LaTSEC: Lattice-Based Threshold Signature from Expander Codes

Identifiable Abort from Everywhere-Short Secret Sharing over $\mathbb{Z}$-Modules

M. Arzaki[1], M. F. Esgin[1], M.-J. O. Saarinen[2], A. Sakzad[1], R. Steinfeld[1]

[1]Faculty of Information Technology, Monash University, Australia

[2]Faculty of Information Technology and Communication Sciences, Tampere University, Finland

Presented on 2026-October-06 at TCPT3

NIST Event for Threshold Call Preview Talks #3

Updated on 2026-10-12 (parameters and sizes aligned with the preprint)





Executive Summary

Two related submissions to the NIST Threshold Call

- SEC (S7, gadget): everywhere-short secret sharing over $\mathbb{Z}$-modules from lossless expanders
  - short shares, small integer reconstruction coefficients
  - 1 share per participant, independent of n, k_p, k_c
  - price: ramp access structure, $k_p < k_c$
- LaTSEC (S1, signing): 3-round Raccoon-style threshold signature from SEC
  - native non-interactive identifiable abort (NI-IA): 1 public abort key per participant, no ZKP
  - secure under MLWE and MSIS
- $n = 60$, $\kappa = 128$:
  - $|\mathsf{sk}_i| = 14.27$ KiB: $\geq 49\times$ smaller than Hermine*
  - $|\mathsf{ak}_i| = 24.00$ KiB: $\geq 85\times$ smaller than Hermine*

*Hermine extrapolated to $n = 60$, outside its parameter-selection range.

Outline

1. Raccoon-style Lattice Signature
2. Threshold Lattice-Based Signatures
3. SEC: The Sharing Gadget
4. LaTSEC: The Threshold Signature
5. Future Plans

Raccoon-style Lattice Signature

Schnorr-like signature over lattices, without rejection sampling [Lyu12, dPKPR24, dPKM+24]

KeyGen

$\boldsymbol{A} := \begin{bmatrix} \boldsymbol{I}_p & \boldsymbol{A}' \end{bmatrix}$, $\boldsymbol{A}' \xleftarrow{\$} \mathcal{R}_q^{p \times (\ell - p)}$; $\boldsymbol{s} \leftarrow \mathcal{D}^{\ell}_{\sigma_{\boldsymbol{s}}}$;
$\boldsymbol{t} := \boldsymbol{A}\boldsymbol{s}$
$\mathsf{sk} := \boldsymbol{s}$, $\mathsf{vk} := (\boldsymbol{A}, \boldsymbol{t})$

Sign(sk, msg)

$\boldsymbol{r} \leftarrow \mathcal{D}^{\ell}_{\sigma_{\boldsymbol{r}}}$; $\boldsymbol{w} := \boldsymbol{A}\boldsymbol{r}$; $c := H_c(\mathsf{vk}, \mathsf{msg}, \boldsymbol{w})$
$\boldsymbol{z} := c\boldsymbol{s} + \boldsymbol{r}$; output $(c, \boldsymbol{z})$

Verify(vk, msg, $(c, \boldsymbol{z})$)

$\boldsymbol{w}' := \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}$; accept iff $c = H_c(\mathsf{vk}, \mathsf{msg}, \boldsymbol{w}')$ and $\|\boldsymbol{z}\|_2 \leq B_2$

Correctness.
$\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} = \boldsymbol{A}(c\boldsymbol{s} + \boldsymbol{r}) - c\boldsymbol{A}\boldsymbol{s} = \boldsymbol{A}\boldsymbol{r} = \boldsymbol{w}$.

Security, in one picture.

- $\boldsymbol{t} = \boldsymbol{A}'\boldsymbol{s}' + \boldsymbol{e}$ is a systematic MLWE sample.
- $\boldsymbol{z} = c\boldsymbol{s} + \boldsymbol{r}$ is a **noisy linear hint** on $\boldsymbol{s} \Rightarrow$ Hint-MLWE [KLSS23].
- A wide $\boldsymbol{r}$ floods $c\boldsymbol{s}$: no rejection sampling, so no restarts when thresholdized.

Keep in mind

(1) $\boldsymbol{z}$ must be **short**; (2) signing leaks **noisy linear hints** on the key.

Thresholdizing Raccoon

Sharing a short signing key

$$\boldsymbol{s} = \sum_{i \in \mathsf{act}} \lambda_i \boldsymbol{s}_i, \qquad \boldsymbol{z}_i := c\boldsymbol{s}_i + \boldsymbol{r}_i, \qquad \boldsymbol{z} := \sum_{i \in \mathsf{act}} \lambda_i \boldsymbol{z}_i \qquad \Longrightarrow \qquad \|\boldsymbol{z}\| \leq \|\boldsymbol{\lambda}\|_1 \cdot \max_i \|\boldsymbol{z}_i\|$$

Shamir [Sha79]
$\boldsymbol{s}_i \xleftarrow{\$} \mathcal{R}_q^\ell$, $\lambda_i \in \mathbb{Z}_q$ large $\Rightarrow$ $\boldsymbol{z}_i$ long $\Rightarrow$ mask $\boldsymbol{z}_i$ [dPKM+24]
$\Rightarrow$ IA needs extra machinery [ENP24, dPKN+25, GHK+25]

Everywhere-short [dPENP24, BCdP+26]
$\|\boldsymbol{s}_i\|$, $\|\boldsymbol{\lambda}\|$ small $\Rightarrow$ $\|\boldsymbol{z}_i\| \leq B_1$ $\Rightarrow$ check $\boldsymbol{A}\boldsymbol{z}_i - c\boldsymbol{t}_i \stackrel{?}{=} \boldsymbol{w}_i$ $(\boldsymbol{t}_i := \boldsymbol{A}\boldsymbol{s}_i)$
$\Rightarrow$ **native NI-IA**

