Multi-key Linearly Homomorphic Signature Scheme
Homomorphic Signatures: A Use Case

Elena’s average happiness from January to May?
The Simplest MKLHS scheme

History of Homomorphic Signatures

[Desmedt93] [JohnsonMSW02] [BonehF09] [BonehF11] [GorbunovW15] [FioreNMP16] [SchabhüserBB19]

[Desmedt93] [BonehF09] [BonehF11] [GorbunovW15] [FioreNMP16] [SchabhüserBB19]

Homomorphic Signatures

MKHS

MK version

waaaayy

Simpler

[AranhaP19]

The Simplest MKLHS scheme
The Simplest MKLHS scheme

Bilinear groups (type-2 pairings)
co-DDH in the random oracle model

\[ \sigma = (H(l) \cdot g^m)^{sk} \]

Achieves context-hiding in the multi-key settings

Compiler: any signature \( \rightarrow \) MKLHS

Bilinear groups. DDH + Flexible DH inversion hardness*

Schabhüser, Butin, Buchmann (CT-RSA2019) Context Hiding Multi-Key Linearly Homomorphic Authenticators
Quick Numeric Comparison with [SchabhüserBB19]

Signatures sizes assuming point compression

<table>
<thead>
<tr>
<th>Related work (BN-382)</th>
<th>Related work (BLS12-381)</th>
<th>This work (BN-382)</th>
<th>This work (BLS12-381)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secret Key (sk)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Public Key (pk)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fresh Signatures</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evaluated Signatures</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Protocol

<table>
<thead>
<tr>
<th>Size of</th>
<th>Schabhüser et al.</th>
<th>mklhs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secret Key (sk)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Public Key (pk)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fresh Signatures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evaluated Signatures</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Quick Numeric Comparison with [SchabhüserBB19]

### Implementation of MKHS (BLS12-381)

<table>
<thead>
<tr>
<th></th>
<th>Related Work</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key generation</td>
<td>30000000</td>
<td>0</td>
</tr>
<tr>
<td>Signature computation</td>
<td>10000000</td>
<td>0</td>
</tr>
<tr>
<td>Homomorphic evaluation</td>
<td>10000000</td>
<td>0</td>
</tr>
<tr>
<td>Verification</td>
<td>10000000</td>
<td>10000000</td>
</tr>
</tbody>
</table>

### Implementation of MKHS (BN-382)

<table>
<thead>
<tr>
<th></th>
<th>Related Work</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key generation</td>
<td>40000000</td>
<td>0</td>
</tr>
<tr>
<td>Signature computation</td>
<td>20000000</td>
<td>0</td>
</tr>
<tr>
<td>Homomorphic evaluation</td>
<td>20000000</td>
<td>0</td>
</tr>
<tr>
<td>Verification</td>
<td>20000000</td>
<td>20000000</td>
</tr>
</tbody>
</table>

Related Work.
The notion of homomorphic signature schemes (HS) was introduced by Johnson et al. in [20], together with a security model for HS and a concrete construction for redactable signatures. Intuitively, this first example of HS allows to erase part of the message as an homomorphic operation: given a message $m$ and a signature of $m$, anyone can derive from a new signature $\sigma_0$ for any message $m_0$ obtained after redacting $m$. Subsequent proposals extended the kind of operations supported by HS. The main bulk of work comes with constructions of linearly homomorphic signatures for linear network coding [3,7,8,10]. More recently, Catalano et al. [11] address the question of HS for higher degree functions and show applications to efficient verifiable computation of polynomial functions. The first construction of (leveled) fully homomorphic signatures (FHS) is due to Gorbunov et al. [18]. The scheme presented in [18] is a lattice-based HS capable of evaluating arbitrary boolean circuits of bounded polynomial depth over signed data. However, none of the aforementioned schemes support computations on data signed by multiple clients.

