs and delegatable anonymous credentials. In *CRYPTO '09*, LNCS 5677, Springer, 2009.
10. M. Bellare, C. Namprempre, G. Neven. Security proofs for identity-based identification and signature schemes. *J. Cryptology* **22**(1), 2009. Preliminary version in *EUROCRYPT '04*, LNCS 3027, Springer, 2004.
11. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS '93*, ACM Press, 1993.
12. M. Bellare and P. Rogaway. The exact security of digital signatures – How to sign with RSA and Rabin. In *EUROCRYPT '96*, LNCS 1070, Springer, 1996.
13. A. Bender, J. Katz, R. Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology* **22**(1), 2009. Extended abstract in *TCC '06*, LNCS 3876, Springer, 2006.
14. J. Bethencourt, D. Boneh, B. Waters. Cryptographic methods for storing ballots on a voting machine. In *NDSS '07*, Internet Society, 2007.
15. D. Boneh, X. Boyen, E.-J. Goh. Hierarchical identity-based encryption with constant size ciphertext. In *EUROCRYPT '05*, LNCS 3494, Springer, 2005.
16. D. Boneh, X. Boyen, H. Shacham. Short group signatures. In *CRYPTO '04*, LNCS 3152, 2004.
17. D. Boneh, M. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.* **32**(3). Earlier version in *CRYPTO '01*, LNCS 2139, Springer, 2001.
18. D. Boneh, M. Hamburg. Generalized identity based and broadcast encryption schemes. In *ASIACRYPT '08*, LNCS 5350, Springer, 2008.
19. D. Boneh, B. Lynn, H. Shacham. Short signatures from the Weil pairing. In *ASIACRYPT '01*, LNCS 2248, Springer, 2001.
20. D. Boneh, C. Gentry, B. Lynn, H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT '03*, LNCS 2656, Springer, 2003.
21. R. Canetti, O. Goldreich, S. Halevi. The random oracle methodology, revisited. In *STOC '98*, ACM Press, 1998.
22. N. Chandran, J. Groth, A. Sahai. Ring signatures of sub-linear size without random oracles. In *ICALP '07*, LNCS 4596, Springer, 2007.
23. M. Chase, M. Kohlweiss, A. Lysyanskaya, S. Meiklejohn. Malleable proof systems and applications. In *EUROCRYPT '12*, LNCS 7237, Springer, 2012.
24. Y. Desmedt. Computer security by redefining what a computer is. In *New Security Paradigms Workshop (NSPW '93)*, 1993.
25. E. Freire, D. Hofheinz, K. Paterson, C. Striecks. Programmable hash functions in the multilinear setting. In *CRYPTO '13*, LNCS 8043, Springer, 2013.
26. D. Galindo, J. Herranz, E. Kiltz. On the generic construction of identity-based signatures with additional properties. In *ASIACRYPT '06*, LNCS 4284, Springer, 2006.
27. S. Garg, C. Gentry, S. Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT '13*, LNCS 7881, Springer, 2013.
28. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS '13*, IEEE Computer Society, 2013.
29. M. Gerbush, A. Lewko, A. O'Neill, B. Waters. Dual form signatures: An approach for proving security from static assumptions. In *ASIACRYPT '12*, LNCS 7658, Springer, 2012.
30. C. Gentry, A. Silverberg. Hierarchical ID-based cryptography. In *ASIACRYPT '02*, LNCS 2501, 2002.
31. J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT '06*, LNCS 4284, Springer, 2006.
32. J. Groth, A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT '08*, LNCS 4965, Springer, 2008.

33. D. Hofheinz, E. Kiltz. Programmable hash functions and their applications. In *CRYPTO '08*, LNCS 5157, Springer, 2008.
34. S. Hohenberger, A. Sahai, B. Waters. Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. In *CRYPTO '13*, LNCS 8043, Springer, 2013.
35. S. Hohenberger, A. Sahai, B. Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In *EUROCRYPT '14*, LNCS 8441, Springer, 2014.
36. R. Johnson, D. Molnar, D. Song, D. Wagner. Homomorphic signature schemes. In *CT-RSA '02*, LNCS 2271, Springer, 2002.
37. E. Kiltz, A. Mityagin, S. Panjwani, B. Raghavan. Append-only signatures. In *ICALP '05*, LNCS 3580, Springer, 2005.
38. A. Lewko, B. Waters. Unbounded HIBE and attribute-based encryption. In *EUROCRYPT '11*, LNCS 6632, Springer, 2011.
39. S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, B. Waters. Sequential aggregate signatures and multisignatures without random oracles. In *EUROCRYPT '06*, LNCS 4004, 2006.
40. D. Micciancio. Oblivious data structures: Applications to cryptography. In *STOC' 97*, ACM Press, 1997.
41. D. Molnar, T. Kohno, N. Sastry, D. Wagner. Tamper-evident, history-independent, subliminal-free data structures on PROM storage –or– How to store ballots on a voting machine. In *S&P '06*, IEEE Computer Society, 2006.
42. T. Moran, M. Naor, G. Segev. Deterministic history-independent strategies for storing information on write-once memories. In *ICALP '07*, LNCS 4596, Springer, 2007.
43. M. Naor, V. Teague. Anti-persistence: History independent data structures. In *STOC '01*, ACM Press, 2001.
44. R. Rivest, A. Shamir, Y. Tauman. How to leak a secret. In *ASIACRYPT '01*, LNCS 2248, Springer, 2001.
45. H. Shacham, B. Waters. Efficient ring signatures without random oracles. In *PKC '07*, LNCS 4450, Springer, 2007.
46. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO '84*, LNCS 196, Springer, 1984.
47. B. Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT '05*, LNCS 3494, Springer, 2005.
48. B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO '09*, LNCS 5677, Springer, 2009.