Thresholdizing Raccoon

Sharing a short signing key

$$\boldsymbol{s} = \sum_{i \in \mathsf{act}} \lambda_i \boldsymbol{s}_i, \qquad \boldsymbol{z}_i := c\boldsymbol{s}_i + \boldsymbol{r}_i, \qquad \boldsymbol{z} := \sum_{i \in \mathsf{act}} \lambda_i \boldsymbol{z}_i \qquad \Longrightarrow \qquad \|\boldsymbol{z}\| \le \|\boldsymbol{\lambda}\|_1 \cdot \max_i \|\boldsymbol{z}_i\|$$

Shamir [Sha79]

$\boldsymbol{s}_i \xleftarrow{\$} \mathcal{R}_q^\ell$, $\lambda_i \in \mathbb{Z}_q$ large $\Rightarrow$ $\boldsymbol{z}_i$ long $\Rightarrow$ mask $\boldsymbol{z}_i$ [dPKM+24]

$\Rightarrow$ IA needs extra machinery [ENP24, dPKN+25, GHK+25]

Everywhere-short [dPENP24, BCdP+26]

$\|\boldsymbol{s}_i\|$, $\|\boldsymbol{\lambda}\|$ small $\Rightarrow$ $\|\boldsymbol{z}_i\| \le B_1$ $\Rightarrow$ check $\boldsymbol{A}\boldsymbol{z}_i - c\boldsymbol{t}_i \stackrel{?}{=} \boldsymbol{w}_i$ $(\boldsymbol{t}_i := \boldsymbol{A}\boldsymbol{s}_i)$

$\Rightarrow$ **native NI-IA**

Thresholdizing Raccoon

Sharing a short signing key

$$\boldsymbol{s} = \sum_{i \in \mathsf{act}} \lambda_i \boldsymbol{s}_i, \qquad \boldsymbol{z}_i := c\boldsymbol{s}_i + \boldsymbol{r}_i, \qquad \boldsymbol{z} := \sum_{i \in \mathsf{act}} \lambda_i \boldsymbol{z}_i \qquad \Longrightarrow \qquad \|\boldsymbol{z}\| \le \|\boldsymbol{\lambda}\|_1 \cdot \max_i \|\boldsymbol{z}_i\|$$

Shamir [Sha79]

$\boldsymbol{s}_i \xleftarrow{\$} \mathcal{R}_q^\ell$, $\lambda_i \in \mathbb{Z}_q$ large $\Rightarrow$ $\boldsymbol{z}_i$ long $\Rightarrow$ mask $\boldsymbol{z}_i$ [dPKM+24]

$\Rightarrow$ IA needs extra machinery [ENP24, dPKN+25, GHK+25]

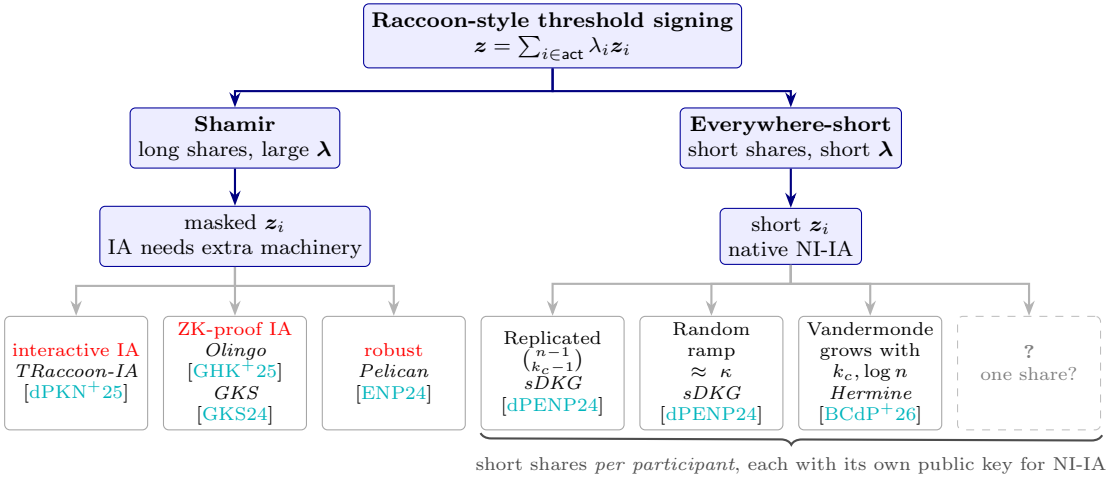
Everywhere-short [dPENP24, BCdP+26]

$\|\boldsymbol{s}_i\|$, $\|\boldsymbol{\lambda}\|$ small $\Rightarrow$ $\|\boldsymbol{z}_i\| \le B_1$ $\Rightarrow$ check $\boldsymbol{A}\boldsymbol{z}_i - c\boldsymbol{t}_i \stackrel{?}{=} \boldsymbol{w}_i$ ($\boldsymbol{t}_i := \boldsymbol{A}\boldsymbol{s}_i$)

