

“Preview Writeup”: In anticipation of a package submission to the NIST Threshold Call

Title: Thimza

Subtitle: Stateless Threshold Raccoon Signatures at Low Bandwidth

Version: Preview Writeup 0.2 (2026-10-11)¹

Team name: Thimza Team

Team members: Navid Abapour, Hamidreza Amini Khorasgani

Abstract: Thimza is a post-quantum threshold signature scheme from the Raccoon family, which is built upon Shamir Secret Sharing, Raccoon Signature, and the SHAKE256 pseudorandom function, with a signing protocol of three rounds that aims at active security with selective abort. Thimza seems to be the most time-efficient lattice-based threshold signature scheme, faster than Hermine (NIST-MPTS 2026), Ringtail (S&P'25), EKT (CRYPTO'24), TalonG (EUROCRYPT'26), ThRaccoon (EUROCRYPT'24), and the one with the lowest communication cost among interchangeable schemes.

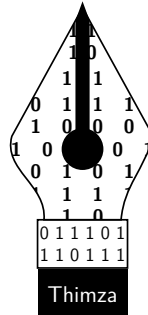
Three particular design decisions allow Thimza to outperform other protocols. First, Thimza employs the mask, which is a pseudorandom vector that sums to 0 and is constructed using seeds that are shared in pairs among signers. In the case of Thimza, each element of the partial signatures is masked by the corresponding mask element. Masks in Thimza are computed from the view of the transcript by the signer itself; therefore, forging a message only leads to aborting the execution, while the protocol does not require authenticated channels. Second, there is no registry of session identifiers, since a signer has to maintain just one token for each ongoing session. Third, Thimza does not employ any expensive cryptographic components (like zero-knowledge proofs) or complicated primitives (homomorphic encryption or lattice trapdoors). Regarding the bandwidth, Thimza can be applied for up to 1024 signers, transferring 16.3 KiB per signer (at the 128-bit post-quantum security level). At $t = 1024$, this is 1.6 times less than TalonG and 3.4 times less than TRaccoon, transferring 12.8 KiB signatures, and the communication cost is independent of the threshold.

Under the assumptions MLWE and MSIS, Thimza is TS-UF-0 secure even though the forgeries and alterations by the adversary result in the fact that the honest signers having different views of the transcript, and its single-signer version, the Raccoon signature, is EUF-CMA secure. The proofs of these facts are presented in our draft. They will also be included in our submission.

Proposed crypto-systems: Thimza, Threshold Signing over Module Lattices (Category S1.1).

Keywords: Threshold Cryptography; NIST Threshold Call; Threshold Signatures; Raccoon; Lattice-Based Cryptography; Post-Quantum Cryptography

¹Revised preliminary version submitted to NIST-MPTC for review



Preview writeup. This document is provided to NIST for online publication, to foster public awareness and support public discussion within the scope of the NIST First Call for Multi-Party Threshold Schemes [NIST-IR8214C]. This “preview writeup” represents a good-faith plan for a subsequent “package submission”. However, until the deadline for package submission, the team may still modify its own composition and the submission plan, including possible changes to the technical scope, and/or the used techniques or achieved results.

Team members: Navid Abapour^{i1,a1,*}, Hamidreza Amini Khorasgani^{i2,a2,*}

Open Researcher and Contributor Identifiers (ORCID):

i1 (0000-0003-4293-8518); i2 (0009-0007-5723-6910)

Affiliations:

^{a1} Surrey Centre for Cyber Security, University of Surrey @ UK

^{a2} Department of Computer Science, Purdue University @ USA

Associateship clarifications:

* Ph.D. Student.

Main contacts:

- **Team mailing list:** thimza@googlegroups.com
- **Primary technical contact person:** Navid Abapour, n.abapour@surrey.ac.uk
- **Secondary contact person:** Hamidreza Amini Khorasgani, haminikh@purdue.edu

Produced by humans. The team hereby confirms that the content in this preview writeup: (i) was produced by the team members, and (ii) was not produced by generative artificial intelligence (AI), with the possible exception of AI-proposed grammar improvements, minor integrated suggestions, or some well-identified and short localized portions of auxiliary content (e.g., some illustration); and (iii) was proofread by the team members.

1. Introduction

In this preview, we present Thimza², a post-quantum (PQ) threshold signature scheme over module lattices that we plan to submit to the NIST Threshold Call in Category S1 (Special Signing), and whose full description, proofs and evaluation are in our preliminary draft [AA26].

