Politecnico di Torino
1859
Dipartimento di Scienze Matematiche "G. L. Lagrange"

DISMA
ECCELLENZA 2018·2022

Telsy

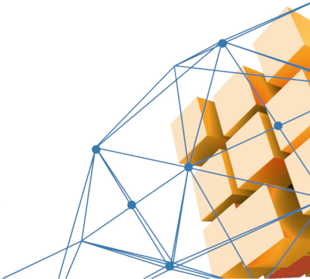
Sequential Aggregation of Multivariate Trapdoor Signatures

Edoardo Signorini
`edoardo.signorini@telsy.it`

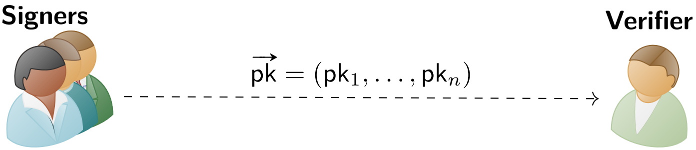
Telsy S.p.A.

Research at CryptoLabTN: Post-Quantum Cryptography

31/05/2023



Aggregate Signatures



$$(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$$

$$\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m_i)$$

Aggregate Signatures

Signers

Verifier

$\overrightarrow{\mathsf{pk}} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$ - - - - - - - - - - →

$$(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$$
$$\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m_i)$$
$$\Sigma \leftarrow \mathsf{AggSign}(\vec{m}, \vec{\sigma})$$

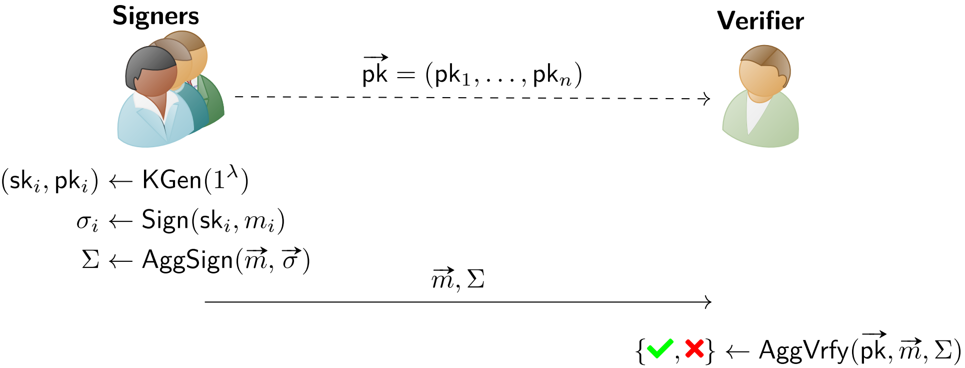
$$\xrightarrow{\vec{m}, \Sigma}$$

$$\{✓, ✗\} \leftarrow \mathsf{AggVrfy}(\overrightarrow{\mathsf{pk}}, \vec{m}, \Sigma)$$

Goal

Combine multiple σ_i in a single Σ such that $|\Sigma| \ll \sum_i |\sigma_i|$

Aggregate Signatures



Signers

Verifier

$\overrightarrow{\mathsf{pk}} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$

$(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$

$\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, m_i)$

$\Sigma \leftarrow \mathsf{AggSign}(\vec{m}, \vec{\sigma})$

$\vec{m}, \Sigma$

$\{✓, ✗\} \leftarrow \mathsf{AggVrfy}(\overrightarrow{\mathsf{pk}}, \vec{m}, \Sigma)$

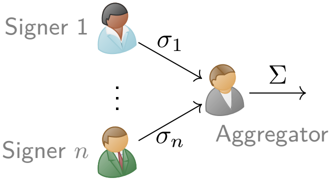
Goal

Combine multiple σ_i in a single Σ such that $|\Sigma| \ll \sum_i |\sigma_i|$

- Reduce bandwidth consumption
- Generalize multisignatures
- Certificate chains
- Blockchains

