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

Benoît Libert
ENS de Lyon

Marc Joye
Palo Alto, USA

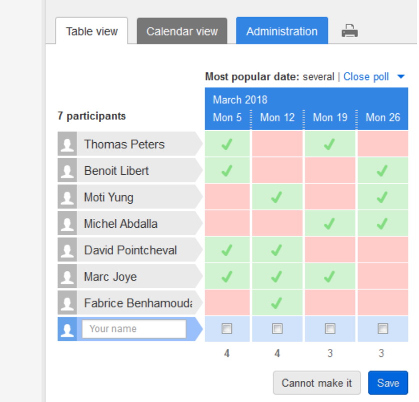
Moti Yung
New York, USA

Thomas Peters
ENS, Paris

Maryland - the 31st of March

Append-Only Signature

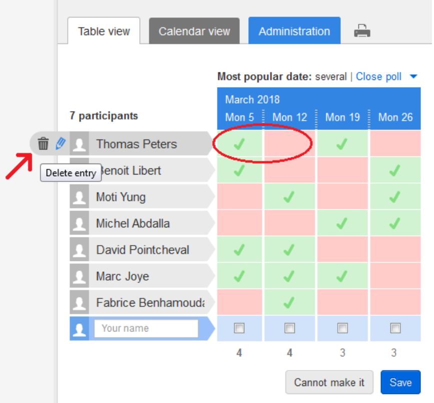
Allowing adding lines/opinions in a poll



Avoiding the above misbehavior $\Longrightarrow$ Ensuring non "redactness"

Append-Only Signature

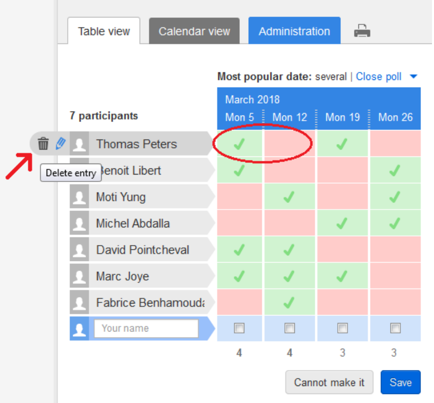
Allowing adding lines/opinions in a poll

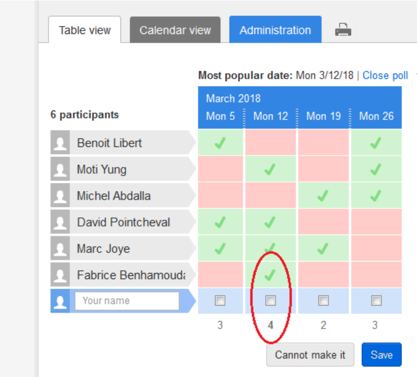


Avoiding the above misbehavior $\Longrightarrow$ Ensuring non "redactness"

Append-Only Signature

Allowing adding lines/opinions in a poll

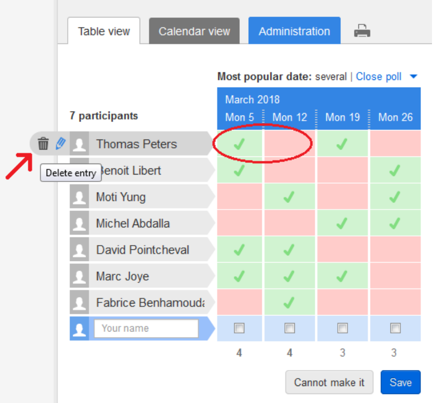


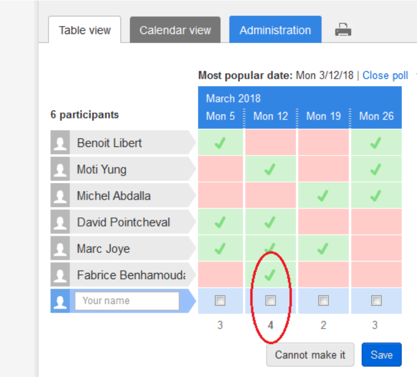


Avoiding the above misbehavior $\Longrightarrow$ Ensuring non "redactness"

Append-Only Signature

Allowing adding lines/opinions in a poll





Avoiding the above misbehavior $\Longrightarrow$ Ensuring non "redactness"

Append-Only Signature: Unforgeability

No PPT adversary can forge a signature with noticeable advantage in

PK $\longrightarrow$

$\vdots$

$M_i = \{m_{i,1}, \ldots, m_{i,n_i}\}$ $\longleftarrow$

Σ_i $\longrightarrow$

$\vdots$

M^*, Σ^* $\longleftarrow$

$\textsc{Sign}(\textsc{sk}, \cdot)$

$\textsc{SignDerive}(\textsc{pk}, \cdot, \cdot)$

If $\textsc{Verify}(M^*, \Sigma^*) \neq 1$ or $\boxed{M_i \subset M^*}$ for some $i = 1, \ldots, q$

Then $(M^*, \Sigma^*) \neq$ *forgery*

Append-Only Signature: Unforgeability

No PPT adversary can forge a signature with noticeable advantage in

$\xrightarrow{\mathsf{PK}}$

$\vdots$

$\xleftarrow{M_i = \{\mathsf{m}_{i,1}, ..., m_{i,n_i}\}}$

$\xrightarrow{\Sigma_i}$

$\vdots$

$\xleftarrow{M^*, \Sigma^*}$

$\textsc{Sign}(\textsc{sk}, \cdot)$ $\textsc{SignDerive}(\textsc{pk}, \cdot, \cdot)$