## A Groth-Sahai Proof Systems

In [32], Groth and Sahai described efficient non-interactive witness indistinguishable (NIWI) proof systems of which one instantiation relies on the DLIN assumption. This instantiation uses prime order groups and a common reference string containing three vectors $\vec{f_1}, \vec{f_2}, \vec{f_3} \in \mathbb{G}^3$, where $\vec{f_1} = (f_1, 1, g)$, $\vec{f_2} = (1, f_2, g)$ for some $f_1, f_2 \in \mathbb{G}$. To commit to a group element $X \in \mathbb{G}$, the prover chooses $r, s, t \xleftarrow{R} \mathbb{Z}_p^*$ and computes $\vec{C} = (1, 1, X) \cdot \vec{f_1}^r \cdot \vec{f_2}^s \cdot \vec{f_3}^t$. On a perfectly sound common reference string, we have $\vec{f_3} = \vec{f_1}^{\xi_1} \cdot \vec{f_2}^{\xi_2}$ where $\xi_1, \xi_2 \in \mathbb{Z}_p^*$. Commitments $\vec{C} = (f_1^{r+\xi_1 t}, f_2^{s+\xi_2 t}, X \cdot g^{r+s+t(\xi_1+\xi_2)})$ are extractable as their distribution coincides with that of Boneh-Boyen-Shacham (BBS) ciphertexts [16] and the committed $X$ can be extracted using $\beta_1 = \log_g(f_1)$, $\beta_2 = \log_g(f_2)$. In the witness indistinguishability (WI) setting, the vector $\vec{f_3}$ is chosen outside the span of $(\vec{f_1}, \vec{f_2})$, so that $\vec{C}$ is a perfectly hiding commitment. Under the DLIN assumption, the two kinds of CRS can be exchanged for one another without the adversary noticing.

To convince the verifier that committed variables satisfy a set of relations, the prover computes one commitment per variable and one proof element per equation. Such NIWI proofs can be efficiently generated for pairing-product equations, which are relations of the type

$$\prod_{i=1}^{n} e(\mathcal{A}_i, \mathcal{X}_i) \cdot \prod_{i=1}^{n} \cdot \prod_{j=1}^{n} e(\mathcal{X}_i, \mathcal{X}_j)^{a_{ij}} = t_T \,, \tag{4}$$

for variables $\mathcal{X}_1, \ldots, \mathcal{X}_n \in \mathbb{G}$ and constants $t_T \in \mathbb{G}_T$, $\mathcal{A}_1, \ldots, \mathcal{A}_n \in \mathbb{G}$, $a_{ij} \in \mathbb{Z}_p$, for $i, j \in \{1, \ldots, n\}$.

In pairing-product equations, proving a quadratic equation requires 9 group elements. Linear equations (i.e., where $a_{ij} = 0$ for all $i, j$ in Eq. (4)) are slightly more economical to prove as they only cost 3 group elements each.

In [9], Belenkiy *et al.* showed that Groth-Sahai proofs are perfectly randomizable. Given commitments $\{\vec{C}_{\mathcal{X}_i}\}_{i=1}^n$ and a NIWI proof $\vec{\pi}_{\text{PPE}}$ that committed $\{\mathcal{X}\}_{i=1}^n$ satisfy (4), anyone can publicly compute re-randomized commitments $\{\vec{C}_{\mathcal{X}'_i}\}_{i=1}^n$ and a re-randomized proof $\vec{\pi}'_{\text{PPE}}$ of the same statement. Moreover, $\{\vec{C}_{\mathcal{X}'_i}\}_{i=1}^n$ and $\vec{\pi}'_{\text{PPE}}$ are distributed as freshly generated commitments and proof. This property was used in, *e.g.*, [23].

## B Structure-Preserving Signatures

In many privacy-enhancing cryptographic protocols, it is convenient to have signature schemes allowing to sign elements from the domain group $\mathbb{G}$ of a bilinear map without destroying their algebraic structure (in particular, without hashing them first).

The first example of *structure-preserving* signature was proposed by Groth [31] (although the "structure-preserving" terminology only appeared in [1, 2]) but, due to very large signatures, it was more a feasibility result than a practical solution. Abe, Haralambiev and Ohkubo [1, 2] (AHO) described a very efficient structure-preserving signature we remind the reader in this section.

The description below assumes public parameters $\mathsf{pp} = \big((\mathbb{G}, \mathbb{G}_T),\ g\big)$ consisting of bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$, where $\lambda \in \mathbb{N}$ and a generator $g \in \mathbb{G}$.

**Keygen($\mathsf{pp}, n$):** given an upper bound $n \in \mathbb{N}$ on the number of group elements per signed message, choose generators $G_r, H_r \xleftarrow{R} \mathbb{G}$. Pick $\gamma_z, \delta_z \xleftarrow{R} \mathbb{Z}_p$ and $\gamma_i, \delta_i \xleftarrow{R} \mathbb{Z}_p$, for $i = 1$ to $n$. Then, compute $G_z = G_r^{\gamma_z}$, $H_z = H_r^{\delta_z}$ and $G_i = G_r^{\gamma_i}$, $H_i = H_r^{\delta_i}$ for each $i \in \{1, \ldots, n\}$. Finally, choose $\alpha_a, \alpha_b \xleftarrow{R} \mathbb{Z}_p$ and define $A = e(G_r, g^{\alpha_a})$ and $B = e(H_r, g^{\alpha_b})$. The public key is defined to be

$$pk = \big(G_r,\ H_r,\ G_z,\ H_z,\ \{G_i, H_i\}_{i=1}^n,\ A,\ B\big) \in \mathbb{G}^{2n+4} \times \mathbb{G}_T^2$$

while the private key is $sk = \big(\alpha_a, \alpha_b, \gamma_z, \delta_z, \{\gamma_i, \delta_i\}_{i=1}^n\big)$.