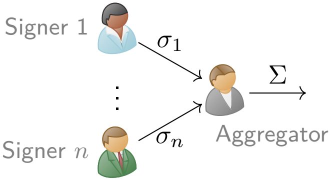
Types of Aggregate Signature

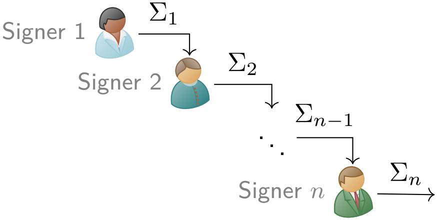
- General Aggregate Signature
  - Public aggregation by third party
  - No interaction required by signers
  - Only known construction are based on bilinear pairings [BGLS03]



Types of Aggregate Signature

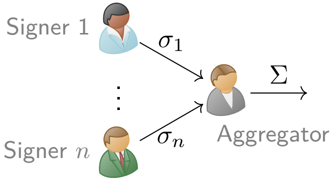
- General Aggregate Signature
  - Public aggregation by third party
  - No interaction required by signers
  - Only known construction are based on bilinear pairings [BGLS03]
- Sequential Aggregate Signature (SAS)
  - Signatures are iteratively aggregated
  - Aggregation by signers only
  - Can be built from trapdoor permutation [LMRS04; Nev08; BGR12; GOR18]

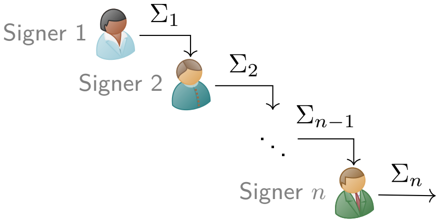




Types of Aggregate Signature

- General Aggregate Signature
  - Public aggregation by third party
  - No interaction required by signers
  - Only known construction are based on bilinear pairings [BGLS03]
- Sequential Aggregate Signature (SAS)
  - Signatures are iteratively aggregated
  - Aggregation by signers only
  - Can be built from trapdoor permutation [LMRS04; Nev08; BGR12; GOR18]

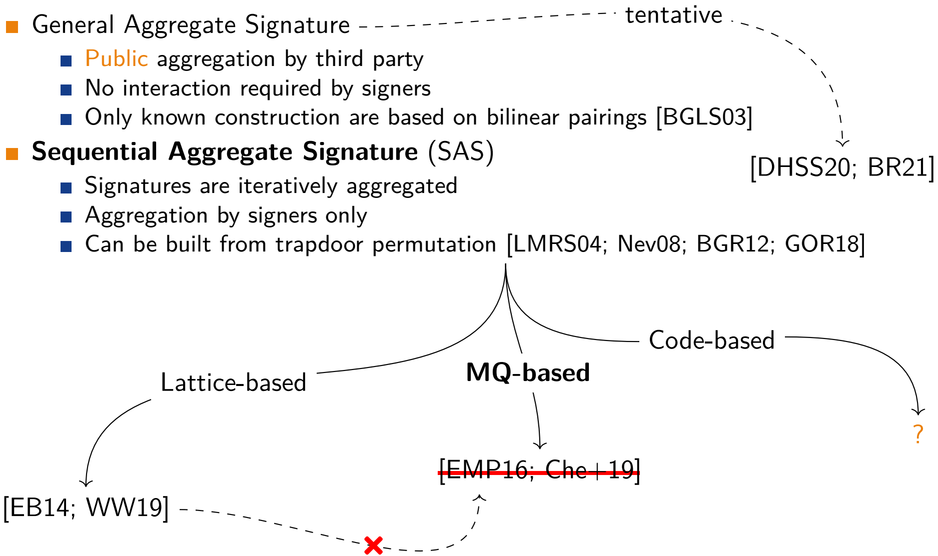




Can (S)AS be built from post-quantum assumptions?

Types of Aggregate Signature

- General Aggregate Signature
  - Public aggregation by third party
  - No interaction required by signers
  - Only known construction are based on bilinear pairings [BGLS03]
- **Sequential Aggregate Signature** (SAS)
  - Signatures are iteratively aggregated
  - Aggregation by signers only
  - Can be built from trapdoor permutation [LMRS04; Nev08; BGR12; GOR18]


tentative

[DHSS20; BR21]

Lattice-based

MQ-based

Code-based

[EB14; WW19]

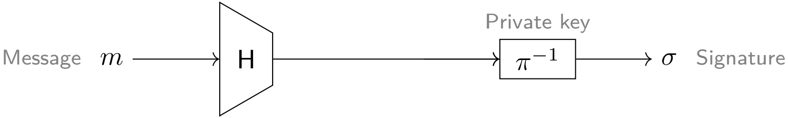
~~[EMP16; Che+19]~~

?