If $\textsc{Verify}(M^*, \Sigma^*) \neq 1$ or $\boxed{M_i \subset M^*}$ for some $i = 1, \ldots, q$

Then $(M^*, \Sigma^*) \neq$ *forgery*

Append-Only Signature: Unforgeability

No PPT adversary can forge a signature with noticeable advantage in

$\xrightarrow{\quad \mathsf{PK} \quad}$

$\vdots$

$\xleftarrow{\quad M_i = \{\mathsf{m}_{i,1}, \ldots, m_{i,n_i}\} \quad}$

$\xrightarrow{\quad \Sigma_i \quad}$

$\vdots$

$\xleftarrow{\quad M^*, \Sigma^* \quad}$

$\textsc{Sign}(\textsc{sk}, \cdot)$ $\qquad$ $\textsc{SignDerive}(\textsc{pk}, \cdot, \cdot)$

If $\textsc{Verify}(M^*, \Sigma^*) \neq 1$ or $\boxed{M_i \subset M^*}$ for some $i = 1, \ldots, q$

Then $(M^*, \Sigma^*) \neq$ *forgery*

Privacy of AOS: History-Hiding

Hiding the history of appended messages, which is implied by...

Context-Hiding [Ahn *et al.* (TCC'12)]

Derived signatures should "look" like fresh signatures, *even* if original (honestly generated) signatures are given

$\Rightarrow$ Guarantees *unlinkability* between derivatives of a signature

Complete Context-Hiding [Attrapadung-Libert-Peters (Asiacrypt'12)]

For all $M \subset \mathcal{M}$ along with a *possibly maliciously generated* valid signature Σ and for any M' such that $M \subset M'$:

$$\{\textsc{sk}, \Sigma, \textsc{Sign}(\textsc{sk}, M')\} \sim^S \{\textsc{sk}, \Sigma, \textsc{SignDerive}(\textsc{pk}, (\Sigma, M), M' \setminus M)\}$$

$\Rightarrow$ The definition takes into account *e.g.* randomizable Σ

Privacy of AOS: History-Hiding

Hiding the history of appended messages, which is implied by...

Context-Hiding [Ahn *et al.* (TCC'12)]

Derived signatures should "look" like fresh signatures, *even* if original (honestly generated) signatures are given

$\Rightarrow$ Guarantees *unlinkability* between derivatives of a signature

Complete Context-Hiding [Attrapadung-Libert-Peters (Asiacrypt'12)]

For all $M \subset \mathcal{M}$ along with a *possibly maliciously generated* valid signature Σ and for any M' such that $M \subset M'$:

$$\{\textsc{sk}, \Sigma, \textsc{Sign}(\textsc{sk}, M')\} \sim^{S} \{\textsc{sk}, \Sigma, \textsc{SignDerive}(\textsc{pk}, (\Sigma, M), M' \setminus M)\}$$

$\Rightarrow$ The definition takes into account *e.g.* randomizable Σ

Privacy of AOS: History-Hiding

Hiding the history of appended messages, which is implied by...

Context-Hiding [Ahn *et al.* (TCC'12)]

Derived signatures should "look" like fresh signatures, *even* if original (honestly generated) signatures are given

$\Rightarrow$ Guarantees *unlinkability* between derivatives of a signature

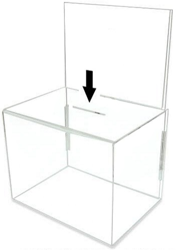
Complete Context-Hiding [Attrapadung-Libert-Peters (Asiacrypt'12)]

For all $M \subset \mathcal{M}$ along with a *possibly maliciously generated* valid signature Σ and for any M' such that $M \subset M'$:

$$\{\textsc{sk}, \Sigma, \textsc{Sign}(\textsc{sk}, M')\} \sim^{S} \{\textsc{sk}, \Sigma, \textsc{SignDerive}(\textsc{pk}, (\Sigma, M), M' \setminus M)\}$$

$\Rightarrow$ The definition takes into account *e.g.* randomizable Σ

History-hiding append-only signatures can be viewed as



Main functionality

- Allowing anyone to add messages or opinions (e.g. approval votes...)
- Security: preventing withdrawing other's inputs (secure archive)

History-Hiding property

- Not considering as strings but as sets
- Privacy: removing the order $\Longrightarrow$ Hides influences in successive appendings

Homomorphic Signatures: Related Work

- Desmedt (NSPW'93): Call for constructions
- Johnson-Molnar-Song-Wagner (CT-RSA'02):

  Formal definitions of homomorphic signatures
- Ahn-Boneh-Camenisch-Hohenberger-shelat-Waters (TCC'12):

  Generalized model, context-hiding privacy, constructions
- Attrapadung-Libert-Peters (Asiacrypt'12 & PKC'13):

  Stronger context-hiding privacy, separation results, improved constructions

Homomorphic Signatures: Related Work

- Desmedt (NSPW'93): Call for constructions
- Johnson-Molnar-Song-Wagner (CT-RSA'02):

  Formal definitions of homomorphic signatures
- Ahn-Boneh-Camenisch-Hohenberger-shelat-Waters (TCC'12):

  Generalized model, context-hiding privacy, constructions
- Attrapadung-Libert-Peters (Asiacrypt'12 & PKC'13):

  Stronger context-hiding privacy, separation results, improved constructions

Homomorphic Signatures: Related Work

- Desmedt (NSPW'93): Call for constructions
- Johnson-Molnar-Song-Wagner (CT-RSA'02):

  Formal definitions of homomorphic signatures
- Ahn-Boneh-Camenisch-Hohenberger-shelat-Waters (TCC'12):