**Sign($sk, (M_1, \ldots, M_n)$):** to sign a vector $(M_1, \ldots, M_n) \in \mathbb{G}^n$ using $sk$, choose $\zeta, \rho_a, \rho_b, \omega_a, \omega_b \xleftarrow{R} \mathbb{Z}_p$ and compute $\theta_1 = g^\zeta$ as well as

$$\theta_2 = g^{\rho_a - \gamma_z \zeta} \cdot \prod_{i=1}^n M_i^{-\gamma_i}\,, \qquad \theta_3 = G_r^{\omega_a}\,, \qquad \theta_4 = g^{(\alpha_a - \rho_a)/\omega_a}\,,$$

$$\theta_5 = g^{\rho_b - \delta_z \zeta} \cdot \prod_{i=1}^n M_i^{-\delta_i}\,, \qquad \theta_6 = H_r^{\omega_b}\,, \qquad \theta_7 = g^{(\alpha_b - \rho_b)/\omega_b}\ .$$

The signature consists of $\sigma = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7) \in \mathbb{G}^7$.

**Verify($pk, (M_1, \ldots, M_n), \sigma$):** given $\sigma = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7)$, return 1 iff

$$A = e(G_z, \theta_1) \cdot e(G_r, \theta_2) \cdot e(\theta_3, \theta_4) \cdot \prod_{i=1}^n e(G_i, M_i)\,,$$

$$B = e(H_z, \theta_1) \cdot e(H_r, \theta_5) \cdot e(\theta_6, \theta_7) \cdot \prod_{i=1}^n e(H_i, M_i)\ .$$

The scheme was proved [1, 2] existentially unforgeable under chosen-message attacks under the $q$-SFP assumption, where $q$ is the number of signing queries.

As showed in [1, 2], signature components $\{\theta_i\}_{i=2}^7$ can be publicly randomized to obtain a different signature $\{\theta'_i\}_{i=1}^7 \leftarrow \mathsf{ReRand}(pk, \sigma)$ on $(M_1, \ldots, M_n)$. After randomization, we have $\theta'_1 = \theta_1$ while $\{\theta'_i\}_{i=2}^7$ are uniformly distributed among the values $(\theta_2, \ldots, \theta_7)$ such that the equalities

$$e(G_r, \theta'_2) \cdot e(\theta'_3, \theta'_4) = e(G_r, \theta_2) \cdot e(\theta_3, \theta_4)$$
$$e(H_r, \theta'_5) \cdot e(\theta'_6, \theta'_7) = e(H_r, \theta_5) \cdot e(\theta_6, \theta_7)$$

hold. This re-randomization is performed by choosing $\varrho_2, \varrho_5, \mu, \nu \xleftarrow{R} \mathbb{Z}_p$ and computing

$$\begin{aligned} \theta'_2 &= \theta_2 \cdot \theta_4^{\varrho_2}\,, \quad \theta'_3 = (\theta_3 \cdot G_r^{-\varrho_2})^{1/\mu}\,, \quad \theta'_4 = \theta_4^{\mu}\,, \\ \theta'_5 &= \theta_5 \cdot \theta_7^{\varrho_5}\,, \quad \theta'_6 = (\theta_6 \cdot H_r^{-\varrho_5})^{1/\nu}\,, \quad \theta'_7 = \theta_7^{\nu}\ . \end{aligned} \tag{5}$$

As a result, $\{\theta'_i\}_{i\in\{3,4,6,7\}}$ are statistically independent of the message and other signature components. This implies that, in privacy-preserving protocols, re-randomized $\{\theta'_i\}_{i\in\{3,4,6,7\}}$ can be safely given in the clear as long as $(M_1, \ldots, M_n)$ and $\{\theta'_i\}_{i\in\{1,2,5\}}$ are given in committed form.

## C The BBW Construction

To our knowledge, the only known construction of subliminal-free HH-AOS is the second construction given by Bethencourt, Boneh and Waters [14], which builds on the aggregate signature of Boneh, Gentry, Lynn and Shacham [20]. In the recent framework [5] of homomorphic signatures, the scheme of [14] can be described as follows.

**Keygen($\lambda$):** given a security parameter $\lambda \in \mathbb{N}$, choose bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$ with a generator $g \xleftarrow{R} \mathbb{G}$. Choose a cryptographic hash function $H : \{0,1\} \to \mathbb{G}$, which is modeled as a random oracle in the security analysis. Choose $\alpha \xleftarrow{R} \mathbb{Z}_p$ and define the private key to be $\mathsf{sk} := g^\alpha$ and the public key consists of

$$\mathsf{pk} := \big((\mathbb{G}, \mathbb{G}_T),\ g,\ e(g,g)^\alpha,\ H\big)\ .$$

**Sign(sk, Msg):** parse the private key as $\mathsf{sk} = g^\alpha$ and the message as $\mathsf{Msg} = \{m_i\}_{i=1}^n$, where $m_i \in \{0,1\}^*$ for each $i$. If $n = 0$ (i.e., if $\mathsf{Msg} = \emptyset$), return $\mathsf{sk}$. Otherwise, do the following.