Can (S)AS be built from post-quantum assumptions? Yes, from lattices

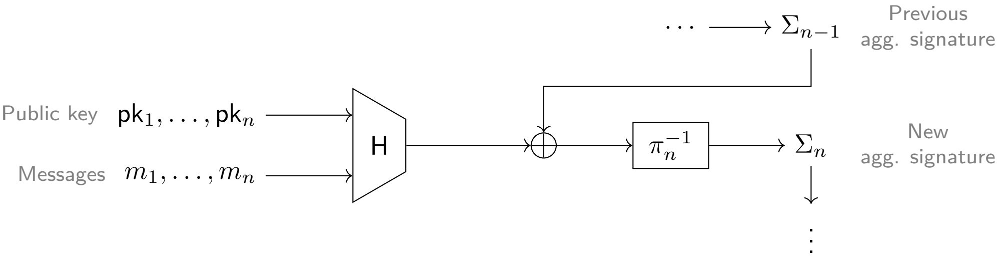
SAS from trapdoor permutation

Full Domain Hash (FDH) signature from trapdoor permutation $\pi\colon \mathcal{X} \to \mathcal{X}$ and opportune hash function $\mathsf{H}\colon \{0,1\}^* \to \mathcal{X}$.



SAS from trapdoor permutation

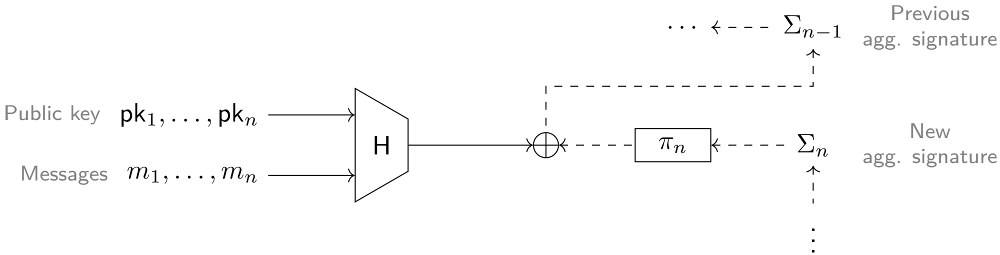
Full Domain Hash (FDH) signature from trapdoor permutation $\pi\colon \mathcal{X} \to \mathcal{X}$ and opportune hash function $\mathsf{H}\colon \{0,1\}^* \to \mathcal{X}$.



- **Aggregation** (simplified) [LMRS04; Nev08]: embed the previous aggregate signature into the new data to be signed

SAS from trapdoor permutation

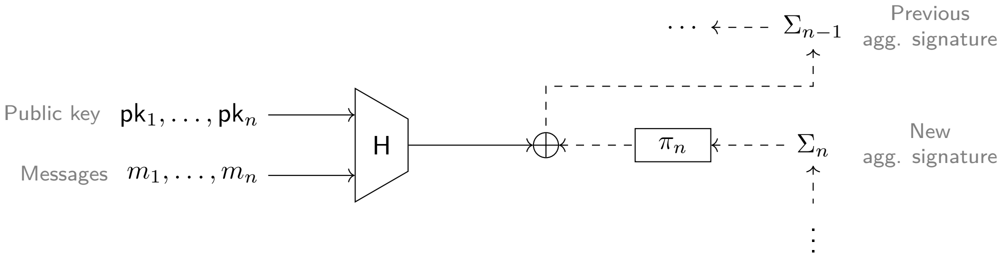
Full Domain Hash (FDH) signature from trapdoor permutation $\pi \colon \mathcal{X} \to \mathcal{X}$ and opportune hash function $\mathsf{H} \colon \{0,1\}^* \to \mathcal{X}$.