  Generalized model, context-hiding privacy, constructions
- Attrapadung-Libert-Peters (Asiacrypt'12 & PKC'13):

  Stronger context-hiding privacy, separation results, improved constructions

Append-Only Signatures: Prior Works

- Kiltz-Mityagin-Panjwani-Raghavan (ICALP'05):

  Formal definitions, generic constructions, instantiations

- Bethencourt-Boneh-Waters (NDSS'07):

  History-Hiding, bounded set or random oracle

- Cui-Li-Yokoyama-Imai (ICC'09): Adaptive security

- Moran-Naor-Segev (ICALP'07): (write-once memories instead of signatures)

Append-Only Signatures: Prior Works

- Kiltz-Mityagin-Panjwani-Raghavan (ICALP'05):

  Formal definitions, generic constructions, instantiations

- Bethencourt-Boneh-Waters (NDSS'07):

  History-Hiding, bounded set or random oracle

- Cui-Li-Yokoyama-Imai (ICC'09): Adaptive security

- Moran-Naor-Segev (ICALP'07): (write-once memories instead of signatures)

Append-Only Signatures: Prior Works

- Kiltz-Mityagin-Panjwani-Raghavan (ICALP'05):

  Formal definitions, generic constructions, instantiations

- Bethencourt-Boneh-Waters (NDSS'07):

  History-Hiding, bounded set or random oracle

- Cui-Li-Yokoyama-Imai (ICC'09): Adaptive security

- Moran-Naor-Segev (ICALP'07): (write-once memories instead of signatures)

Append-Only Signatures: Prior Works

- Kiltz-Mityagin-Panjwani-Raghavan (ICALP'05):

  Formal definitions, generic constructions, instantiations

- Bethencourt-Boneh-Waters (NDSS'07):

  History-Hiding, bounded set or random oracle

- Cui-Li-Yokoyama-Imai (ICC'09): Adaptive security

- Moran-Naor-Segev (ICALP'07): (write-once memories instead of signatures)

Our Contributions

Recast History-Hiding AOS in homomorphic signature frameworks

Efficient History-Hiding AOS in prime-order bilinear groups

- Security **in the standard model under simple assumptions** (DLIN)
- **Constant-size public key** pk for sets of unbounded messages
- Signature of $\mathcal{O}(n)$-size for sets of messages $\{m_1, \ldots, m_n\} \in \mathbb{Z}_p^n$

New application: generic Identity-Based Ring Signatures

- Generic construction from HH-AOS for arbitrary-size rings
- Unforgeability against adaptively (as opposed to selectively) chosen rings
- Full Anonymity even for adversarially-chosen private keys of *ID*'s

Our Contributions

Recast History-Hiding AOS in homomorphic signature frameworks

Efficient History-Hiding AOS in prime-order bilinear groups

- Security **in the standard model under simple assumptions** (DLIN)
- **Constant-size public key** pk for sets of unbounded messages
- Signature of $\mathcal{O}(n)$-size for sets of messages $\{m_1, \ldots, m_n\} \in \mathbb{Z}_p^n$

New application: generic Identity-Based Ring Signatures

- Generic construction from HH-AOS for arbitrary-size rings
- Unforgeability against adaptively (as opposed to selectively) chosen rings
- Full Anonymity even for adversarially-chosen private keys of *ID*'s

HH-AOS in the Standard Model

Challenges:

- Bethencourt-Boneh-Waters (NDSS'07) rely on aggregate signatures

  Multi-linear maps and iO give standard-model adaptations ...

  ... but ruin the efficiency and require *ad hoc* assumptions

- Sequential aggregate signatures (e.g., based on Waters signatures [LOSSW06]) do not work here (see the full version of the paper)

Our solution: key ingredients

- Exploit the randomizability / malleability of Groth-Sahai proofs [GS08]
- Structure-preserving signatures based simple assumptions [ACD+12]
- Programmable hash functions [HK08] and a (one-time) standard-model instantiation of Boneh-Lynn-Schacham signatures [BLS01]

HH-AOS in the Standard Model

Challenges:

- Bethencourt-Boneh-Waters (NDSS'07) rely on aggregate signatures

  Multi-linear maps and iO give standard-model adaptations . . .

  . . . but ruin the efficiency and require *ad hoc* assumptions

- Sequential aggregate signatures (e.g., based on Waters signatures [LOSSW06]) do not work here (see the full version of the paper)

Our solution: key ingredients

- Exploit the randomizability / malleability of Groth-Sahai proofs [GS08]
- Structure-preserving signatures based simple assumptions [ACD+12]
- Programmable hash functions [HK08] and a (one-time) standard-model instantiation of Boneh-Lynn-Schacham signatures [BLS01]

Our Append-Only Signature: Outline

Uses a two-tier construction: To sign a set $\{m_1, \dots, m_n\}$

- Generate a fresh one-time key pair $(X = g^x, x) \in \mathbb{G} \times \mathbb{Z}_p$
- Certify the one-time public key $X = g^x$ using a long term secret key SK
- Use the one-time $x \in \mathbb{Z}_p$ to sign $\{m_1, \dots, m_n\}$ by splitting x into additive shares $x = \omega_1 + \cdots + \omega_n$:

  Compute $\sigma_i = H_{\mathbb{G}}(m_i)^{\omega_i}$ for each $i \in \{1, \dots, n\}$ (and $\mathsf{pk}_i = g^{\omega_i}$)

  Commit to each $\sigma_i = H_{\mathbb{G}}(m_i)^{\omega_i}$ and prove consistency with $X = g^{\sum_{i=1}^n \omega_i}$