$\Rightarrow$ **native NI-IA**

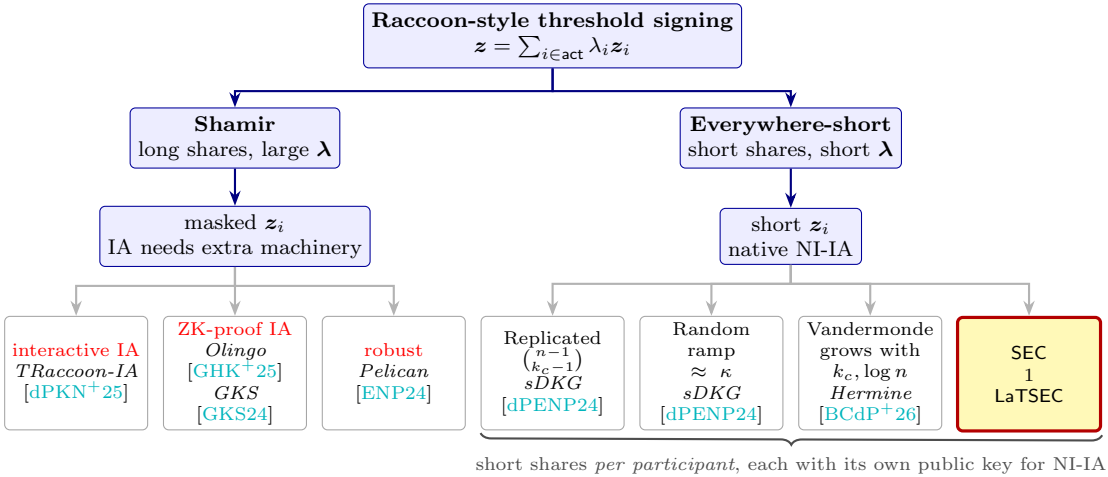
Design Choices

Which sharing layer, and what it costs per participant



Design Choices

Which sharing layer, and what it costs per participant



SEC: Coding-Theoretic Sharing over $\mathbb{Z}$

Brief description

- Follows the linear-algebraic and code-based approach to secret sharing of McEliece and Sarwate [MS81], developed through span programs and black-box secret sharing [KW93, CF02, CX20, ANP23].
- Works over a $\mathbb{Z}$-module: secrets, shares, and reconstruction coefficients are **integer objects**.

$\mathsf{Deal}_{\mathsf{pp}}(\mathbf{s})$

$\mathbf{S} := \mathbf{G}\begin{bmatrix}\mathbf{s}^\top \\ \mathbf{R}\end{bmatrix}$, with $\mathbf{G} = \begin{bmatrix}\mathbf{g} & \hat{\mathbf{G}}\end{bmatrix} \in \mathbb{Z}^{n\times k}$:

$\mathbf{g}$ carries the secret, $\hat{\mathbf{G}}$ (from an expander) spreads the short Gaussian masks $\mathbf{R}$;
party i receives the row $\mathbf{s}_i^\top := \mathbf{S}[i,\cdot]$.

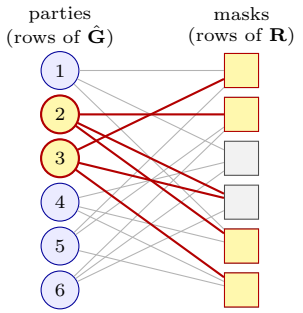
$\mathsf{Rec}_{\mathsf{pp}}(T, (\mathbf{s}_i)_{i\in T})$

Find a **short** $\boldsymbol{\lambda}_T \in \mathbb{Z}^{|T|}$ with $\boldsymbol{\lambda}_T^\top \mathbf{G}_T = \mathbf{e}_1^\top$ (row-HNF, then LLL + Babai + greedy);
output $(\boldsymbol{\lambda}_T^\top \mathbf{S}_T)^\top = \mathbf{s}$.

Technical contribution: design $\mathbf{G}$ so that shares are short, $\boldsymbol{\lambda}_T$ is short, and small coalitions learn nothing useful.

Why a Lossless Expander?

$\hat{\mathbf{G}} = (\mathbf{PH})^\top$, where $\mathbf{H}$ is the bi-adjacency matrix of a lossless bipartite expander



Toy: $n = m = 6$, $c_{\text{exp}} = 3$, $\gamma = \epsilon_{\text{exp}} = 1/3$
$\Rightarrow k_p = 2$.

$U = \{2, 3\}$: 4 unique neighbours (highlighted).

Sparsity $\Rightarrow$ short shares (c_{exp} entries ± 1 per row of $\hat{\mathbf{G}}$):

$$\mathbf{s}_i^\top = g_i\, \mathbf{s}^\top + \textstyle\sum_{j \in N(i)} \pm \mathbf{R}[j, \cdot], \qquad \|\mathbf{s}_i\|_\infty \le \beta_{\mathbf{g}} \beta_{\mathbf{s}} + c_{\text{exp}} \beta_{\mathbf{R}}.$$

Expansion $\Rightarrow$ masks cover small coalitions.

- Unique neighbors $\Rightarrow$ peeling: $\hat{\mathbf{G}}_U$ contains an element of $GL_{|U|}(\mathbb{Z})$ if $|U| \le k_p$.
- $\text{Col}_{\mathbb{Z}}(\hat{\mathbf{G}}_U) = \mathbb{Z}^{|U|} \Rightarrow$ no integer/modular cancellation of the masks.

Two thresholds.

- $k_p = \lceil 2(1 - \epsilon_{\text{exp}}) \lfloor \gamma n \rfloor \rceil - 1$ (certified expansion).
- k_c: random sets of size $\ge k_c$ admissible, $\mathbf{e}_1^\top \in \text{Row}_{\mathbb{Z}}(\mathbf{G}_T)$, w.p. ≥ 0.99.