- **Aggregation** (simplified) [LMRS04; Nev08]: embed the previous aggregate signature into the new data to be signed
- **Verification**: recover each intermediate signature. Requires n steps of verification

SAS from trapdoor permutation

Full Domain Hash (FDH) signature from trapdoor permutation $\pi\colon \mathcal{X} \to \mathcal{X}$ and opportune hash function $\mathsf{H}\colon \{0,1\}^* \to \mathcal{X}$.



- **Aggregation** (simplified) [LMRS04; Nev08]: embed the previous aggregate signature into the new data to be signed
- **Verification**: recover each intermediate signature. Requires n steps of verification

Rigid transposition of FDH approach to post-quantum assumptions seems impractical

Trapdoor functions

A trapdoor function (TDF) T is a tuple of three algorithms $(\mathsf{TrapGen}, \mathsf{F}, \mathsf{I})$:

- $\mathsf{TrapGen}(1^\lambda)$: takes as input a security parameter 1^λ and generates an efficiently computable function $\mathsf{F}\colon \mathcal{X} \to \mathcal{Y}$ and a trapdoor I that allow to invert F.
- $\mathsf{F}(x)$: takes as input $x \in \mathcal{X}$ and outputs $\mathsf{F}(x) \in \mathcal{Y}$.
- $\mathsf{I}(y)$: takes as input $y \in \mathcal{Y}$ and outputs $x \in \mathcal{X}$ such that $\mathsf{F}(x) = y$ or it fails by returning $\perp$.

Trapdoor functions

A trapdoor function (TDF) T is a tuple of three algorithms $(\mathsf{TrapGen}, \mathsf{F}, \mathsf{I})$:

- $\mathsf{TrapGen}(1^\lambda)$: takes as input a security parameter 1^λ and generates an efficiently computable function $\mathsf{F}\colon \mathcal{X} \to \mathcal{Y}$ and a trapdoor I that allow to invert F.
- $\mathsf{F}(x)$: takes as input $x \in \mathcal{X}$ and outputs $\mathsf{F}(x) \in \mathcal{Y}$.
- $\mathsf{I}(y)$: takes as input $y \in \mathcal{Y}$ and outputs $x \in \mathcal{X}$ such that $\mathsf{F}(x) = y$ or it fails by returning $\perp$.

- When T is a permutation, the security of the FDH scheme is reduced to the one-wayness (OW) of T.
- Generic trapdoor functions lose uniformity properties and provable security with FDH.

Trapdoor functions

A trapdoor function (TDF) T is a tuple of three algorithms $(\mathsf{TrapGen}, \mathsf{F}, \mathsf{I})$:

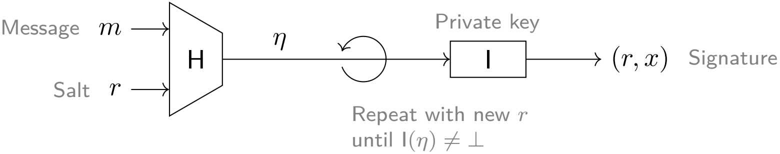
- $\mathsf{TrapGen}(1^\lambda)$: takes as input a security parameter 1^λ and generates an efficiently computable function $\mathsf{F}\colon \mathcal{X} \to \mathcal{Y}$ and a trapdoor I that allow to invert F.
- $\mathsf{F}(x)$: takes as input $x \in \mathcal{X}$ and outputs $\mathsf{F}(x) \in \mathcal{Y}$.
- $\mathsf{I}(y)$: takes as input $y \in \mathcal{Y}$ and outputs $x \in \mathcal{X}$ such that $\mathsf{F}(x) = y$ or it fails by returning $\perp$.

- When T is a permutation, the security of the FDH scheme is reduced to the one-wayness (OW) of T.
- Generic trapdoor functions lose uniformity properties and provable security with FDH.

We can regain provable security using the probabilistic hash-and-sign with retry approach.

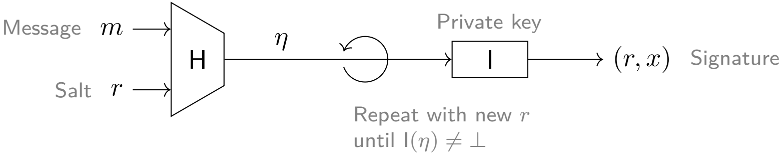
Probabilistic hash-and-sign with retry

Signature from trapdoor function (F, I) and opportune random oracle $\mathsf{H}\colon \mathcal{X} \to \mathcal{Y}$.