Inserts m_{n+1} in a signed $\{m_1, \dots, m_n\}$ by turning a (n, n) additive sharing of $x = \sum_{i=1}^n \omega_i$ into a $(n+1, n+1)$ sharing $x = \sum_{i=1}^{n+1} \omega_i'$ "in the exponent"

Leverages the malleability of GS proofs to derive a proof that $X = g^{\sum_{i=1}^{n+1} \omega_i'}$

Our Append-Only Signature: Outline

Uses a two-tier construction: To sign a set $\{m_1, \ldots, m_n\}$

- Generate a fresh one-time key pair $(X = g^x, x) \in \mathbb{G} \times \mathbb{Z}_p$
- Certify the one-time public key $X = g^x$ using a long term secret key SK
- Use the one-time $x \in \mathbb{Z}_p$ to sign $\{m_1, \ldots, m_n\}$ by splitting x into additive shares $x = \omega_1 + \cdots + \omega_n$:

  Compute $\sigma_i = H_{\mathbb{G}}(m_i)^{\omega_i}$ for each $i \in \{1, \ldots, n\}$ (and $\mathsf{pk}_i = g^{\omega_i}$)

  Commit to each $\sigma_i = H_{\mathbb{G}}(m_i)^{\omega_i}$ and prove consistency with $X = g^{\sum_{i=1}^{n} \omega_i}$

Inserts m_{n+1} in a signed $\{m_1, \ldots, m_n\}$ by turning a (n, n) additive sharing of $x = \sum_{i=1}^{n} \omega_i$ into a $(n+1, n+1)$ sharing $x = \sum_{i=1}^{n+1} \omega_i'$ "in the exponent"

Leverages the malleability of GS proofs to derive a proof that $X = g^{\sum_{i=1}^{n+1} \omega_i'}$

Our Append-Only Signature: Outline

Uses a two-tier construction: To sign a set $\{m_1, \ldots, m_n\}$

- Generate a fresh one-time key pair $(X = g^x, x) \in \mathbb{G} \times \mathbb{Z}_p$
- Certify the one-time public key $X = g^x$ using a long term secret key SK
- Use the one-time $x \in \mathbb{Z}_p$ to sign $\{m_1, \ldots, m_n\}$ by splitting x into additive shares $x = \omega_1 + \cdots + \omega_n$:

  Compute $\sigma_i = H_{\mathbb{G}}(m_i)^{\omega_i}$ for each $i \in \{1, \ldots, n\}$ (and $\mathsf{pk}_i = g^{\omega_i}$)

  Commit to each $\sigma_i = H_{\mathbb{G}}(m_i)^{\omega_i}$ and prove consistency with $X = g^{\sum_{i=1}^{n} \omega_i}$

Inserts m_{n+1} in a signed $\{m_1, \ldots, m_n\}$ by turning a (n, n) additive sharing of $x = \sum_{i=1}^{n} \omega_i$ into a $(n+1, n+1)$ sharing $x = \sum_{i=1}^{n+1} \omega_i'$ "in the exponent"

Leverages the malleability of GS proofs to derive a proof that $X = g^{\sum_{i=1}^{n+1} \omega_i'}$

Our Append-Only Signature: first step (non-HH)

...only achieving unforgeability

$\mathsf{KeyGen}(\mathsf{pp})$ where $\mathsf{pp} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$

- Let $(\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ be a signature scheme with $\mathcal{M}_0 = \mathbb{G}$
- Let $H_{\mathbb{G}} : \{0,1\}^L \rightarrow \mathbb{G}$ such that $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{j=1}^{L} h_j^{m[j]} \in \mathbb{G}$
- Run $(pk, sk) \leftarrow \mathsf{KeyGen}_0(\mathsf{pp})$ and set $\mathsf{PK} = (H_{\mathbb{G}}, \mathsf{pk})$ and $\mathsf{SK} = \mathsf{sk}$

$\mathsf{Sign}(\mathsf{SK}, M = \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Generate a random $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$
- Authenticate X as $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Share x into n pieces: pick $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ s.t. $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each $m_i \in M$ using ω_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Output $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$ as the signature

Our Append-Only Signature: first step (non-HH)

...only achieving unforgeability

KeyGen(pp) where $\mathsf{pp} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$

- Let $(\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ be a signature scheme with $\mathcal{M}_0 = \mathbb{G}$
- Let $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ such that $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{j=1}^{L} h_j^{m[j]} \in \mathbb{G}$
- Run $(pk, sk) \leftarrow \mathsf{KeyGen}_0(\mathsf{pp})$ and set $\mathsf{PK} = (H_{\mathbb{G}}, \mathsf{pk})$ and $\mathsf{SK} = \mathsf{sk}$

Sign$(\mathsf{SK}, M = \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Generate a random $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$
- Authenticate X as $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Share x into n pieces: pick $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ s.t. $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each $m_i \in M$ using ω_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Output $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$ as the signature

Our Append-Only Signature: first step (non-HH)

...only achieving unforgeability

KeyGen(pp) where $\mathrm{pp} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$

- Let $(\mathrm{KeyGen}_0, \mathrm{Sign}_0, \mathrm{Verify}_0)$ be a signature scheme with $\mathcal{M}_0 = \mathbb{G}$
- Let $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ such that $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{j=1}^{L} h_j^{m[j]} \in \mathbb{G}$
- Run $(pk, sk) \leftarrow \mathrm{KeyGen}_0(\mathrm{pp})$ and set $\mathrm{PK} = (H_{\mathbb{G}}, \mathrm{pk})$ and $\mathrm{SK} = \mathrm{sk}$