SEC: Privacy

Computational, reduced to MLWE

- Short shares are inherently leaky ⇒ privacy is **computational**.
- Adversary's view: { shares of any coalition U with $|U| \leq k_p$} + {the noisy linear leakage of every signing query $(\boldsymbol{z}_i = c\boldsymbol{s}_i + \boldsymbol{r}_i)$}.
- We model the privacy of SEC using MatrixHint-MLWE [KLSS23, ENP24, WSS+25].

MLWE hard ⇒ MatrixHint-MLWE hard ⇒ SEC private

Comparison of Sharing Layers

Secret sharing schemes used in lattice-based threshold signatures

Sharing scheme	Used in	Access	Shares per participant	Short	Coeff.	Privacy
Shamir [Sha79]	TRaccoon [dPKM+24], Pelican [ENP24], TRaccoon-IA [dPKN+25], Olingo [GHK+25]	threshold	1	×	large	perfect
Benaloh–Leichter + Boppana [BL90, Bop85]	two-round line [CATZ24, ZT25]	threshold	$O(\mu^{4.3} \log n)$[a]	×[b]	$\{0, \pm 1\}$	perfect[b]
Replicated [dPENP24]	sDKG [dPENP24]	threshold	$\binom{n-1}{k_c-1}$	✓	binary	comput.
Random-assignment ramp [dPENP24]	sDKG [dPENP24]	ramp	typically κ	✓	binary	comput.
Vandermonde [DDCB94, BCdP+26]	Hermine [BCdP+26]	threshold	grows with k_c and $\log n$	✓	binary	comput.
SEC (this work)	LaTSEC (this work)	ramp	1	✓	small	comput.

[a] $\mu := \min\{k_c, n - k_c\}$ [BÇM21, CATZ24]. [b] As instantiated in [CATZ24, ZT25], the sharing randomness is uniform over $\mathcal{R}_q$: shares are not short and privacy is perfect. "Binary" coefficients lie in $\{0, 1\}$; "small" permits bounded signed integers.

SEC gains: 1 share per participant, short shares, bounded reconstruction coefficients.
Trade-off: ramp gap $k_p < k_c$.

LaTSEC

High-level description

Raccoon-style signature [dPKPR24, dPKM+24] with SEC as its sharing *and* aggregation layer.

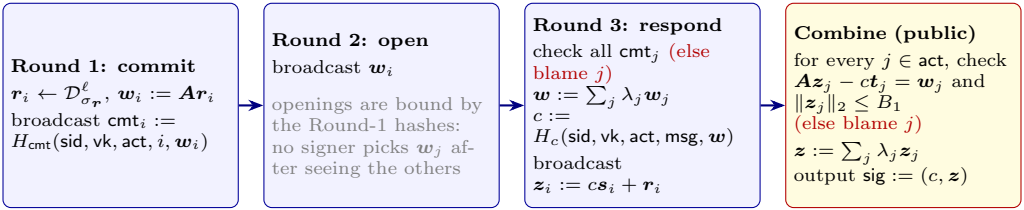
- **Trusted key generation:** $\boldsymbol{s} \leftarrow \mathcal{D}^{\ell}_{\sigma_{\boldsymbol{s}}}$, $(\boldsymbol{s}_i)_{i\in[n]} \leftarrow \mathsf{SEC.Deal}_{\mathsf{pp}}(\boldsymbol{s})$;
  $\mathsf{vk} := (\boldsymbol{A}, \boldsymbol{t} := \boldsymbol{A}\boldsymbol{s})$; $\mathsf{sk}_i := \boldsymbol{s}_i$; abort key $\mathsf{ak}_i := \boldsymbol{t}_i := \boldsymbol{A}\boldsymbol{s}_i$.
- **Thresholds:** unforgeability against $t_{\mathsf{uf}} := k_p$ corruptions; signing by an admissible act ($\mathbf{e}_1^{\top} \in \mathrm{Row}_{\mathbb{Z}}(\mathbf{G}_{\mathsf{act}})$), typically $|\mathsf{act}| \geq t_{\mathsf{sig}} := k_c$.
- $\mathsf{Rec}_{\mathsf{pp}}$ **is linear and commutes with public matrices** $\Rightarrow \boldsymbol{\lambda}_{\mathsf{act}}$ aggregates commitments, responses, and abort keys:

$$(\boldsymbol{s}_i)_i \mapsto \boldsymbol{s}, \qquad (\boldsymbol{t}_i)_i \mapsto \boldsymbol{t}, \qquad (\boldsymbol{w}_i)_i \mapsto \boldsymbol{w}, \qquad (\boldsymbol{z}_i)_i \mapsto \boldsymbol{z},$$

$$\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} = c\boldsymbol{A}\,\mathsf{Rec}_{\mathsf{pp}}(\mathsf{act}, (\boldsymbol{s}_i)_i) + \mathsf{Rec}_{\mathsf{pp}}(\mathsf{act}, (\boldsymbol{A}\boldsymbol{r}_i)_i) - c\boldsymbol{t} = \mathsf{Rec}_{\mathsf{pp}}(\mathsf{act}, (\boldsymbol{w}_i)_i) = \boldsymbol{w}.$$

LaTSEC: Three-Round Signing

Following the workflow of TRaccoon [dPKM+24] and [dPENP24]; authenticated broadcast, untrusted combiner

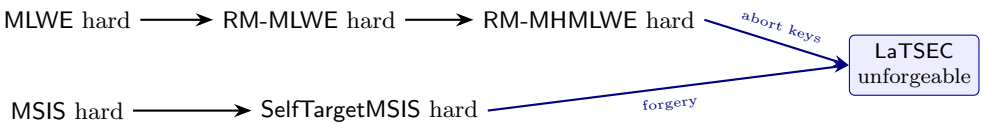


- **NI-IA**: 1 linear equation, 1 norm check per signer, against 1 abort key t_j. No ZKP/extra interaction [dPKN+25, GHK+25].
- IA $\Rightarrow$ verifiably incorrect values only.
- IA comes from the **smallness** of the shares and of λ_{act}.
- Correctness: follows from properties of $\mathsf{Rec}_{\mathsf{pp}}$ $\Rightarrow$ special case of IA (honest signers).