Probabilistic hash-and-sign with retry

Signature from trapdoor function (F, I) and opportune random oracle $\mathsf{H}\colon \mathcal{X} \to \mathcal{Y}$.



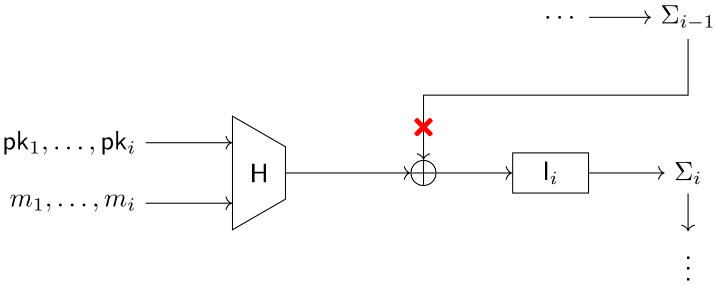
The security of the scheme is based on the one-wayness of F and the following additional property:

The output of the signing algorithm (r, x) is such that:

1. The salt r is indistinguishable from $r \leftarrow\!\!\$\ \{0,1\}^{\lambda}$.
2. The signature x is indistinguishable from $x \leftarrow\!\!\$\ \mathcal{X}$.

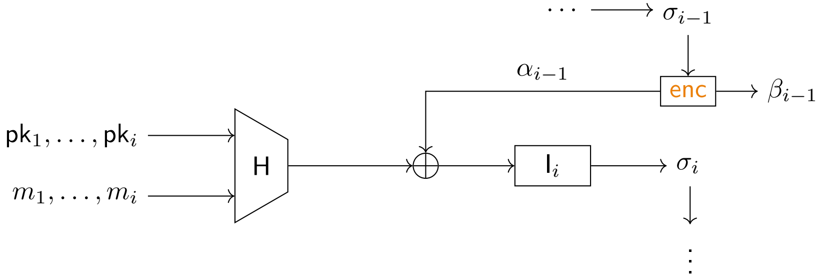
Generalize SAS schemes

Consider a generic trapdoor function (F, I).



Generalize SAS schemes

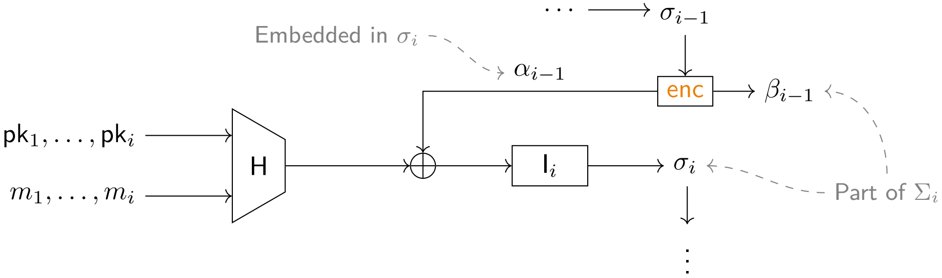
Consider a generic trapdoor function (F, I).



- Use an *efficient* encoding function $\mathsf{enc}\colon \mathcal{X} \to \mathcal{Y} \times \mathcal{X}'$ that splits σ_i as $\mathsf{enc}(\sigma_i) = (\alpha_i, \beta_i)$ [Nev08]

Generalize SAS schemes

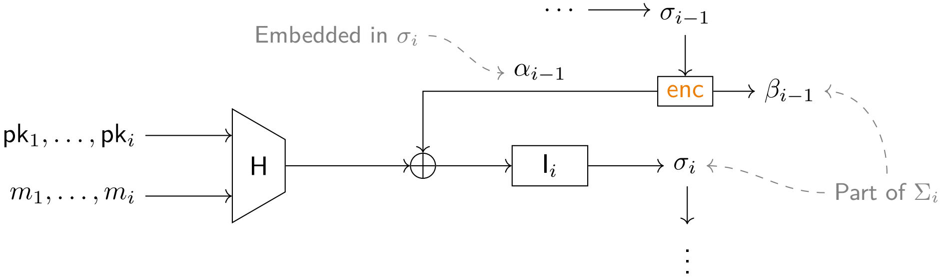
Consider a generic trapdoor function (F, I).