Sign(SK, $M = \{m_1, \ldots, m_n\}$) with $m_1, \ldots, m_n \in \{0,1\}^L$

- Generate a random $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$
- Authenticate X as $\sigma_0 \leftarrow \mathrm{Sign}(\mathrm{sk}, X)$
- Share x into n pieces: pick $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ s.t. $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each $m_i \in M$ using ω_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Output $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$ as the signature

Our Append-Only Signature: first step (non-HH)

...only achieving unforgeability

KeyGen(pp) where $\mathsf{pp} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$

- Let $(\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ be a signature scheme with $\mathcal{M}_0 = \mathbb{G}$
- Let $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ such that $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{j=1}^{L} h_j^{m[j]} \in \mathbb{G}$
- Run $(pk, sk) \leftarrow \mathsf{KeyGen}_0(\mathsf{pp})$ and set $\mathsf{PK} = (H_{\mathbb{G}}, \mathsf{pk})$ and $\mathsf{SK} = \mathsf{sk}$

Sign$(\mathsf{SK}, M = \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Generate a random $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$
- Authenticate X as $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Share x into n pieces: pick $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ s.t. $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each $m_i \in M$ using ω_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Output $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$ as the signature

Our Append-Only Signature: first step (non-HH)

...only achieving unforgeability

$\mathsf{KeyGen}(\mathsf{pp})$ where $\mathsf{pp} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$

- Let $(\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ be a signature scheme with $\mathcal{M}_0 = \mathbb{G}$
- Let $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ such that $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{j=1}^{L} h_j^{m[j]} \in \mathbb{G}$
- Run $(pk, sk) \leftarrow \mathsf{KeyGen}_0(\mathsf{pp})$ and set $\mathsf{PK} = (H_{\mathbb{G}}, \mathsf{pk})$ and $\mathsf{SK} = \mathsf{sk}$

$\mathsf{Sign}(\mathsf{SK}, M = \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Generate a random $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$
- Authenticate X as $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Share x into n pieces: pick $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ s.t. $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each $m_i \in M$ using ω_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Output $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$ as the signature

Our Append-Only Signature: first step (non-HH)

...only achieving unforgeability

KeyGen(pp) where $\mathsf{pp} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$

- Let $(\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ be a signature scheme with $\mathcal{M}_0 = \mathbb{G}$
- Let $H_{\mathbb{G}} : \{0,1\}^L \to \mathbb{G}$ such that $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{j=1}^{L} h_j^{m[j]} \in \mathbb{G}$
- Run $(pk, sk) \leftarrow \mathsf{KeyGen}_0(\mathsf{pp})$ and set $\mathsf{PK} = (H_{\mathbb{G}}, \mathsf{pk})$ and $\mathsf{SK} = \mathsf{sk}$

Sign$(\mathsf{SK}, M = \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Generate a random $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$
- Authenticate X as $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Share x into n pieces: pick $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ s.t. $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each $m_i \in M$ using ω_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Output $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$ as the signature

Our Append-Only Signature: first step (non-HH)

...only achieving unforgeability

KeyGen(pp) where $\mathsf{pp} = (\mathbb{G}, \mathbb{G}_T, p, g, e)$

- Let $(\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ be a signature scheme with $\mathcal{M}_0 = \mathbb{G}$
- Let $H_{\mathbb{G}} : \{0,1\}^L \rightarrow \mathbb{G}$ such that $H_{\mathbb{G}}(m) = h_0 \cdot \prod_{j=1}^{L} h_j^{m[j]} \in \mathbb{G}$
- Run $(pk, sk) \leftarrow \mathsf{KeyGen}_0(\mathsf{pp})$ and set $\mathsf{PK} = (H_{\mathbb{G}}, \mathsf{pk})$ and $\mathsf{SK} = \mathsf{sk}$

Sign$(\mathsf{SK}, M = \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Generate a random $X = g^x$, with $x \xleftarrow{R} \mathbb{Z}_p$
- Authenticate X as $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Share x into n pieces: pick $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ s.t. $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each $m_i \in M$ using ω_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Output $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$ as the signature

...only achieving unforgeability (continuing with verification)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

$\mathsf{Verify}(\mathsf{PK}, \{m_1, \ldots, m_n\}, \Sigma)$ returns 1 only if

- If $\mathsf{Verify}(\mathsf{pk}, X, \sigma_0) = 1$
- If $X = \prod_{i=1}^{n} \sigma_{i,2}$ (i.e. $x = \omega_1 + \cdots + \omega_n$)
- If $e(\sigma_{i,1}, g) = e(H_{\mathbb{G}}(m_i), \sigma_{i,2})$ for all $i = 1, \ldots, n$

...only achieving unforgeability (continuing with verification)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^n \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

$\mathsf{Verify}(\mathsf{PK}, \{m_1, \ldots, m_n\}, \Sigma)$ returns 1 only if

- If $\mathsf{Verify}(\mathsf{pk}, X, \sigma_0) = 1$
- If $X = \prod_{i=1}^n \sigma_{i,2}$ (i.e. $x = \omega_1 + \cdots + \omega_n$)
- If $e(\sigma_{i,1}, g) = e(H_{\mathbb{G}}(m_i), \sigma_{i,2})$ for all $i = 1, \ldots, n$

...only achieving unforgeability (continuing with verification)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \stackrel{R}{\leftarrow} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \stackrel{R}{\leftarrow} \mathbb{Z}_p$ such that $x = \sum_{i=1}^n \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