LaTSEC: Unforgeability

A sequence of hybrids, following [dPENP24]; at most $t_{\mathsf{uf}} = k_p$ corrupted parties

- Key steps of the hybrid argument:
  1. replace vk and the **complete abort-key tuple** by a simulated view (RM-MHMLWE);
  2. embed a SelfTargetMSIS instance: a forgery yields a solution.



- RM-MHMLWE: a new variant of MatrixHint-MLWE, reduced to standard MLWE (next slide).

A New Assumption Variant: RM-MHMLWE

Repeated-Matrix Matrix Hint-MLWE (dimensions omitted)

- Unforgeability proof ⇒ simulate all n abort keys consistently with the corrupted shares.
- $\boldsymbol{t}_i = g_i\,\boldsymbol{A}\boldsymbol{s} + \sum_j \hat{G}_{i,j}\,\boldsymbol{A}\boldsymbol{R}_j$ ⇒ k targets $\boldsymbol{A}\boldsymbol{s}, \boldsymbol{A}\boldsymbol{R}_1, \ldots, \boldsymbol{A}\boldsymbol{R}_{k-1}$ under the **same** $\boldsymbol{A}$.

RM-MHMLWE

Let $\overline{\boldsymbol{A}} := \begin{bmatrix}\boldsymbol{I} & \boldsymbol{A}\end{bmatrix}$ and $\overline{\boldsymbol{A}}^{\langle h\rangle} := \boldsymbol{I}_h \otimes \overline{\boldsymbol{A}}$ (block diagonal, h copies of $\overline{\boldsymbol{A}}$). Distinguish

$$\text{(Real)}\ \Big(\boldsymbol{A},\ \overline{\boldsymbol{A}}^{\langle h\rangle}\boldsymbol{s},\ (\boldsymbol{M}_j)_{j\in[\nu]},\ (\boldsymbol{z}_j)_{j\in[\nu]}\Big) \quad \text{from} \quad \text{(Random)}\ \Big(\boldsymbol{A},\ \boldsymbol{u},\ (\boldsymbol{M}_j)_{j\in[\nu]},\ (\boldsymbol{z}_j)_{j\in[\nu]}\Big),$$

where $\boldsymbol{u}$ is uniform and $\boldsymbol{z}_j := \boldsymbol{M}_j\boldsymbol{s} + \boldsymbol{y}_j$ are noisy hints on the *whole* stacked secret.

- $h = 1$: MatrixHint-MLWE [ENP24]. No hints: multi-secret MLWE [BLP+13, BJRLW21].
- MLWE hard ⇒ RM-MLWE hard ⇒ RM-MHMLWE hard.

Comparison with Lattice Threshold Signatures

Schemes with a mechanism for malicious signers, $\kappa = 128$, chronological order

Scheme	Max. n	$\lvert\mathsf{sk}_i\rvert$	$\lvert\mathsf{ak}_i\rvert$	$\lvert\mathsf{sig}\rvert$	Comm_i	Malicious signers	DKG
GKS [GKS24]	5	n/r	n/r	46.6	3000	proof IA	✓
Pelican [ENP24]	48	n/r	n/r	12.3	26.8+56t	robust	✓
sDKG, replicated [dPENP24]	16	$\geq$ 323.5 GiB*	$\geq$ 644.6 GiB*	10.3	20.0	NI-IA	✓
sDKG, ramp [dPENP24]	1024	$\geq$ 963.8*	$\geq$ 1920*	11.9	25.0	NI-IA	✓
TRaccoon-IA [dPKN+25]	1024	n/r	n/r	12.7	88.2+6.4t (28.2)	interactive IA	×
Olingo [GHK+25]	1024	n/r	n/a	11.3	852 (76)	proof IA	✓
Hermine [BCdP+26]	64	$\geq$ 699.2*	$\geq$ 2039.8*	11.3	73.2	NI-IA	×
LaTSEC (ours)	60	14.27	24.00	42.62	60.13	NI-IA	×

Sizes per participant in kB (LaTSEC in KiB, a 2.4% difference). Parenthesized Comm_i: optimistic execution. n/r: not reported; n/a: not applicable. *Extrapolated to $n = 60$ in [AES+26], privacy-matched at $t = k_p + 1 = 7$ and outside the schemes' parameter ranges (Hermine: selected for $n \leq 16$, $t \leq 8$; replicated sDKG: $n \leq 16$; ramp sDKG: optimistic, $\mu = \kappa$).

- LaTSEC: NI-IA, 1 share per party; $\lvert\mathsf{sk}_i\rvert$ is $\geq 49.0\times$ and $\lvert\mathsf{ak}_i\rvert$ is $\geq 85.0\times$ smaller than in Hermine.
- Trade-offs: $\lvert\mathsf{sig}\rvert$ is 3.4–4.2× larger (no bit dropping yet); three rounds; trusted dealer; ramp gap.