1. Choose $r_1, \ldots, r_n \xleftarrow{R} \mathbb{Z}_p$ and compute

$$\sigma_0 = g^\alpha \cdot \prod_{i=1}^n H(m_i)^{r_i}\,, \qquad\qquad \sigma_i = g^{r_i}\,,$$

   where the messages are indexed in random order.
2. Return the signature

$$\sigma = \big(\sigma_0, \{(m_i, \sigma_i)\}_{i=1}^n\big)\ . \tag{6}$$

**SignDerive(pk, Msg, Msg′):** parse $\mathsf{Msg}$ as $\{m_i\}_{i=1}^n$ and $\mathsf{Msg}'$ as $\mathsf{Msg}' = \mathsf{Msg} \cup \{m'\}$, for some $m' \in \{0,1\}^*$ (if $\mathsf{Msg}'$ cannot be parsed like this, return $\perp$). Then, parse $\sigma$ as per (6). Otherwise,

1. Choose $r'_1, \ldots, r'_n, r' \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and compute

$$\sigma'_0 = \sigma_0 \cdot \prod_{i=1}^{n} H(m_i)^{r'_i}, \qquad \sigma'_i = \sigma_i \cdot g^{r'_i} \ .$$

2. Choose $r'_{n+1} \stackrel{R}{\leftarrow} \mathbb{Z}_p$ and compute

$$\sigma''_0 = \sigma'_0 \cdot H(m')^{r'}, \qquad \sigma''_{n+1} = g^{r'} \ .$$

3. Return the derived signature

$$\sigma = \left(\sigma''_0, \{(m_i, \sigma_i)\}_{i=1}^{n+1}\right), \tag{7}$$

after having re-arranged the pairs $\{(m_i, \sigma_i)\}_{i=1}^{n+1}$ so as to have $\{m_i\}_{i=1}^{n+1}$ arranged in a predetermined lexicographical order.

**Verify(pk, Msg, $\sigma$):** given pk, a signature $\sigma$ and a message $\mathsf{Msg} = \{m_i\}_{i=1}^{n}$, parse $\sigma$ as in (6). Return 1 if and only if

$$e(\sigma_0, g) = e(g, g)^{\alpha} \cdot \prod_{i=1}^{n} e(H(m_i), \sigma_i) \ .$$

In [14], the above scheme was proved unforgeable under the Diffie-Hellman assumption in the random oracle model [11].

Proving the security of the scheme in the standard model seems rather difficult as it seemingly requires to eliminate the random oracle from the BLS and BGLS (aggregate) signatures [19, 20].

It is tempting to believe that the sequential aggregate signature of Lu *et al.* [39] can supersede the BGLS aggregate signature so as to work in the standard model. One interesting property of the scheme of [39] is that the verifier does not learn anything about the aggregation order. Therefore one could *a priori* hope to solve the problem by having the SignDerive algorithm generate a new signer's public key to be included in the derived signature after having run the sequential aggregation algorithm.

Unfortunately, the resulting HH-AOS is insecure because the sequential aggregate signature of [39] is only secure in the known-secret-key model (i.e., the adversary cannot come up with public keys of its own without also revealing the private key). In more details, the scheme of [39] uses public keys of the form

$$pk_i = \left(e(g, g)^{\alpha_i}, (h_{i,0}, \ldots, h_{i,L})\right) \ . \tag{8}$$

If $(\sigma_1, \sigma_2) = (g^{\sum_{i=1}^{n} \alpha_i} \cdot \prod_{i=1}^{n} (h_{i,0} \cdot \prod_{j=1}^{L} h_{i,j}^{m_{i,j}})^r, g^r)$ denotes the aggregate-so-far signature on the messages $\{m_i = m_{i,1}, \ldots, m_{i,L}\}_{i=1}^{n}$ for public keys $\{pk_i\}_{i=1}^{n}$, the $(n+1)$-th signer can compute

$$(\sigma'_1, \sigma'_2) = \left(\sigma_1 \cdot \sigma_2^{\log_g(h_{n+1,0}) + \sum_{j=1}^{L} \log_g(h_{n+1,j}) \cdot m_{n+1,j}}, \sigma_2\right)$$

before re-randomizing $(\sigma'_1, \sigma'_2)$. If we naively try to construct a HH-AOS using the latter scheme, the following problem occurs. Suppose that the HH-AOS signer has a public key of the form (8) and signs the singleton $\{m_1 = m_{1,1} \ldots m_{1,L}\}$ by generating a signature $\left(g^{\alpha_1} \cdot (h_{1,0} \cdot \prod_{j=1}^{L} h_{1,j}^{m_{1,j}})^r, g^r\right)$. Then, the adversary can generate his "public key" as $e(g, g)^{\alpha_2} = e(g, g)^{\omega} / e(g, g)^{\alpha_1}$ for a random

$\omega \in \mathbb{Z}_p$ of his choice. Hence, using $g^{\alpha} = g^{\alpha_1+\alpha_2}$, the second signer can cheat and sign any arbitrary set that does not contain the original message $m_1 = m_{1,1} \dots m_{1,L}$.

In order to eliminate the need for random oracles, one plausible approach is to use multi-linear maps [27] in order to build aggregate signatures in the standard model [34, 35] via standard model instantiations [25, 34] of BLS signatures [19]. For now, this would only come at the price of very large keys.

## D A More Efficient Scheme from the $q$-SFP Assumption

This section shows how to slightly improve the efficiency of the scheme using a different structure-preserving signature, due to Abe, Haralambiev, and Ohkubo [1], which features randomizable signatures (the description of this system is given in Appendix B). Thanks to the latter property, certain signature components can be given in the clear in such a way that NIWI arguments are only needed for linear pairing product equations.

Our second construction also relies on the $q$-SFP problem, the generic hardness of which was demonstrated by Abe *et al.* [1].