$\mathsf{Verify}(\mathsf{PK}, \{m_1, \ldots, m_n\}, \Sigma)$ returns 1 only if

- If $\mathsf{Verify}(\mathsf{pk}, X, \sigma_0) = 1$
- If $X = \prod_{i=1}^n \sigma_{i,2}$ (i.e. $x = \omega_1 + \cdots + \omega_n$)
- If $e(\sigma_{i,1}, g) = e(H_{\mathbb{G}}(m_i), \sigma_{i,2})$ for all $i = 1, \ldots, n$

...only achieving unforgeability (appending messages)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

$\mathsf{SignDerive}(\mathsf{PK}, (\{m_1, \ldots, m_n\}, \Sigma), m_{n+1})$ appends $m_{n+1} \in \{0,1\}^L$

- Select $\omega'_1, \ldots, \omega'_n, \omega'_{n+1} \xleftarrow{R} \mathbb{Z}_p$ such that $0 = \sum_{i=1}^{n+1} \omega'_i$
- Randomize each pair $(\sigma_{i,1}, \sigma_{i,2})$ with ω'_i:

$$\sigma'_{i,1} = \sigma_{i,1} \cdot H_{\mathbb{G}}(m_i)^{\omega'_i} \qquad \sigma'_{i,2} = \sigma_{i,2} \cdot g^{\omega'_i}$$

- Authenticate m_{n+1} as $\sigma'_{n+1,1} = H_{\mathbb{G}}(m_{n+1})^{\omega'_{n+1}}$ and $\sigma'_{n+1,2} = g^{\omega'_{n+1}}$

Correctness is obvious ... but X is not randomizable ...

...only achieving unforgeability (appending messages)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \stackrel{R}{\leftarrow} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \stackrel{R}{\leftarrow} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

$\mathsf{SignDerive}(\mathsf{PK}, (\{m_1, \ldots, m_n\}, \Sigma), m_{n+1})$ appends $m_{n+1} \in \{0,1\}^L$

- Select $\omega'_1, \ldots, \omega'_n, \omega'_{n+1} \stackrel{R}{\leftarrow} \mathbb{Z}_p$ such that $0 = \sum_{i=1}^{n+1} \omega'_i$
- Randomize each pair $(\sigma_{i,1}, \sigma_{i,2})$ with ω'_i:

$$\sigma'_{i,1} = \sigma_{i,1} \cdot H_{\mathbb{G}}(m_i)^{\omega'_i} \qquad \sigma'_{i,2} = \sigma_{i,2} \cdot g^{\omega'_i}$$

- Authenticate m_{n+1} as $\sigma'_{n+1,1} = H_{\mathbb{G}}(m_{n+1})^{\omega'_{n+1}}$ and $\sigma'_{n+1,2} = g^{\omega'_{n+1}}$

Correctness is obvious ... but X is not randomizable ...

...only achieving unforgeability (appending messages)

Sign(SK, $\{m_1, \ldots, m_n\}$) with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathrm{Sign}(\mathrm{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

SignDerive(PK, ($\{m_1, \ldots, m_n\}, \Sigma$), m_{n+1}) appends $m_{n+1} \in \{0,1\}^L$

- Select $\omega'_1, \ldots, \omega'_n, \omega'_{n+1} \xleftarrow{R} \mathbb{Z}_p$ such that $0 = \sum_{i=1}^{n+1} \omega'_i$
- Randomize each pair $(\sigma_{i,1}, \sigma_{i,2})$ with ω'_i:

$$\sigma'_{i,1} = \sigma_{i,1} \cdot H_{\mathbb{G}}(m_i)^{\omega'_i} \qquad \sigma'_{i,2} = \sigma_{i,2} \cdot g^{\omega'_i}$$

- Authenticate m_{n+1} as $\sigma'_{n+1,1} = H_{\mathbb{G}}(m_{n+1})^{\omega'_{n+1}}$ and $\sigma'_{n+1,2} = g^{\omega'_{n+1}}$

Correctness is obvious ... but X is not randomizable ...

...only achieving unforgeability (appending messages)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

$\mathsf{SignDerive}(\mathsf{PK}, (\{m_1, \ldots, m_n\}, \Sigma), m_{n+1})$ appends $m_{n+1} \in \{0,1\}^L$

- Select $\omega'_1, \ldots, \omega'_n, \omega'_{n+1} \xleftarrow{R} \mathbb{Z}_p$ such that $0 = \sum_{i=1}^{n+1} \omega'_i$
- Randomize each pair $(\sigma_{i,1}, \sigma_{i,2})$ with ω'_i:

$$\sigma'_{i,1} = \sigma_{i,1} \cdot H_{\mathbb{G}}(m_i)^{\omega'_i} \qquad \sigma'_{i,2} = \sigma_{i,2} \cdot g^{\omega'_i}$$

- Authenticate m_{n+1} as $\sigma'_{n+1,1} = H_{\mathbb{G}}(m_{n+1})^{\omega'_{n+1}}$ and $\sigma'_{n+1,2} = g^{\omega'_{n+1}}$

Correctness is obvious ... but X is not randomizable ...