Future Plans

Limitations, open questions, and the submission package

Limitations and open questions

- **Ramp gap.** We aim for $k_p \geq 0.1n$; for this: $k < k_c \leq k + 6$ (k: columns of $\mathbf{G}$). Can it be narrowed?
- **Open problem.** Everywhere-short, $O(1)$ shares per participant, exact threshold.
- **Signature size.** Bit dropping/compression.
- **Trusted dealer** $\Rightarrow$ DKG for SEC shares.
- **Rounds.** 3 rounds $\Rightarrow$ 2 rounds?
- **Scale.** Exhaustive expansion check $\Rightarrow n > 60$?

Specification

- CS1: SEC (S7), standalone.
- CS2: LaTSEC (S1), with its conventional Raccoon-style core.

Code

- Available: SageMath toolkit for SEC and LaTSEC.
- Planned: Python reference implementation and test vectors; Rust performance implementation.

Thank You!

Thank You for Your Attention!

Preprint: https://sites.google.com/monash.edu/arzaki-phdresearch/home

Code: https://github.com/arzaki-cstheory/SEC-LaTSEC

Email: muhammad dot arzaki at monash dot edu

Appendix: Everywhere-Short Secret Sharing

Goal and question

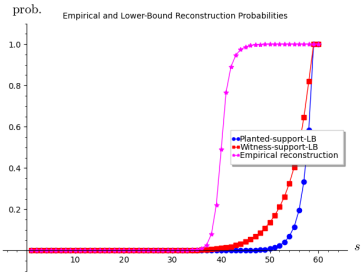
- **Everywhere-short** [dPENP24, BCdP+26]: short secret ⇒ short shares, short reconstruction vectors, and linear reconstruction.
- Prior constructions pay for shortness in the **number of short sub-shares per participant**: grows with n and k_c (or with κ).
- Under NI-IA, every sub-share carries public verification material:
  ⇒ multiplicity inflates both **secret-share storage** and **public abort-key storage**.

Question

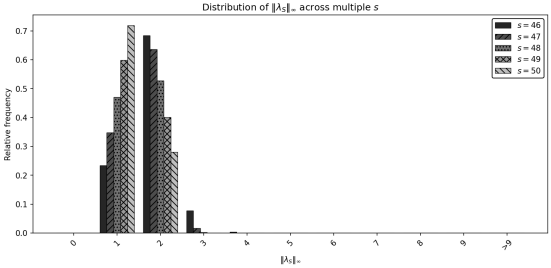
Is there an everywhere-short secret sharing scheme with **one share per participant**? At what cost?

Appendix: SEC in Practice

Selected instances ($n = 40$ and $n = 60$)



Probability that a random set of size s reconstructs ($n = 60$).



Distribution of $\|\boldsymbol{\lambda}_T\|_\infty$ for $|T| \in [46, 50]$ ($n = 60$).

n	k	c_{exp}	γ	ϵ_{exp}	k_p	k_c	k_c^{det}	$\beta_{\boldsymbol{\lambda}} = \max \|\boldsymbol{\lambda}_T\|_\infty$	$\omega_{\boldsymbol{\lambda}} = \max \text{wt}_H(\boldsymbol{\lambda}_T)$
40	20	3	0.125	0.480	5	26	35	4	24
60	40	3	0.100	0.450	6	46	56	8	42

$\beta_{\boldsymbol{\lambda}}, \omega_{\boldsymbol{\lambda}}$: observed maxima over the tested certified sets of size $\geq k_c$; k_c^{det}: every set of this size reconstructs.

Appendix: LaTSEC Parameters

Ring degree $d = 512$; sizes in KiB; lattice-estimator security estimates [AES+26]

n	κ	k_p	k_c	$\lceil \lg q \rceil$	(ℓ, p)	ω_c	$\lvert \mathsf{sk}_i \rvert$	$\lvert \mathsf{ak}_i \rvert$	$\lvert \mathsf{vk} \rvert$	$\lvert \mathsf{sig} \rvert$
40	128	5	26	61	(12,6)	30	13.883	22.875	22.906	40.750
40	192	5	26	63	(16,8)	44	19.029	31.500	31.531	56.147
40	256	5	26	64	(20,10)	60	23.952	40.000	40.031	70.981
60	128	6	46	64	(12,6)	30	14.273	24.000	24.031	42.622
60	192	6	46	66	(16,8)	44	19.547	33.000	33.031	58.641
60	256	6	46	66	(20,10)	60	24.599	41.250	41.281	74.098

Unforgeability holds against at most k_p corrupted parties; signing uses admissible sets of size at least k_c. Shares and signatures are ideal entropy-coded sizes; vk and abort keys are encoded without bit dropping.

- $n = 60$, $\kappa = 128 \Rightarrow \mathsf{Comm}_i = 60.13$ KiB, $\lvert \mathsf{ak} \rvert = 1440$ KiB.
- $\lvert \mathsf{sk}_i \rvert$ depends on κ and $\lg q$, not on k_p or k_c.

Appendix: SEC Privacy in More Detail

Hard-core local leakage resilience

- Short shares of a short secret may reveal partial information, so conventional privacy is too strong.
- What we protect: the **hard-core function** $f_{\boldsymbol{A}}(\boldsymbol{s}) := \boldsymbol{A}\boldsymbol{s} \bmod q$ (the verification key), given
  - the shares of any coalition U with $|U| \le k_p$, and
  - the noisy linear leakage released by every signing query ($\boldsymbol{z}_i = c\boldsymbol{s}_i + \boldsymbol{r}_i$).
- The whole view is 1 **stacked noisy linear hint** on the secret, with a jointly distributed perturbation.