**Definition 11 ([1]).** *In a group $\mathbb{G}$, the **$q$-Simultaneous Flexible Pairing Problem** ($q$-SFP) is, given elements $\big(g_z, h_z, g_r, h_r, a, \tilde{a}, b, \tilde{b}\big) \in \mathbb{G}^8$ as well as $q$ tuples $(z_j, r_j, s_j, t_j, u_j, v_j, w_j) \in \mathbb{G}^7$ such that*

$$e(a, \tilde{a}) = e(g_z, z_j) \cdot e(g_r, r_j) \cdot e(s_j, t_j)\,, \qquad e(b, \tilde{b}) = e(h_z, z_j) \cdot e(h_r, u_j) \cdot e(v_j, w_j)\,,$$

*to find a new tuple $(z^{\star}, r^{\star}, s^{\star}, t^{\star}, u^{\star}, v^{\star}, w^{\star}) \in \mathbb{G}^7$ satisfying the above equalities and such that $z^{\star} \notin \{1_{\mathbb{G}}, z_1, \dots, z_q\}$.*

At the expense of relying on the stronger $q$-SFP assumption, we thus shorten signatures by 29 group elements, which is really noticeable for small sets.

Note that this specific SPS scheme was proved existentially unforgeable under regular chosen-message attacks. Under the $q$-SFP assumption, it thus satisfies a stronger security notion than RMA security but we do not rely on this property here.

**Keygen($\lambda$):**

1. Choose bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ or prime order $p > 2^{\lambda}$ with a generator $g \xleftarrow{R} \mathbb{G}$. Then, generate a Groth-Sahai CRS $\mathbf{f} = (\vec{f_1}, \vec{f_2}, \vec{f_3})$ for the perfect witness indistinguishability setting. Namely, choose $\vec{f_1} = (f_1, 1, g)$, $\vec{f_2} = (1, f_2, g)$, and $\vec{f_3} = \vec{f_1}^{\xi_1} \cdot \vec{f_2}^{\xi_2} \cdot (1, 1, g)^{-1}$, with $f_1, f_2 \xleftarrow{R} \mathbb{G}$, $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p$.
2. Generate a key pair $(sk_{\mathsf{aho}}, pk_{\mathsf{aho}})$ for the AHO signature in order to sign messages consisting of a single group element. This key pair are

$$pk_{\mathsf{aho}} = \big(G_r,\ H_r,\ G_z = G_r^{\gamma_z},\ H_z = H_r^{\delta_z},\ G_1 = G_r^{\gamma_1}, H_1 = H_r^{\delta_1},\ A,\ B\big)$$

   and $sk_{\mathsf{aho}} = \big(\alpha_a, \alpha_b, \gamma_z, \delta_z, \gamma_1, \delta_1\big)$.
3. Generate public parameters for a Waters hash function. Namely, choose a set of generators $(h_0, h_1, \dots, h_L) \xleftarrow{R} \mathbb{G}^{L+1}$. These are used to define the function $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ such that any string $m = m[1] \dots m[L] \in \{0,1\}^L$ is mapped to $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{i=1}^{L} h_i^{m[i]}$.

The public key is defined to be $\mathsf{pk} := \big((\mathbb{G}, \mathbb{G}_T), g,\ \mathbf{f},\ pk_{\mathsf{aho}},\ \{h_i\}_{i=0}^{L}\big)$ and the private key is $\mathsf{sk} = sk_{\mathsf{aho}}$. The public key defines $\Sigma = \{0,1\}^L$.

**Sign(sk, Msg):** on input of a message $\mathsf{Msg} = \{m_i\}_{i=1}^{n}$, where $m_i \in \{0,1\}^L$ for each $i$, and the private key $\mathsf{sk} = sk_{\mathsf{aho}}$, do the following.