Thimza can be seen as a threshold version of the Raccoon signature scheme [PKPR24], itself an instance of Lyubashevsky’s Fiat–Shamir framework [Lyu12] in which rejection sampling is replaced by noise flooding. In Raccoon, the verification key $\mathbf{t}$ is a rounded noisy image of a short secret $\mathbf{s}$ under a public matrix $\mathbf{A}$ expanded from a seed. A signer commits to $\mathbf{w} = \mathbf{A}\mathbf{r} + \mathbf{e}'$ for fresh short $\mathbf{r}$, hashes the rounded commitment and the message into a short challenge c , and responds with $\mathbf{z} = c\mathbf{s} + \mathbf{r}$, whose wide randomness floods the secret. The verifier recomputes the rounded commitment from $\mathbf{A}\mathbf{z}$ and $c\mathbf{t}$, corrects it with a hint $\mathbf{h}$, and checks the hash and the shortness of $(\mathbf{z}, \mathbf{h})$. Security is based on Module Learning With Errors (MLWE) and self-target Module Short Integer Solution (MSIS). In Thimza, the signing key is shared with Shamir’s scheme over a cyclotomic ring, and any t (participation threshold) of the n key holders run a three-round protocol whose output is a Raccoon signature. Each signer hides its share of the response with a *mask*, a pseudorandom vector computed from seeds shared pairwise with the other signers, and the masks cancel in the sum. Thimza is interchangeable with Raccoon signatures, with regard to the conventional signature verification, since a relying party verifies a threshold signature with the unmodified Raccoon verifier.

The symmetric building blocks of Thimza, such as the expansion of the public matrix, the two transcript hashes, the challenge sampler and the pseudorandom function used for masking, are built from SHAKE256, standardized by NIST in FIPS 202 [Nat15]. Thimza supports large committees, of up to 1024 signers, as the Threshold Call [NIST-IR8214C] recommends, which is the range where the cost of a threshold protocol is dominated by what each signer must send and compute for every other signer.

Apart from optimal masking via an antisymmetric construction that achieves the lower bound of $t - 1$ evaluations of the pseudorandom function (PRF) per signer, Thimza removes the unbounded registry of Threshold Raccoon [PKMMPS24], where each signer stores every session identifier it has used, since a reused identifier would repeat a mask. By design, this solves the open problem of Threshold Raccoon. Thimza shows that session-bounded state, one token per open session, is both necessary and sufficient. For protocols with three rounds, such as Threshold Raccoon, Thimza compresses the message of the second round from 15.3 KiB to 4.1 KiB, since a tight parameter analysis lets each signer drop the $\nu_c = 36$ least significant bits of each commitment coefficient. At the 128-bit PQ security level and $t = 1024$ a signer sends 16.3 KiB over the whole protocol, the signature is 12.8 KiB and the verification key is 3.8 KiB.

We plan to submit: (i) a self-contained specification with algorithms, system model and concrete parameter sets for the three NIST PQ security levels; (ii) an open source reference implementation with benchmarks, known answer tests and reproducible scripts; and (iii) an evaluation of communication and computation against the other lattice-based proposals and existing systems.

²Composing “Th” for **Th**reshold and the Turkish word *imza*, meaning “signature”, Thimza: Threshold Signature.

2. Specification

Organization. The specification document will be modular. The first part fixes the arithmetic, starting with ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^\varphi + 1)$ with φ a power of two and q a prime supporting a number theoretic transform, the discrete Gaussian samplers, the rounding operators and the serialization rules. The second part specifies the four supporting functions built from SHAKE256. The third part specifies the secret sharing over $\mathcal{R}_q$ and the masking mechanism, both of which are self-contained. The fourth part specifies the core modules KeyGen, Sign₁, Sign₂, Sign₃, Combine and Verify. The fifth part gives parameter sets and the security estimation. Here, we go through the scheme from a general and high-point of view.