- Use an *efficient* encoding function enc: $\mathcal{X} \to \mathcal{Y} \times \mathcal{X}'$ that splits σ_i as $\mathsf{enc}(\sigma_i) = (\alpha_i, \beta_i)$ [Nev08]
- The aggregate signature is given by $\Sigma_n = (\beta_1, \ldots, \beta_{n-1}, \sigma_n)$
- [EMP16; Che+19] claim that this construction can be instantiated with every multivariate signature scheme

Generalize SAS schemes

Consider a generic trapdoor function (F, I).

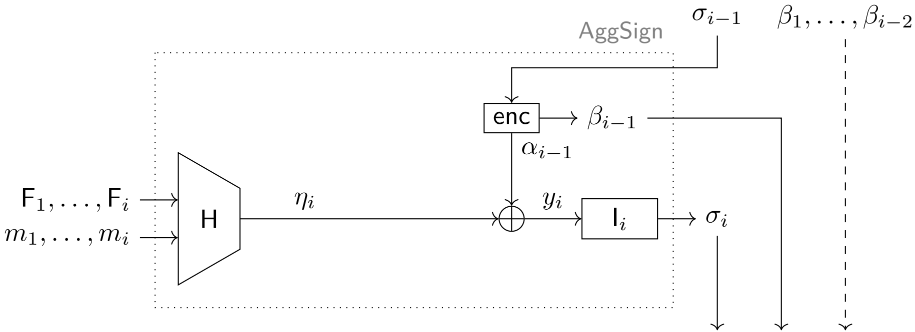


- Use an *efficient* encoding function enc: $\mathcal{X} \to \mathcal{Y} \times \mathcal{X}'$ that splits σ_i as enc$(\sigma_i) = (\alpha_i, \beta_i)$ [Nev08]
- The aggregate signature is given by $\Sigma_n = (\beta_1, \ldots, \beta_{n-1}, \sigma_n)$
- [EMP16; Che+19] claim that this construction can be instantiated with ~~every~~ multivariate signature scheme

Not with UOV!

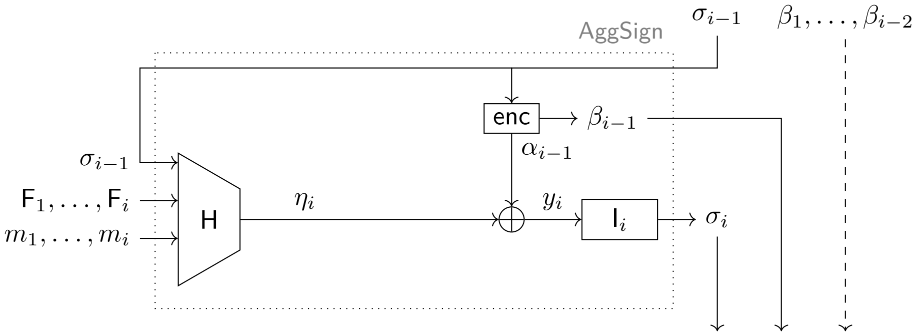
Steps to provable security

The following aggregate scheme is *not* provably secure (and sometimes provably *in*secure) with generic TDF.



Steps to provable security

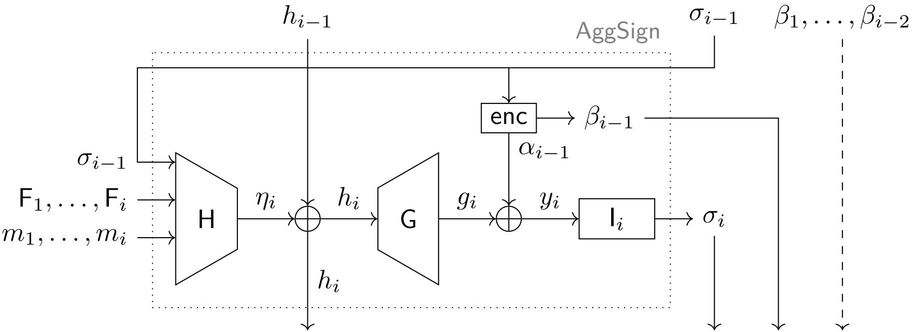
The following aggregate scheme is *not* provably secure (and sometimes provably *in*secure) with generic TDF.



