Thresholding Post-Quantum Signatures

A survey of techniques

Francesco De Sclavis [1,2] Matteo Nardelli [1] Marco Pedicini [2]

[1]IT Department, Bank of Italy

[2]Department of Mathematics and Physics, Roma Tre University

Cryptographic Tools for Blockchains 2026

Introduction

Threshold signature: requires at least t out of n parties to sign

- remove single point of failure
- decentralize trust

Example applications:

- joint payments
- consensus algorithms
- secure wallets
- distributed management of keys or identities

This paper: survey of post-quantum techniques

Introduction

Desirable properties:

- functional interchangeability: same verification as base signature
  - no protocol changes required, e.g. for Bitcoin
- can deal with malicious parties: either (non-interactive) identifiable aborts or robustness
- untrusted setup (DKG)
- low communication costs/rounds ($\rightarrow$ speed/latency) and small pubkey/signature ($\rightarrow$ transaction size)

Constructing a (t, n)-scheme

Threshold template for Schnorr signature $z = r + c \cdot s$:

- use Shamir's secret sharing: split secret in shares $\{s_i\}_{i=1}^{n}$ such that $s = \sum_{i \in S} \lambda_i s_i$
- each party generates partial response $z_i = r_i + c\lambda_i s_i$; aggregate as $z = \sum_{i \in S} z_i$
- Verify as the base signature

Example: FROST is partially interactive (1 round with pre-processing) and has all properties

Constructing a (t, n)-scheme

Question: can we apply the same template to post-quantum signatures?

- Lattice-based: yes with caveats
- Group-action-based: similar template, but has limitations
- Hash-based: no

Lattice-based

1. Fiat-Shamir with Aborts:
   - Schnorr-like id scheme + Fiat-Shamir, based on *SIS/LWE*
   - Output $(c, \mathbf{z})$ where
   $$\mathbf{z} = \mathbf{r} + c \cdot \mathbf{s}$$
   - Abort if $\mathbf{z}$ not short (rejection sampling)
2. GPV framework (using gadget trapdoor):
   - Find short pre-image $\mathbf{z}$: $\mathbf{A}\,\mathbf{z} = \mathbf{H}(m)$
   - Use secret $\mathbf{S}$ to sample solution (trapdoor sampling) and get
   $$\mathbf{z} = \mathbf{r} + \mathbf{S}\,\mathbf{c}$$
   - Different role for $\mathbf{c}$, but same formal expression for $\mathbf{z}$

Lattice-based: noise flooding

Problem: Rejection sampling and trapdoor sampling are hard to distribute

Solution:

- Skip rejection sampling; replace trapdoor sampling with algebraic computation
- $\mathbf{z}$ now can leak info on $\mathbf{s}$: employ large noise $\mathbf{r}$ to hide (noise flooding)

Consequence: bigger modulus q required $\rightarrow$ bigger public keys and signatures

Lattice-based: dealing with Lagrange coefficients

Problem: Lagrange coefficients can be very large $\rightarrow \mathbf{z}_i$ leaks info on $\mathbf{s}_i$

Solutions:

a. One-time random masking (TRaccoon): pairs of users share private masks for $\mathbf{z}_i$, that cancel out on $\mathbf{z}$
   - Con: share $O(n^2)$ values privately, hinders identifiable aborts

b. Linear secret sharing with small reconstruction coefficients
   - Con: more shares to send ($\binom{n}{t}$ for replicated secret sharing)

Lattice-based: preventing malicious manipulation

Active security: prevent manipulation, lattices require shares to stay short

a. Everywhere-short SS (Hermine): closest to FROST (partially non-interactive, partial verification), with limited threshold t
b. Verifiable short SS (Pelican): robust DKG and signature with overhead of rounds and $3t$ active signers

Partial verification: provides non-interactive identifiable aborts

Lattice-based: other techniques

1. Partial lattice trapdoors: split trapdoor and find partial pre-images
   - Pro: non-interactive, partial verification
   - Con: public key and signature grow with t
2. Linear hash functions: in FROST replace $s \mapsto g^s$ with a LHF
   - Pro: partially non-interactive
   - Con: requires SS with small reconstruction coefficients
3. Homomorphic commitments: perform computations on commitments to hide information on aborts
   - can do rejection sampling, used as building block (e.g. with MPC)

Group-action-based

Group action: map $\mathcal{G} \times \mathcal{E} \to \mathcal{E}$ compatible with operation of group $\mathcal{G}$:

- $[0]E = E$ for all $E \in \mathcal{E}$
- $[k][l]E = [k + l]E$ for all $k, l \in \mathcal{G}$ and $E \in \mathcal{E}$

Notation: from isogeny-based cryptography

Assumptions:

- efficiently computable
- exists unique solution s for $E_1 = [s]E_2$
- computing such s is hard (Group Action Inverse Problem)
  - $(s, g) \mapsto g^s$, $s \in \mathbb{Z}_q^*$ and g in cyclic group of order q, GAIP = DLP

Group-action-based: threshold evaluation

Schnorr-like signature based on GAIP:

- Schnorr template, but replace exponentiation with group action
- Threshold evaluation of group action:

$$[r_1 + \cdots + r_k]E_0 = [r_k][r_1 + \cdots + r_{k-1}]E_0$$

Computations are not independent but sequential

- many rounds of communication
- partial verification interactive

Group-action-based: properties

Isogeny-based instantiation (Threshold CSI-FiSh):

- very small public keys and signatures
- high computation and communication costs (especially with DKG)
- reduce costs in exchange for increased public key

GRASS framework:

- non-abelian group actions, sequential operations have an order
- can have code-based instantiation
- relies on replicated SS or multiplicative non-abelian SS (GRASS+)

Hash-based

Stateful HBS: OTS + Merklee tree authentication path

Threshold version:

- Cannot apply previous template
- Setup: for fixed leaf, split secrets and path as $x = \bigoplus_{i=1}^{n} x_i \oplus \text{CRV}.x$
- Sign: aggregator uses shares and public CRV.x for reconstruction
- Coalitions with unique identifier for (t, n) case

Limitations: requires trusted setup, unfeasible for stateless, number of coalitions grows as $\binom{n}{t}$

General techniques

1. Threshold FHE: Evaluate signing circuit with FHE, output partial decryptions
   - 1 round, applied for FHE-friendly lattice-based
   - Black-box approach: expensive
2. Multi-party computation: Evaluate signing algorithm with MPC
   - Active security, modular, applied for MPC-friendly lattice-based
   - Black-box approach: expensive, feasible for UOV-based
3. STARKs: Prove t signatures verify correctly
   - Transparent setup, applied in OTS with ad hoc representation
   - No functional interchangeability, signature grows with t

Conclusion

- Lattice-based: most mature, suited for most applications
  - Closest to FROST but not drop-in replacement
- Isogeny-based (group actions): small sizes, but intensive computation and sequential evaluation
- Hash-based: not based on hard problems, least suited for threshold
- General frameworks: good starting point, but requires tweaking

Next step: full SoK comparison