Key Generation. The dealer expands a short seed as $\text{seed}_\mathbf{A}$ into the public matrix $\mathbf{A}$ over the ring $\mathcal{R}_q$, so that the verification key stores 2λ bits instead of a full matrix. It samples the short secret $\mathbf{s}$ and the short error $\mathbf{e}$ from a discrete Gaussian of width σ_t , and publishes the verification key $\mathbf{t}$, which is $\mathbf{A}\mathbf{s} + \mathbf{e}$ with its ν_t least significant bits removed by the rounding operator $\lfloor \cdot \rfloor_{\nu_t}$. Once the key is fixed, the dealer shares the secret with Shamir's scheme as it draws uniform ring vectors $\mathbf{p}_1, \dots, \mathbf{p}_{t-1}$ as the coefficients of a polynomial whose constant term is $\mathbf{s}$, and gives party i the value of that polynomial at the public evaluation point α_i . Finally it draws one λ -bit seed $\text{sd}_{\{i,j\}}$ for every unordered pair of parties and hands each party its own share, the seeds of the pairs it belongs to, and the verification key. Any t shares reconstruct $\mathbf{s}$ through the Lagrange coefficients $\lambda_{T,i}$ of the signer set T , and any $t-1$ of them are uniform and do not carry information about it.

Signing. A signer first draws randomness $\mathbf{r}_i$ and its noise $\mathbf{e}'_i$ with the flooding width σ_w , forms the commitment $\mathbf{w}_i = \mathbf{A}\mathbf{r}_i + \mathbf{e}'_i$, truncates it to $\bar{\mathbf{w}}_i$ by dropping ν_c least significant bits, and broadcasts only the 32-byte hash cmt_i of it. The randomness and the truncated commitment form the signer's *token* for this session. This round mentions neither the signer set T nor any session identifier, and all of its arithmetic can be prepared while the machine is idle. The truncated commitment is opened in the next round. Once the openings arrive and the signer set is known, the signer checks every opening against its hash, hashes the whole transcript into the digest vw , rebuilds the aggregate $\mathbf{W}$ from the truncated openings, rounds it to $\mathbf{w}$ by dropping ν_w bits, and derives the Fiat–Shamir challenge c from $\mathbf{w}$ and the message. It then forms its mask $\mathbf{m}_i$ by evaluating the pseudorandom function on vw under each of the $t-1$ seeds it shares with the other signers, with the sign $\varepsilon_{ij} = +1$ when $i < j$ and -1 otherwise, so that the masks of the signer set add up to zero. Its partial signature is $\mathbf{z}_i = c\lambda_{T,i}\mathbf{s}_i + \mathbf{r}_i + \mathbf{m}_i$, after which the token is erased.

Combination and verification. The combiner, who may be any party and is not trusted, adds the partial signatures. The masks cancel and the sum is $c\mathbf{s} + \sum_{j \in T} \mathbf{r}_j$, which is exactly a Raccoon response under the full key. It then computes the hint $\mathbf{h}$ that repairs the two roundings, namely the difference between the aggregate commitment $\mathbf{w}$ that the signers agreed on and the value a verifier reconstructs from $\mathbf{z}$ and the rounded key $\mathbf{t}$. The signature is the triple $(c, \mathbf{z}, \mathbf{h})$. Verification recomputes that reconstruction, adds the hint, checks that hashing the result reproduces the challenge c , and checks that the response and the scaled hint are short with respect to the bound B .

Pairwise seeds, masks, views and tokens roles. Without its mask, a partial signature $c\lambda_{T,i}s_i + \mathbf{r}_i$ would carry information about the share s_i , while the mask makes it uniform, so only the sum, a Raccoon response, leaves the signer set. The pairwise seeds, fixed at key generation, let the masks cancel for any signer set chosen at signing time, and a seed held by two honest signers is unknown to the adversary, so its term hides both of their partial signatures. Since masks are evaluated on the view, different openings delivered to two honest signers stop their common term from cancelling, so the sum is uniformly random, fails verification and reveals nothing, and since every view contains the signer's own fresh commitment, no mask is reused and no registry is needed. Answering the last round twice with one token, under two different challenges, would expose the share, so each token is consumed once and erased.

System model. Key generation is performed once by a trusted dealer; a distributed key generation can replace it, and any protocol that outputs a Shamir sharing of a short secret together with pairwise seeds is a drop-in substitute. The threshold profile is (t, n) for any $1 \leq t \leq n \leq 1024$ with a dishonest majority. Thimza considers trust assumptions as (I) a trusted dealer that runs key generation once, erases (s, e) and delivers each share and its pairwise seeds over a private channel; (II) a static adversary that fixes at most $t - 1$ corrupted parties before key generation, i.e., before seeing any key material, as in the static model of [PKMMPS24; BKLMTT25], and thereafter observes and controls the whole network; (III) a broadcast medium that is neither authenticated nor reliable, so that any message may be dropped, reordered, duplicated or forged, including one that claims to come from an honest signer; (IV) honest parties that consume each token of the first round at most once and then erase it; (V) a random oracle for the three hash functions and a pseudorandom function for the masks; (VI) and the hardness of MLWE and self-target MSIS at the chosen parameters against an adversary making at most Q_s signing queries. Against such an active adversary, Thimza aims at security with selective abort. Safety does not depend on the network, so the adversary cannot obtain a valid signature on a message that no honest signer agreed to sign, extract key material, or make an honest signer reuse a token. Liveness does, since the adversary can make an execution abort, possibly for only some honest signers, ending with no signature or with one that the verifier rejects, so a signature needs a synchronous network and all t signers of T to follow the protocol.