- F_i is not injective and aggregate signatures are not unique on fixed input.

Steps to provable security

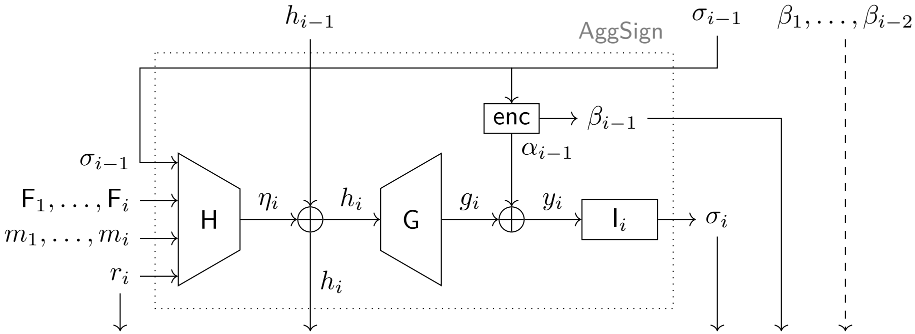
The following aggregate scheme is *not* provably secure (and sometimes provably *in*secure) with generic TDF.



- F_i is not injective and aggregate signatures are not unique on fixed input.
- If σ_{i-1} is part of the input to H it is not possible to directly retrieve it during verification.

Steps to provable security

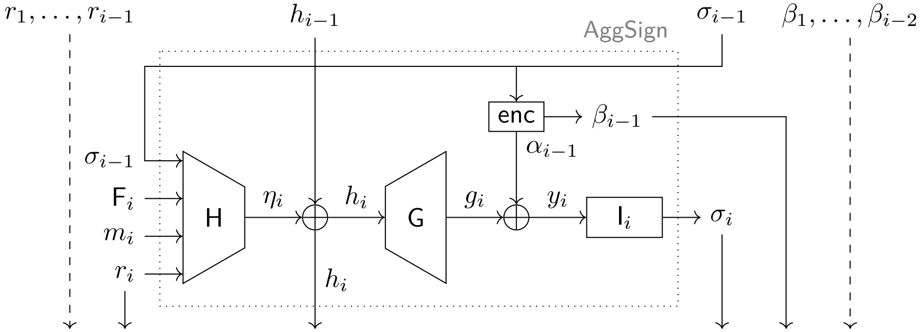
The following aggregate scheme is *not* provably secure (and sometimes provably *in*secure) with generic TDF.



- F_i is not injective and aggregate signatures are not unique on fixed input.
- If σ_{i-1} is part of the input to H it is not possible to directly retrieve it during verification.
- Aborts on I_i may leak information.

A secure SAS scheme

The following aggregate scheme is provably secure in the random oracle model with generic TDF. Let H: $\{0,1\}^* \to \{0,1\}^{2\lambda}$, G: $\{0,1\}^{2\lambda} \to \mathcal{Y}$ be random oracles.



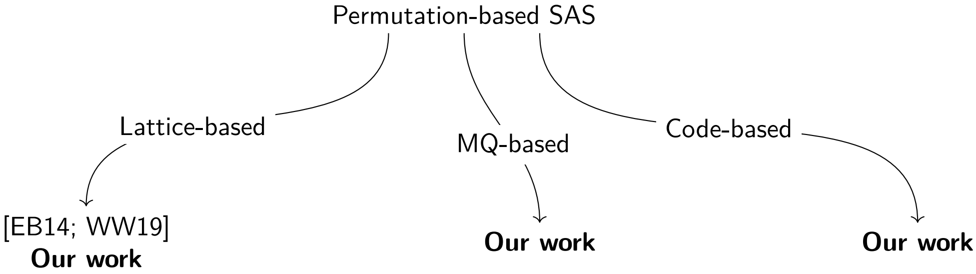
Compared with the previous construction

- Good: is provable secure (but not fully *black-box*)
- Good: is an **history-free** sequential aggregate signature scheme.
- Bad: the full n party signature has an overhead of length $2\lambda + n\lambda$.