1. Generate a fresh one-time public key $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$. Generate a Groth-Sahai commitment $\vec{C}_X = (1,1,X) \cdot \vec{f}_1^{\,r_X} \cdot \vec{f}_2^{\,s_X} \cdot \vec{f}_3^{\,t_X}$, with $r_X, s_X, t_X \xleftarrow{R} \mathbb{Z}_p$.
2. Generate an AHO signature $(\theta_1, \ldots, \theta_7) \in \mathbb{G}^7$ on the group element $X \in \mathbb{G}$. Then, for each $j \in \{1,2,5\}$, generate Groth-Sahai commitments $\vec{C}_{\theta_j} = (1,1,\theta_j) \cdot \vec{f}_1^{\,r_{\theta_j}} \cdot \vec{f}_2^{\,s_{\theta_j}} \cdot \vec{f}_3^{\,t_{\theta_j}}$. Finally, generate NIWI arguments $\vec{\pi}_{\mathsf{aho},1}, \vec{\pi}_{\mathsf{aho},2} \in \mathbb{G}^3$ that committed variables $(X, \theta_1, \theta_2, \theta_5)$ satisfy
$$\begin{aligned} A \cdot e(\theta_3, \theta_4)^{-1} &= e(G_z, \theta_1) \cdot e(G_r, \theta_2) \cdot e(G_1, X) \\ B \cdot e(\theta_6, \theta_7)^{-1} &= e(H_z, \theta_1) \cdot e(H_r, \theta_5) \cdot e(H_1, X) \end{aligned} \ . \tag{9}$$
These proofs are obtained as
$$\begin{aligned} \vec{\pi}_{\mathsf{aho},1} &= \big(G_z^{-r_{\theta_1}} G_r^{-r_{\theta_2}} G_1^{-r_X},\ G_z^{-s_{\theta_1}} G_r^{-s_{\theta_2}} G_1^{-s_X},\ G_z^{-t_{\theta_1}} G_r^{-t_{\theta_2}} G_1^{-t_X}\big)\,, \\ \vec{\pi}_{\mathsf{aho},2} &= \big(H_z^{-r_{\theta_1}} H_r^{-r_{\theta_5}} H_1^{-r_X},\ H_z^{-s_{\theta_1}} H_r^{-s_{\theta_5}} H_1^{-s_X},\ H_z^{-t_{\theta_1}} H_r^{-t_{\theta_5}} H_1^{-t_X}\big)\ . \end{aligned}$$
3. Choose $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ subject to the constraint $\sum_{i=1}^{n} \omega_i = x$. Then, for $i = 1$ to $n$, compute
$$\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}\,, \qquad\qquad \sigma_{i,2} = g^{\omega_i}\,,$$
where the messages are indexed in some pre-determined lexicographical order. Then, compute commitments $\vec{C}_{\sigma_{i,1}} = (1,1,\sigma_{i,1}) \cdot \vec{f}_1^{\,r_{i,1}} \cdot \vec{f}_2^{\,s_{i,1}} \cdot \vec{f}_3^{\,t_{i,1}}$, and $\vec{C}_{\sigma_{i,2}} = (1,1,\sigma_{i,2}) \cdot \vec{f}_1^{\,r_{i,2}} \cdot \vec{f}_2^{\,s_{i,2}} \cdot \vec{f}_3^{\,t_{i,2}}$ to $\{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n}$. Then, generate a NIWI argument $\vec{\pi}_i$ that $e(\sigma_{i,1}, g) = e(H_{\mathbb{G}}(m_i), \sigma_{i,2})$. This argument is obtained as
$$\vec{\pi}_i = (\pi_{i,1}, \pi_{i,2}, \pi_{i,3}) = \big(g^{r_{i,1}} \cdot H_{\mathbb{G}}(m_i)^{-r_{i,2}},\ g^{s_{i,1}} \cdot H_{\mathbb{G}}(m_i)^{-s_{i,2}}, g^{t_{i,1}} \cdot H_{\mathbb{G}}(m_i)^{-t_{i,2}}\big)$$
and satisfies
$$E\big(g, \vec{C}_{\sigma_{i,1}}\big) = E\big(H_{\mathbb{G}}(m_i), \vec{C}_{\sigma_{i,2}}\big) \cdot E\big(\pi_{i,1}, \vec{f}_1\big) \cdot E\big(\pi_{i,2}, \vec{f}_2\big) \cdot E\big(\pi_{i,3}, \vec{f}_3\big)\ . \tag{10}$$
4. Finally, generate a NIWI proof $\vec{\pi}_{sum}$ that $X = \prod_{i=1}^{n} \sigma_{i,2}$. This proof is obtained as
$$\vec{\pi}_{sum} = (\pi_{s,1}, \pi_{s,2}, \pi_{s,3}) = \big(g^{r_X - \sum_{i=1}^{n} r_{i,2}},\ g^{s_X - \sum_{i=1}^{n} s_{i,2}},\ g^{t_X - \sum_{i=1}^{n} t_{i,2}}\big)$$
which satisfies
$$E\big(g, \vec{C}_X \cdot \prod_{i=1}^{n} \vec{C}_{\sigma_{i,2}}^{-1}\big) = E\big(\pi_{s,1}, \vec{f}_1\big) \cdot E\big(\pi_{s,2}, \vec{f}_2\big) \cdot E\big(\pi_{s,3}, \vec{f}_3\big)\ . \tag{11}$$

Return the signature
$$\sigma = \Big(\vec{C}_X, \{\vec{C}_{\theta_j}\}_{j\in\{1,2,5\}}, \{\theta_j\}_{j\in\{3,4,6,7\}}, \vec{\pi}_{\mathsf{aho},1}, \vec{\pi}_{\mathsf{aho},2}, \{(m_i, \vec{C}_{\sigma_{i,1}}, \vec{C}_{\sigma_{i,2}}, \vec{\pi}_i)\}_{i=1}^{n},\ \vec{\pi}_{sum}\Big)\ . \tag{12}$$

**SignDerive(pk, ($\sigma$, Msg), Msg′):** given $\mathsf{Msg} = \{m_i\}_{i=1}^n$, return $\perp$ if $\mathsf{Msg}' \neq \mathsf{Msg} \cup \{m'\}$ for some $m' \in \Sigma$. Otherwise, parse $\sigma$ as in (12) and do the following.

1. Re-randomize the commitment $\vec{C}_X$ and the proofs $\vec{\pi}_{\mathsf{aho},1}$, $\vec{\pi}_{\mathsf{aho},2}$, $\vec{\pi}_{sum}$. Let $\vec{C}''_X$, $\vec{\pi}'_{\mathsf{aho},1}$, $\vec{\pi}'_{\mathsf{aho},2}$ and $\vec{\pi}'_{sum}$ be the randomized commitment and proofs. Note that, in all of these commitments and proofs, the underlying exponents $(r_X, s_X, t_X)$ have been updated.
2. Re-randomize $\{\vec{C}_{\theta_j}\}_{j\in\{2,5\}}$ and $\{\theta_j\}_{j\in\{3,4,6,7\}}$ by choosing $\rho_2, \rho_5, \mu, \nu$ and computing

$$\begin{aligned} \vec{C}'_{\theta_2} &= \vec{C}_{\theta_2} \cdot (1, 1, \theta_4^{\rho_2}), & \theta'_3 &= \big(\theta_3 \cdot G_r^{-\rho_2}\big)^{1/\mu}, & \theta'_4 &= \theta_4^{\mu}, \\ \vec{C}'_{\theta_5} &= \vec{C}_{\theta_5} \cdot (1, 1, \theta_7^{\rho_5}), & \theta'_6 &= \big(\theta_6 \cdot H_r^{-\rho_5}\big)^{1/\nu}, & \theta'_7 &= \theta_7^{\nu}\ . \end{aligned}$$