...only achieving unforgeability (appending messages)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

$\mathsf{SignDerive}(\mathsf{PK}, (\{m_1, \ldots, m_n\}, \Sigma), m_{n+1})$ appends $m_{n+1} \in \{0,1\}^L$

- Select $\omega'_1, \ldots, \omega'_n, \omega'_{n+1} \xleftarrow{R} \mathbb{Z}_p$ such that $0 = \sum_{i=1}^{n+1} \omega'_i$
- Randomize each pair $(\sigma_{i,1}, \sigma_{i,2})$ with ω'_i:

$$\sigma'_{i,1} = \sigma_{i,1} \cdot H_{\mathbb{G}}(m_i)^{\omega'_i} \qquad \sigma'_{i,2} = \sigma_{i,2} \cdot g^{\omega'_i}$$

- Authenticate m_{n+1} as $\sigma'_{n+1,1} = H_{\mathbb{G}}(m_{n+1})^{\omega'_{n+1}}$ and $\sigma'_{n+1,2} = g^{\omega'_{n+1}}$

Correctness is obvious ... but X is not randomizable ...

...only achieving unforgeability (security)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \stackrel{R}{\leftarrow} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \stackrel{R}{\leftarrow} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Append-Only Unforgeability

1. If $\Pi_0 = (\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ is secure against eXtended RMA
2. If $H_{\mathbb{G}}$ is an $(1,q)$-programmable hash function (CDH)

Then security follows...

Programmability [HK08]: The Waters hash $H_{\mathbb{G}}(m) = g^{J(m)} h^{K(m)}$

- Secretly computable $J(\cdot)$ and $K(\cdot)$ (in the reduction)
- For any $m_1, \ldots, m_{q+1}$ only 1 has $K(m_i) = 0$ (with good probability)

...only achieving unforgeability (security)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Append-Only Unforgeability

1. If $\Pi_0 = (\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ is secure against eXtended RMA
2. If $H_{\mathbb{G}}$ is an $(1, q)$-programmable hash function (CDH)

Then security follows...

Programmability [HK08]: The Waters hash $H_{\mathbb{G}}(m) = g^{J(m)} h^{K(m)}$

- Secretly computable $J(\cdot)$ and $K(\cdot)$ (in the reduction)
- For any $m_1, \ldots, m_{q+1}$ only 1 has $K(m_i) = 0$ (with good probability)

...only achieving unforgeability (security)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Append-Only Unforgeability

1. If $\Pi_0 = (\mathsf{KeyGen}_0, \mathsf{Sign}_0, \mathsf{Verify}_0)$ is secure against eXtended RMA
2. If $H_{\mathbb{G}}$ is an $(1, q)$-programmable hash function (CDH)

Then security follows...

Programmability [HK08]: The Waters hash $H_{\mathbb{G}}(m) = g^{J(m)} h^{K(m)}$

- Secretly computable $J(\cdot)$ and $K(\cdot)$ (in the reduction)
- For any $m_1, \ldots, m_{q+1}$ only 1 has $K(m_i) = 0$ (with good probability)

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Let $\Sigma^* = (X^*, \sigma_0^*, \{\sigma_{i,1}^*, \sigma_{i,2}^*\}_{i=1}^{n^*})$ be a forgery on $M^* = \{m_1^*, \ldots, m_{n^*}^*\}$

1. If X^* is fresh, σ_0^* is a forgery on the scheme Π_0
2. Otherwise $X^* = X_j$ for some j-th query on $M_j = \{m_1, \ldots, m_{n_j}\}$

Given a CDH instance (g, g^a, g^b), in the reduction $H_{\mathbb{G}}(m) = g^{J(m)}(g^b)^{K(m)}$

- Guess j in advance and set $X_j = g^a$ then sign it
- Guess a message index $i \in \{1, \ldots, n_j\}$ such that $m_i \notin M^*$
- Hope m_i is the only one in $M^* \cup M_j$ such that $K(m_i) = 0$

$\mathsf{Sign}(SK, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Let $\Sigma^* = (X^*, \sigma_0^*, \{\sigma_{i,1}^*, \sigma_{i,2}^*\}_{i=1}^{n^*})$ be a forgery on $M^* = \{m_1^*, \ldots, m_{n^*}^*\}$

1. If X^* is fresh, σ_0^* is a forgery on the scheme Π_0
2. Otherwise $X^* = X_j$ for some j-th query on $M_j = \{m_1, \ldots, m_{n_j}\}$

Given a CDH instance (g, g^a, g^b), in the reduction $H_{\mathbb{G}}(m) = g^{J(m)}(g^b)^{K(m)}$

- Guess j in advance and set $X_j = g^a$ then sign it
- Guess a message index $i \in \{1, \ldots, n_j\}$ such that $m_i \notin M^*$
- Hope m_i is the only one in $M^* \cup M_j$ such that $K(m_i) = 0$

$\mathsf{Sign}(\mathsf{SK}, \{m_1, \ldots, m_n\})$ with $m_1, \ldots, m_n \in \{0,1\}^L$

- Compute $X = g^x$, for $x \xleftarrow{R} \mathbb{Z}_p$, and $\sigma_0 \leftarrow \mathsf{Sign}(\mathsf{sk}, X)$
- Select $\omega_1, \ldots, \omega_n \xleftarrow{R} \mathbb{Z}_p$ such that $x = \sum_{i=1}^{n} \omega_i$
- Authenticate each m_i as $\sigma_{i,1} = H_{\mathbb{G}}(m_i)^{\omega_i}$ and $\sigma_{i,2} = g^{\omega_i}$

Let $\Sigma^* = (X^*, \sigma_0^*, \{\sigma_{i,1}^*, \sigma_{i,2}^*\}_{i=1}^{n^*})$ be a forgery on $M^* = \{m_1^*, \ldots, m_{n^*}^*\}$

1. If X^* is fresh, σ_0^* is a forgery on the scheme Π_0
2. Otherwise $X^* = X_j$ for some j-th query on $M_j = \{m_1, \ldots, m_{n_j}\}$