Unauthenticated channels. Threshold Raccoon authenticates its views with a tag of 16 bytes per signer of T , which at $t = 1024$ adds 16 KiB per signer, about the whole transcript of Thimza. Unauthenticated messages suit signers behind an untrusted relay, such as hardware security modules or offline devices whose messages a coordinator carries, and validator committees on a public gossip network. Multicast over point-to-point links gives the same guarantees, and so do authenticated channels alone, since a corrupted signer can still send different openings to different honest signers. An authenticated and reliable broadcast gives all honest signers the same view, so they abort together, which yields security with unanimous abort, while identifiable abort needs partial signatures that can be verified one by one, as in Hermine [BCPEKNPT26].

Security. We will prove unforgeability against chosen message attacks in the quantum random oracle model (QROM), under static corruption of up to $t - 1$ parties, from MLWE and self-target MSIS, the same assumptions that underlie Raccoon [PKPR24] and its threshold variants [PKMMPS24; BKLMTT25]. The security notion is the standard game for threshold signatures,

strengthened in the network model as described above. The reduction is parameterized by the number of signing queries rather than by the number of random oracle queries, which is what allows the parameter sets to stay at the sizes we mention here. Security strengths will be estimated with the core-SVP methodology using the lattice estimator [APS15], and we will share the resulting estimates for every parameter set. We will also discuss strong unforgeability and the notions above TS-UF-0 in the threshold signature unforgeability (TS-UF) hierarchy [BCKMTZ22], and state which of them the scheme achieves. In addition to the proofs for Thimza in our preliminary draft [AA26], we will also provide machine-checked proofs for it in QROM. As an interactive theorem prover with support for expressing proofs in the computational models of cryptography, EasyCrypt has been used extensively for checking security mechanisms, so we will provide EasyCrypt proofs at: <https://github.com/navid-divan/thimza/tree/main/easycrypt-proofs>.

3. Open Source Implementation

Code structure. The core code will consist of an arithmetic backend over $\mathcal{R}_q$ with a number theoretic transform, samplers for the uniform and discrete Gaussian distributions, the SHAKE256-based supporting functions, the secret sharing and masking modules, the six core modules of Section 2, and a serializer. We plan a reference implementation in Python with a dependency on a SHAKE256 implementation with standard toolchains, including build and benchmark scripts. The randomized parts are separated from the deterministic ones so that every module can be driven from fixed random bytes.

Code progress and availability. A prototype that implements all six core modules (KeyGen, Sign₁, Sign₂, Sign₃, Combine and Verify), checks its own output against an independent Raccoon verifier and reproduces the benchmarks is complete at: <https://github.com/navid-divan/thimza>.

Networking model. In our protocol, parties only communicate via an unreliable channel, and each message is a multicast to the other signers, which may be realized as a broadcast where the communication layer supports one. Our wrapper will implement it both over point-to-point links, where each signer sends to every other, and through an untrusted coordinator that collects and redistributes the messages, so that a signer uploads each message once instead of $t - 1$ times. We expect this deployment to be the most common. Because unforgeability does not require authenticated channels, the message transport does not rely on key material beyond the pairwise seeds, although a deployment that cares about liveness will normally still authenticate its links.

Testing. Testing will cover three levels. Known answer tests fix the message, the signer set and all random bytes, so that a conforming implementation reproduces every intermediate value. Property tests run the full protocol for many threshold profiles and check that the signature is accepted by an independently written verifier. Adversarial tests inject the behaviors the design must tolerate, such as inconsistent transcripts delivered to different honest signers, forged messages that claim to come from an honest signer, replayed rounds, and truncated or reordered broadcasts. The expected outcome is that an execution may fail but that no partial signature ever leaks key material and that a token is never consumed twice.