MatrixHint-MLWE [KLSS23, ENP24, WSS+25]

Distinguish (Real) $\big(\boldsymbol{A},\ \overline{\boldsymbol{A}}\boldsymbol{s},\ (\boldsymbol{M}_i)_i,\ (\boldsymbol{z}_i)_i\big)$ from (Random) $\big(\boldsymbol{A},\ \boldsymbol{u},\ (\boldsymbol{M}_i)_i,\ (\boldsymbol{z}_i)_i\big)$,
where $\overline{\boldsymbol{A}} := \begin{bmatrix}\boldsymbol{I} & \boldsymbol{A}\end{bmatrix}$, $\boldsymbol{u}$ is uniform, and $\boldsymbol{z}_i := \boldsymbol{M}_i\boldsymbol{s} + \boldsymbol{y}_i$ are noisy linear hints.

Appendix: LaTSEC Identifiable Abort and Correctness

Both from one high-probability event $\mathsf{Good}_{\mathsf{tail}}$ (all Gaussian samples within their bounds)

Theorem (Identifiable abort)

If $B_1 \geq \omega_c \sqrt{D}\,(\beta_{\mathbf{g}}\beta_{\boldsymbol{s}} + c_{\exp}\beta_{\mathbf{R}}) + \beta_{\boldsymbol{r}}$, $B_2 \geq n\beta_{\boldsymbol{\lambda}} B_1$, *and* $\|\boldsymbol{\lambda}_{\mathsf{act}}\|_\infty \leq \beta_{\boldsymbol{\lambda}}$ *for every admissible* act, *then* $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{TS\text{-}IA}}(\kappa) \leq 3 \cdot 2^{-\kappa}$ *for every PPT* $\mathcal{A}$.

- Honest signers are never blamed: their openings match and $\|c\boldsymbol{s}_i + \boldsymbol{r}_i\|_2 \leq B_1$.
- Combine does not abort $\Rightarrow$ every accepted $\boldsymbol{z}_j$ passes the checks: $\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} = \boldsymbol{w}$ and $\|\boldsymbol{z}\|_2 \leq \sum_j |\lambda_j|\,\|\boldsymbol{z}_j\|_2 \leq n\beta_{\boldsymbol{\lambda}} B_1 \leq B_2$.

Correctness (corollary)

The non-aborting branch with all signers honest: every honest execution outputs an accepting signature, except with probability at most $3 \cdot 2^{-\kappa}$.

Appendix: Testing and Evaluation Plan

- Seeded, reproducible construction of $\mathbf{G}$; tests re-derive the published parameters.
- Admissible *and* inadmissible signer sets: the admissibility predicate must agree with $\mathsf{Rec}_{\mathsf{pp}}$.
- Malicious paths: inconsistent openings, out-of-bound $\boldsymbol{z}_i$, replayed sessions; the blamed party must be exactly the deviating one.
- Reports: timings, admissible-set fractions, and $\|\boldsymbol{\lambda}\|$ distributions.

References I

[AES+26] Muhammad Arzaki, Muhammed F. Esgin, Markku-Juhani O. Saarinen, Amin Sakzad, and Ron Steinfeld, *LaTSEC: Lattice-Based Threshold Signature Scheme from Expander Codes*, Preprint, October 2026.

[ANP23] Benny Applebaum, Oded Nir, and Benny Pinkas, *How to recover a secret with $O(n)$ additions*, Advances in Cryptology – CRYPTO 2023, Part I (Helena Handschuh and Anna Lysyanskaya, eds.), Lecture Notes in Computer Science, vol. 14081, Springer, 2023, pp. 236–262.

[BCdP+26] Giacomo Borin, Sofía Celi, Rafaël del Pino, Thomas Espitau, Shuichi Katsumata, Guilhem Niot, Thomas Prest, and Kaoru Takemure, *Hermine: An efficient lattice-based FROST-like threshold signature*, Advances in Cryptology – ASIACRYPT 2026, 2026, To appear.

[BÇM21] Marshall Ball, Alper Çakan, and Tal Malkin, *Linear threshold secret-sharing with binary reconstruction*, 2nd Conference on Information-Theoretic Cryptography (ITC 2021) (Stefano Tessaro, ed.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 199, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, pp. 12:1–12:22.

[BJRLW21] Katharina Boudgoust, Corentin Jeudy, Adeline Roux-Langlois, and Weiqiang Wen, *On the hardness of Module-LWE with binary secret*, Topics in Cryptology – CT-RSA 2021 (Kenneth G. Paterson, ed.), Lecture Notes in Computer Science, vol. 12704, Springer, 2021, pp. 503–526.

[BL90] Josh Cohen Benaloh and Jerry Leichter, *Generalized secret sharing and monotone functions*, Advances in Cryptology – CRYPTO '88 (Shafi Goldwasser, ed.), Lecture Notes in Computer Science, vol. 403, Springer, 1990, pp. 27–35.

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé, *Classical hardness of learning with errors*, Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC 2013) (Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, eds.), ACM, 2013, pp. 575–584.

[Bop85] Ravi B. Boppana, *Amplification of probabilistic Boolean formulas*, 26th Annual Symposium on Foundations of Computer Science (FOCS 1985), IEEE Computer Society, 1985, pp. 20–29.

[CATZ24] Rutchathon Chairattana-Apirom, Stefano Tessaro, and Chenzhi Zhu, *Partially non-interactive two-round lattice-based threshold signatures*, Advances in Cryptology – ASIACRYPT 2024, Part IV (Kai-Min Chung and Yu Sasaki, eds.), Lecture Notes in Computer Science, vol. 15487, Springer, 2024, pp. 268–302.