Agrawal et al. [1] expand the horizon of applications by considering multi-source signatures in the context of network coding. A few years later, Fiore et al. [13] formalize the concept of multi-key homomorphic authenticators and provided the first constructions of a multi-key homomorphic signature scheme (MKHS) and of a multi-key homomorphic MAC. The MKHS in [13] is designed...
**MKHS: Our Construction**

- **Setup** ($1^\lambda$) $\rightarrow$ pp $\rightarrow$ BilinGroup
  
  - KeyGen(pp) $\rightarrow$ (sk, pk, id)
    
    - $\text{sk} \leftarrow \mathbb{Z}_q^*$
    - $\text{pk} \leftarrow g_2^{\text{sk}}$

  - Sign(sk, $\ell$, $m$) $\rightarrow$ $\sigma$
    
    - $\gamma \leftarrow (H(\ell) \cdot g_1^m)^{sk}$
    - $\mu \leftarrow m$

  - Verify($\mathcal{P}$, $\{pk_{id}\}$, $m$, $\sigma$) $\rightarrow$ 😊
    
    - $m = \sum_{j=1}^{t} \mu_j$
    - $e(\gamma, g_2) = \prod_{j=1}^{t} e(g_1^{\mu_j} \cdot \prod_{i \in I_j} H_i^{f_i}, \text{pk}_j)$

**TYPE-2 PAIRINGS**

- + hash into $G_a$
- + homomorphism $G_2 \rightarrow G_a$

**Eval**($f$, $\sigma_1$,...,$\sigma_n$) $\rightarrow$ $\sigma = (\gamma, \mu_1, \ldots, \mu_t)$

- $\gamma \leftarrow \prod_{i=1}^{n} \gamma_i^{f_i}$
- $\mu_j \leftarrow \sum_{i \in I_j} f_i \cdot \mu_i$

- $H_i^{f_i} \cdot g_1^{\sum_{i \in I_1} f_i \cdot m_i} \cdot H_i^{f_i} \cdot g_1^{\sum_{i \in I_2} f_i \cdot m_i}$

The Simplest MKLHS scheme
Theorem 1. The mklhs scheme is secure in the random oracle model assuming that the co-CDH problem is computationally infeasible. In detail, let $\mathcal{A}$ be a probabilistic polynomial-time adversary in the security experiment (HUFCMA) described in Section 2.2, then its advantage is bounded by

$$\text{Adv}_{\text{mklhs}, \mathcal{A}}[\lambda] \leq \frac{1}{2} \cdot \left[ R_H + Q_{id} \cdot \text{Adv}_{\mathcal{B}}^{\text{coCDH}}[\lambda] \right].$$

where $\mathcal{B}$ is a polynomial-time algorithm that solves a co-CDH instance with probability $\text{Adv}_{\mathcal{B}}^{\text{coCDH}}[\lambda]$, $R_H = \frac{1}{\text{poly}(\lambda)}$ is determined by the prime number $q$ corresponding to the order of the group $G_1$ (the range of the hash function) and $Q_{id} = \text{poly}(\lambda)$ is the total number of identities generated during the game.
Security Model: Malicious Evaluatos

The Simplest MKLHS scheme
Security Model: Forgeries

 Forgery: \( \text{Ver}(\mathcal{P}^*, \{\text{pk}_{\text{id}}\}, m^*, \sigma^*) = 1 \)

1) the adversary successfully tampers with the output of the function

\[ m^* \neq f^*(m_1, \ldots, m_t) \]

2) the adversary never did an authentication query for a label in \( \mathcal{P}^* \)

\[ \exists \ k \in [t] \ : \ \ell_k^* \notin Q_{\text{labels}} \]
Why not ... security against corrupted users?

some sort of input secrecy?

The Simplest MKLHS scheme

[FioreNMP16] + first construction

MKHS

[FioreP18] + HS → MKHS

[FioreP18] + direct construction

Simplest

[AranhaP19]

+ context-hiding

DS → MKLHS

[SchabhüserBB19]

+ security under insider corruption

+ full signature succinctness

DS + SNARK → MKHS

[LaiTWC18]

This presentation may contain copyrighted images the use of which has not always been specifically authorized by the copyright owner. I am making the material available for educational purposes only and I believe this constitutes a 'fair use'.