   Note that, although the committed values inside $\vec{C}'_{\theta_2}, \vec{C}'_{\theta_5}$ have been updated, $\pi'_{\mathsf{aho},1}, \pi'_{\mathsf{aho},2}$ are still valid arguments for the new committed values. Then, compute $\{\vec{C}''_{\theta_j}\}_{j\in\{1,2,5\}}$ by re-randomizing[9] the commitments $\vec{C}_{\theta_1}$, $\{\vec{C}'_{\theta_j}\}_{j\in\{2,5\}}$ and the arguments $\pi'_{\mathsf{aho},1}, \pi'_{\mathsf{aho},2}$ again. Let $\vec{\pi}''_{\mathsf{aho},1}, \vec{\pi}''_{\mathsf{aho},2}$ denote the re-randomized arguments.
3. Choose $\omega'_1, \ldots, \omega'_{n+1} \xleftarrow{R} \mathbb{Z}_p$ subject to the constraint $\sum_{i=1}^{n+1} \omega'_i = 0$. For $i = 1$ to $n$, compute

$$\vec{C}'_{\sigma_{i,1}} = (1, 1, H(m_i)^{\omega'_i}) \cdot \vec{C}_{\sigma_{i,1}}, \qquad \vec{C}'_{\sigma_{i,2}} = (1, 1, g^{\omega'_i}) \cdot \vec{C}_{\sigma_{i,2}}\ .$$

   Note that $\vec{\pi}_i = (\pi_{i,1}, \pi_{i,2}, \pi_{i,3})$ still satisfies the equation

$$E\big(g, \vec{C}'_{\sigma_{i,1}}\big) = E\big(H_{\mathbb{G}}(m_i), \vec{C}'_{\sigma_{i,2}}\big) \cdot E\big(\pi_{i,1}, \vec{f}_1\big) \cdot E\big(\pi_{i,2}, \vec{f}_2\big) \cdot E\big(\pi_{i,3}, \vec{f}_3\big)\ .$$

4. Set $\sigma_{n+1,1} = H_{\mathbb{G}}(m')^{\omega'_{n+1}}$ and $\sigma_{n+1,2} = g^{\omega'_{n+1}}$. Then, pick random $r_{n+1,1}, s_{n+1,1}, t_{n+1,1} \xleftarrow{R} \mathbb{Z}_p$, $r_{n+1,2}, s_{n+1,2}, t_{n+1,2} \xleftarrow{R} \mathbb{Z}_p$ and compute

$$\begin{aligned} \vec{C}'_{\sigma_{n+1,1}} &= (1, 1, \sigma_{n+1,1}) \cdot \vec{f}_1^{\,r_{n+1,1}} \cdot \vec{f}_2^{\,s_{n+1,1}} \cdot \vec{f}_3^{\,t_{n+1,1}} \\ \vec{C}'_{\sigma_{n+1,2}} &= (1, 1, \sigma_{n+1,2}) \cdot \vec{f}_1^{\,r_{n+1,2}} \cdot \vec{f}_2^{\,s_{n+1,2}} \cdot \vec{f}_3^{\,t_{n+1,2}} \end{aligned}$$

   as well as a NIWI argument

$$\vec{\pi}_{n+1} = \big(g^{r_{n+1,1}} \cdot H_{\mathbb{G}}(m')^{-r_{n+1,2}},\ g^{s_{n+1,1}} \cdot H_{\mathbb{G}}(m')^{-s_{n+1,2}}, g^{t_{n+1,1}} \cdot H_{\mathbb{G}}(m')^{-t_{n+1,2}}\big)$$

   for the equality $e(\sigma_{n+1,1}, g) = e(H_{\mathbb{G}}(m'), \sigma_{n+1,2})$.
5. Update $\vec{\pi}'_{sum} = (\pi'_{s,1}, \pi'_{s,2}, \pi'_{s,3})$ by computing

$$\vec{\pi}''_{sum} = (\pi''_{s,1}, \pi''_{s,2}, \pi''_{s,3}) = \big(\pi'_{s,1} \cdot g^{-r_{n+1,2}},\ \pi'_{s,2} \cdot g^{-s_{n+1,2}},\ \pi'_{s,3} \cdot g^{-t_{n+1,2}}\big)\ .$$

6. Finally, re-randomize the commitments $\{\vec{C}'_{i,1}, \vec{C}'_{i,2}\}_{i=1}^{n+1}$ and the proofs $\{\vec{\pi}_i\}_{i=1}^{n+1}$, $\vec{\pi}''_{sum}$ so as to obtain $\{\vec{C}''_{i,1}, \vec{C}''_{i,2}\}_{i=1}^{n+1}$ and $\{\vec{\pi}''_i\}_{i=1}^{n+1}$, $\vec{\pi}'''_{sum}$.

Return the derived signature

$$\sigma' = \Big(\vec{C}''_X, \{\vec{C}''_{\theta_j}\}_{j\in\{1,2,5\}}, \{\theta_j\}_{j\in\{3,4,6,7\}}, \vec{\pi}''_{\mathsf{aho},1}, \vec{\pi}''_{\mathsf{aho},2}, \{(m_i, \vec{C}''_{\sigma_{i,1}}, \vec{C}''_{\sigma_{i,2}}, \vec{\pi}''_i)\}_{i=1}^{n+1},\ \vec{\pi}'''_{sum}\Big) \quad (13)$$

after having re-organized the indexation of $\{(m_i, \vec{C}''_{\sigma_{i,1}}, \vec{C}''_{\sigma_{i,2}}, \vec{\pi}''_i)\}_{i=1}^{n+1}$ according to the pre-determined lexicographical order for $\{m_i\}_{i=1}^{n+1}$.