Wrapping up

- Many post-quantum trapdoor signature are built from the hash-and-sign with retry approach.
- The same issues regarding provable security are also encountered for aggregated signatures.
- Inability to extend the naive FDH demonstration is the reason why simple constructions of aggregate signatures are not provable secure.

ia.cr/2023/784



References i

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. “Aggregate and Verifiably Encrypted Signatures from Bilinear Maps”. In: *EUROCRYPT 2003*. Ed. by Eli Biham. Vol. 2656. LNCS. Springer, Heidelberg, May 2003, pp. 416–432.

[BGR12] Kyle Brogle, Sharon Goldberg, and Leonid Reyzin. “Sequential Aggregate Signatures with Lazy Verification from Trapdoor Permutations - (Extended Abstract)”. In: *ASIACRYPT 2012*. Ed. by Xiaoyun Wang and Kazue Sako. Vol. 7658. LNCS. Springer, Heidelberg, Dec. 2012, pp. 644–662.

[BR21] Katharina Boudgoust and Adeline Roux-Langlois. *Compressed Linear Aggregate Signatures Based on Module Lattices*. Cryptology ePrint Archive, Report 2021/263. `https://eprint.iacr.org/2021/263`. 2021.

[Che+19] Jiahui Chen, Jie Ling, Jianting Ning, Zhiniang Peng, and Yang Tan. “MQ Aggregate Signature Schemes with Exact Security Based on UOV Signature”. In: *Information Security and Cryptology - 15th International Conference, Inscrypt 2019, Nanjing, China, December 6-8, 2019, Revised Selected Papers*. Ed. by Zhe Liu and Moti Yung. Vol. 12020. Lecture Notes in Computer Science. 2019, pp. 443–451.

References ii

[DHSS20] Yarkın Doröz, Jeffrey Hoffstein, Joseph H. Silverman, and Berk Sunar. *MMSAT: A Scheme for Multimessage Multiuser Signature Aggregation*. Cryptology ePrint Archive, Report 2020/520. `https://eprint.iacr.org/2020/520`. 2020.

[EB14] Rachid El Bansarkhani and Johannes Buchmann. "Towards Lattice Based Aggregate Signatures". In: *AFRICACRYPT 14*. Ed. by David Pointcheval and Damien Vergnaud. Vol. 8469. LNCS. Springer, Heidelberg, May 2014, pp. 336–355.

[EMP16] Rachid El Bansarkhani, Mohamed Saied Emam Mohamed, and Albrecht Petzoldt. "MQSAS - A Multivariate Sequential Aggregate Signature Scheme". In: *ISC 2016*. Ed. by Matt Bishop and Anderson C. A. Nascimento. Vol. 9866. LNCS. Springer, Heidelberg, Sept. 2016, pp. 426–439.

[GOR18] Craig Gentry, Adam O'Neill, and Leonid Reyzin. "A Unified Framework for Trapdoor-Permutation-Based Sequential Aggregate Signatures". In: *PKC 2018, Part II*. Ed. by Michel Abdalla and Ricardo Dahab. Vol. 10770. LNCS. Springer, Heidelberg, Mar. 2018, pp. 34–57.

References iii

[LMRS04] Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. "Sequential Aggregate Signatures from Trapdoor Permutations". In: *EUROCRYPT 2004*. Ed. by Christian Cachin and Jan Camenisch. Vol. 3027. LNCS. Springer, Heidelberg, May 2004, pp. 74–90.

[Nev08] Gregory Neven. "Efficient Sequential Aggregate Signed Data". In: *EUROCRYPT 2008*. Ed. by Nigel P. Smart. Vol. 4965. LNCS. Springer, Heidelberg, Apr. 2008, pp. 52–69.

[WW19] Zhipeng Wang and Qianhong Wu. "A Practical Lattice-Based Sequential Aggregate Signature". In: *ProvSec 2019*. Ed. by Ron Steinfeld and Tsz Hon Yuen. Vol. 11821. LNCS. Springer, Heidelberg, Oct. 2019, pp. 94–109.

Telsy

Thank you for your attention

Politecnico
di Torino