Given a CDH instance (g, g^a, g^b), in the reduction $H_{\mathbb{G}}(m) = g^{J(m)}(g^b)^{K(m)}$

- Guess j in advance and set $X_j = g^a$ then sign it
- Guess a message index $i \in \{1, \ldots, n_j\}$ such that $m_i \notin M^*$
- Hope m_i is the only one in $M^* \cup M_j$ such that $K(m_i) = 0$

Our History-Hiding Append-Only Signature

Add a layer of Groth-Sahai proofs above $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$

- Mallebility of GS proofs allows keeping the derivability
- Perfectly hiding CRS provides NIWI proofs: none info. on X
- Perfect randomizability completely redistributes the proof of X

...compeletly context-hiding follows (and then history-hiding)

Hardness Assumptions

1. **Decision Linear Problem (DLIN)**: given $(g, g^a, g^b, g^{ac}, g^{bd}, g^{\eta}) \in \mathbb{G}^6$, decide whether $\eta = c+d$ or $\eta \in_R \mathbb{Z}_p$

First switch the CRS into a perfectly sound CRS (extractable proof)...

[ACD+12]: DLIN-based instantiation of XRMA secure signature

Our History-Hiding Append-Only Signature

Add a layer of Groth-Sahai proofs above $\Sigma = (X, \sigma_0, \{(\sigma_{i,1}, \sigma_{i,2})\}_{i=1}^{n})$

- Mallebility of GS proofs allows keeping the derivability
- Perfectly hiding CRS provides NIWI proofs: none info. on X
- Perfect randomizability completely redistributes the proof of X

...compeletly context-hiding follows (and then history-hiding)

Hardness Assumptions

1. **Decision Linear Problem (DLIN)**: given $(g, g^a, g^b, g^{ac}, g^{bd}, g^{\eta}) \in \mathbb{G}^6$, decide whether $\eta = c+d$ or $\eta \in_R \mathbb{Z}_p$

First switch the CRS into a perfectly sound CRS (extractable proof)...

[ACD+12]: DLIN-based instantiation of XRMA secure signature

Identity-Based Ring Signatures from HH-AOS

Let $(\mathsf{AO.Keygen}, \mathsf{AO.Sign}, \mathsf{AO.SignDerive}, \mathsf{AO.Verify})$ be an AO Signature

$\mathsf{Setup}(\lambda)$: Output $(\mathsf{msk}, \mathsf{mpk}) := (\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{AO.Keygen}(\lambda)$

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

$\mathsf{Sign}(\mathsf{mpk}, d_{id}, M, \mathcal{R})$: given $id \in \mathcal{R} = \{id_1, \ldots, id_r\}$

- Encode M et $\mathcal{R}$ as $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$
- Compute $\sigma \leftarrow \mathsf{AO.SignDerive}(\mathsf{PK}, \{(d_{id}, \{0\|id\})\}, L)$

$\mathsf{Verify}(\mathsf{mpk}, M, \mathcal{R}, \sigma)$:

- Encode M et $\mathcal{R}$ as $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$
- Output $\mathsf{AO.Verify}(\mathsf{pk}, L, \sigma)$

Identity-Based Ring Signatures from HH-AOS

Let (AO.Keygen, AO.Sign, AO.SignDerive, AO.Verify) be an AO Signature

Setup(λ): Output $(\mathsf{msk}, \mathsf{mpk}) := (\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{AO.Keygen}(\lambda)$

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

Sign(mpk, d_{id}, M, $\mathcal{R}$): given $id \in \mathcal{R} = \{id_1, \ldots, id_r\}$

- Encode M et $\mathcal{R}$ as $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$
- Compute $\sigma \leftarrow \mathsf{AO.SignDerive}(\mathsf{PK}, \{(d_{id}, \{0\|id\})\}, L)$

Verify(mpk, M, $\mathcal{R}$, σ):

- Encode M et $\mathcal{R}$ as $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$
- Output $\mathsf{AO.Verify}(\mathsf{pk}, L, \sigma)$

Identity-Based Ring Signatures from HH-AOS

Let $(\mathsf{AO.Keygen}, \mathsf{AO.Sign}, \mathsf{AO.SignDerive}, \mathsf{AO.Verify})$ be an AO Signature

$\mathsf{Setup}(\lambda)$: Output $(\mathsf{msk}, \mathsf{mpk}) := (\mathsf{SK}, \mathsf{PK}) \leftarrow \mathsf{AO.Keygen}(\lambda)$

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

$\mathsf{Sign}(\mathsf{mpk}, d_{id}, M, \mathcal{R})$: given $id \in \mathcal{R} = \{id_1, \ldots, id_r\}$

- Encode M et $\mathcal{R}$ as $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$
- Compute $\sigma \leftarrow \mathsf{AO.SignDerive}(\mathsf{PK}, \{(d_{id}, \{0\|id\})\}, L)$

$\mathsf{Verify}(\mathsf{mpk}, M, \mathcal{R}, \sigma)$:

- Encode M et $\mathcal{R}$ as $L = \{0\|id_1, \ldots, 0\|id_r, 1\|M\|\mathcal{R}\}$
- Output $\mathsf{AO.Verify}(\mathsf{pk}, L, \sigma)$

Conclusion

We gave:

- The first HH-AOS for abribtarily large sets with fixed-size keys in the standard model
  - Based on simple assumptions
  - Based on a new design principle (different from [BBW07])
- New application to generic identity-based ring signatures
- A new view of AOS schemes in homomorphic signatures frameworks

  Also gives AOS satisfying stronger privacy definitions

Open problem:

Extension supporting other set homomorphic operations (e.g., set union)

Thank you!

Questions?