[CF02] Ronald Cramer and Serge Fehr, *Optimal black-box secret sharing over arbitrary Abelian groups*, Advances in Cryptology – CRYPTO 2002 (Moti Yung, ed.), Lecture Notes in Computer Science, vol. 2442, Springer, 2002, pp. 272–287.

References II

[CX20] Ronald Cramer and Chaoping Xing, *Blackbox secret sharing revisited: A coding-theoretic approach with application to expansionless near-threshold schemes*, Advances in Cryptology – EUROCRYPT 2020, Part I (Anne Canteaut and Yuval Ishai, eds.), Lecture Notes in Computer Science, vol. 12105, Springer, 2020, pp. 499–528.

[DDCB94] Yvo Desmedt, Giovanni Di Crescenzo, and Mike Burmester, *Multiplicative non-abelian sharing schemes and their application to threshold cryptography*, Advances in Cryptology – ASIACRYPT '94 (Josef Pieprzyk and Reihaneh Safavi-Naini, eds.), Lecture Notes in Computer Science, vol. 917, Springer, 1994, pp. 19–32.

[dPENP24] Rafaël del Pino, Thomas Espitau, Guilhem Niot, and Thomas Prest, *How to Shortly Share a Short Vector: DKG with Short Shares and Application to Lattice-Based Threshold Signatures with Identifiable Aborts*, Manuscript, 2024.

[dPKM+24] Rafaël del Pino, Shuichi Katsumata, Mary Maller, Fabrice Mouhartem, Thomas Prest, and Markku-Juhani O. Saarinen, *Threshold Raccoon: Practical threshold signatures from standard lattice assumptions*, Advances in Cryptology – EUROCRYPT 2024, Part II (Marc Joye and Gregor Leander, eds.), Lecture Notes in Computer Science, vol. 14652, Springer, 2024, pp. 219–248.

[dPKN+25] Rafaël del Pino, Shuichi Katsumata, Guilhem Niot, Michael Reichle, and Kaoru Takemure, *Unmasking TRaccoon: A lattice-based threshold signature with an efficient identifiable abort protocol*, Advances in Cryptology – CRYPTO 2025, Part VI (Yael Tauman Kalai and Seny F. Kamara, eds.), Lecture Notes in Computer Science, vol. 16005, Springer, 2025, pp. 423–456.

[dPKPR24] Rafaël del Pino, Shuichi Katsumata, Thomas Prest, and Mélissa Rossi, *Raccoon: A masking-friendly signature proven in the probing model*, Advances in Cryptology – CRYPTO 2024, Part I (Leonid Reyzin and Douglas Stebila, eds.), Lecture Notes in Computer Science, vol. 14920, Springer, 2024, pp. 409–444.

[ENP24] Thomas Espitau, Guilhem Niot, and Thomas Prest, *Flood and submerse: Distributed key generation and robust threshold signature from lattices*, Advances in Cryptology – CRYPTO 2024, Part VII (Leonid Reyzin and Douglas Stebila, eds.), Lecture Notes in Computer Science, vol. 14926, Springer, 2024, pp. 425–458.

[GHK+25] Kamil Doruk Gur, Patrick Hough, Jonathan Katz, Caroline Sandsbråten, and Tjerand Silde, *Olingo: Threshold lattice signatures with DKG and identifiable abort*, Cryptology ePrint Archive, Paper 2025/1789, 2025, To appear at ACM CCS 2026.

[GKS24] Kamil Doruk Gur, Jonathan Katz, and Tjerand Silde, *Two-round threshold lattice-based signatures from threshold homomorphic encryption*, Post-Quantum Cryptography – PQCrypto 2024, Part II (Markku-Juhani O. Saarinen and Daniel Smith-Tone, eds.), Lecture Notes in Computer Science, vol. 14772, Springer, 2024, pp. 266–300.

References III

[KLSS23] Duhyeong Kim, Dongwon Lee, Jinyeong Seo, and Yongsoo Song, *Toward practical lattice-based proof of knowledge from Hint-MLWE*, Advances in Cryptology – CRYPTO 2023, Part V (Helena Handschuh and Anna Lysyanskaya, eds.), Lecture Notes in Computer Science, vol. 14085, Springer, 2023, pp. 549–580.

[KW93] Mauricio Karchmer and Avi Wigderson, *On span programs*, Proceedings of the Eighth Annual Structure in Complexity Theory Conference, IEEE Computer Society, 1993, pp. 102–111.

[Lyu12] Vadim Lyubashevsky, *Lattice signatures without trapdoors*, Advances in Cryptology – EUROCRYPT 2012 (David Pointcheval and Thomas Johansson, eds.), Lecture Notes in Computer Science, vol. 7237, Springer, 2012, pp. 738–755.

[MS81] Robert J. McEliece and Dilip V. Sarwate, *On sharing secrets and Reed–Solomon codes*, Communications of the ACM **24** (1981), no. 9, 583–584.

[Sha79] Adi Shamir, *How to share a secret*, Communications of the ACM **22** (1979), no. 11, 612–613.

[WSS+25] Hongxiao Wang, Ron Steinfeld, Markku-Juhani O. Saarinen, Muhammed F. Esgin, and Siu-Ming Yiu, *mmCipher: Batching post-quantum public key encryption made bandwidth-optimal*, Cryptology ePrint Archive, Paper 2025/1000, 2025.

[ZT25] Chenzhi Zhu and Stefano Tessaro, *The algebraic one-more MISIS problem and applications to threshold signatures*, Advances in Cryptology – CRYPTO 2025, Part I (Yael Tauman Kalai and Seny F. Kamara, eds.), Lecture Notes in Computer Science, vol. 16000, Springer, 2025, pp. 548–581.