[9] This amounts to multiplying these commitments by commitments to the group element $1_{\mathbb{G}}$.

**Verify(pk, Msg, $\sigma$):** given pk, and a message $\mathsf{Msg} = \{m_i\}_{i=1}^n$, where $m_i \in \Sigma$ for each $i$, parse the signature $\sigma$ as in (12). Then, do the following.

1. Return 0 if $\vec{\pi}_{\mathsf{aho},1} = (\pi_1, \pi_2, \pi_3)$ and $\vec{\pi}_{\mathsf{aho},2} = (\pi_4, \pi_5, \pi_6)$ do not satisfy

$$(1_{\mathbb{G}_T}, 1_{\mathbb{G}_T}, A) \cdot E\big(\theta_3, (1,1,\theta_4)\big)^{-1} = E(G_z, \vec{C}_{\theta_1}) \cdot E(G_r, \vec{C}_{\theta_2}) \cdot E(G_1, \vec{C}_X) \cdot \prod_{j=1}^{3} E(\pi_j, \vec{f}_j)$$

$$(1_{\mathbb{G}_T}, 1_{\mathbb{G}_T}, B) \cdot E\big(\theta_6, (1,1,\theta_7)\big)^{-1} = E(H_z, \vec{C}_{\theta_1}) \cdot E(H_r, \vec{C}_{\theta_5}) \cdot E(H_1, \vec{C}_X) \cdot \prod_{j=1}^{3} E(\pi_{j+3}, \vec{f}_j) \ .$$

2. Return 0 if not all arguments $\{\vec{\pi}_i = (\pi_{i,1}, \pi_{i,2}, \pi_{i,3})\}_{i=1}^n$ satisfy (10).
3. Return 0 if $\vec{\pi}_{sum} = (\pi_{s,1}, \pi_{s,2}, \pi_{s,3})$ does not satisfy (11).

If the above checks were all successful, return 1.

In the above construction, we can sign sets $\mathsf{Msg} = \{m_i\}_{i=1}^n$ of cardinality $n$ using $9n + 25$ group elements. The security analysis is basically the same as in our first scheme. The sole difference is that, while the latter can rely on a RMA-secure SPS system [4] that does not require any other assumption than DLIN, the $q$-SFP assumption is needed here.

## E Completely Context-Hiding AOS Schemes

An append-only signature (AOS) [37] is a tuple of algorithms $(\mathsf{Setup}, \mathsf{Sign}, \mathsf{Append}, \mathsf{Verify})$, where Setup, Sign and Verify work in the usual way (except that messages consist of ordered tuples here). As for Append, it takes as input the public key pk, an ordered tuple $M = (m_1, \ldots, m_n)$, a message $m$ and a signature $\sigma$ such that $\mathsf{Verify}(\mathsf{pk}, M, \sigma) = 1$. It outputs a signature $\sigma'$ such that $\mathsf{Verify}(\mathsf{pk}, (m_1, \ldots, m_n, m), \sigma') = 1$. Note that algorithm Sign could be omitted from the above syntax since messages can be signed by running Append on an empty tuple $M = \emptyset$ and viewing the private key sk as a signature on the empty set (as done in [5]). We chose to retain Sign in the syntax for convenience.

The security definition of [37] basically demands that it be infeasible to come up with a signature on a tuple $M^\star = (m_1^\star, \ldots, m_{n^\star}^\star)$ without having obtained a signature on a prefix of $M^\star$.

Let $\Pi = (\mathsf{Keygen}, \mathsf{Sign}, \mathsf{SignDerive}, \mathsf{Verify})$ be a completely context-hiding and unforgeable HH-AOS scheme. Using $\Pi$, we obtain a completely context-hiding ordinary AOS scheme using the following simple construction.

**Setup($\lambda$):** run $(\mathsf{sk}, \mathsf{pk}) \leftarrow \Pi.\mathsf{Keygen}(\lambda)$ and output $(\mathsf{sk}, \mathsf{pk})$.

**Sign(sk, $M$):** In order to sign an ordered tuple $M = (m_1, \ldots, m_n)$, encode $M$ as a set $L = \{(1, m_1), \ldots, (n, m_n)\}$ of cardinality $n$. Then, use sk to compute and output $\sigma \leftarrow \Pi.\mathsf{Sign}(\mathsf{sk}, L)$.

**Append(pk, $\sigma$, $M$, $m$):** parse $M$ as $(m_1, \ldots, m_n)$ and encode it as a $n$-set $L = \{(1, m_1), \ldots, (n, m_n)\}$ of cardinality $n$. Then, compute and output $\sigma \leftarrow \Pi.\mathsf{SignDerive}(\mathsf{pk}, \{(\sigma, L)\}, (n+1, m))$.

**Verify(pk, $M$, $\sigma$):** given pk, parse $M$ as $(m_1, \ldots, m_n)$, define the set $L = \{(1, m_1), \ldots, (n, m_n)\}$ of cardinality $n$. Return 1 if $\Pi.\mathsf{Verify}(\mathsf{pk}, L, \sigma) = 1$ and 0 otherwise.

Note that, since we append to each individual message $m$ its position in the ordered tuple $M$ when encoding it as a set $L$, the underlying HH-AOS scheme $\Pi$ does not need to support multi-sets.

**Theorem 3.** *The above scheme is a completely context-hiding and unforgeable AOS system provided $\Pi$ is as completely context-hiding and unforgeable HH-AOS scheme.*

The proof of the above theorem is straightforward and omitted.