Scheme	t	comm.	$ \sigma $	$ \text{vk} $
Thimza	1024	16.3	12.8	3.8
TRaccoon [PKMMPS24]	1024	55.8	12.8	3.8
EKT [EKT24]	1024	271.9	13.8	4.2
Ringtail [BKLMTT25]	1024	614.5	13.9	4.5
TalonG [†] [GTJLLWY26]	1024	26.9	17.7	2.0
Hermine [BCPEKNPT26]	64	72.5	12.5	2.8

[†]Published values, as our TalonG code uses rounding instead of modulus switching.

Table 1: Sizes in kibibytes at the 128-bit PQ security level, measured with one serializer at the largest t that each scheme supports.

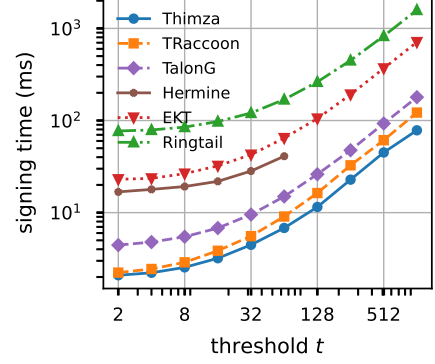


Figure 1: Measured single-core signing time per signer as a function of the threshold t , for Thimza, TRaccoon, Hermine, TalonG, EKT and Ringtail (log-log axes).

4. Experimental Evaluations

Performance. Table 1 shows sizes at the 128-bit PQ security level. Its communication column counts each message of a signer once, as for a broadcast, so a signer receives $t - 1$ times this amount and, over point-to-point links without a coordinator, also sends each message $t - 1$ times. Communication per signer does not depend on the threshold, since no message of a signer carries data for each individual signer, such as an authentication tag. Computation per signer grows linearly in t , because a signer evaluates one pseudorandom function per other signer; in our prototype, which shares one arithmetic backend across all measured schemes, this is 2.2 ms at $t = 4$ and 78 ms at $t = 1024$ on a single core, against 122 ms for TRaccoon, 179 ms for TalonG, 710 ms for EKT and 1584 ms for Ringtail at the same threshold, and 41 ms for Hermine at $t = 64$, where our own figure is 6.8 ms. Verification does not depend on t and is that of Raccoon.

In terms of time, Figure 1 shows that Thimza’s per-signer computation time is the lowest curve at every value of t , as it remains $1.56\times$ faster than TRaccoon and up to $20\times$ faster than Ringtail at $t = 1024$, and the gap does not close as the committee grows.

Platform. The baseline platform of a single computer with 16 cores and 64 GB of RAM is comfortable for this scheme. Memory is not a constraint, since a signer holds one share, $n - 1$ seeds and one token, which is under 50 KiB at the largest supported committee. The work of a signer is parallel across the $t - 1$ evaluations of the pseudorandom function, so cores translate directly into lower wall-clock time. The interesting challenge is not the platform but the network because the protocol has three rounds, so in a wide area deployment the round trips dominate at small committees while computation and bandwidth dominate at large ones. Therefore, we will provide end-to-end latency under a modeled network alongside local timings.

5. Licensing, Patent Claims, and Funding

The core code will be released under the Apache License 2.0, and we are not aware of any known patent applicable to this submission nor plan to submit for one.

6. Comparison with Other Submissions

As encouraged by NIST MPTC call [NIST-IR8214C] to facilitate assessment, here, we bring some notes on comparing with other submissions. Three other lattice-based threshold signature schemes were submitted to NIST MPTC call: Tanuki, Hermine, and Quorus. We already saw Thimza's efficiency over Hermine in Section 4 but we bring up further technical notes here. It is worth noting that we observe that our Thimza, along with the other three schemes, can be/seem to be complementary rather than competing alternatives.

Tanuki [BEKKK+26] merges the two-round constructions of Espitau, Katsumata and Takemure and of Ringtail. It shares our underlying signature and our threshold profile, and its first round can be preprocessed, which gives it a single online round and therefore the lowest latency wherever preprocessing can be amortized. Its preprocessing message is a wide matrix, which is why its total communication seems to be an order of magnitude above ours, and its proof goes through an algebraic one-more assumption. Thimza and Tanuki are the two ends of a trade between latency and bandwidth at the same threshold profile, and since both produce Raccoon signatures a deployment can switch between them without modifying its verifiers.

Hermine [BCPEKNPT26] adds two properties that we do not have, which are identification of a misbehaving signer without extra interaction, and proactive refresh of the shares. It obtains them by replacing the masks with a secret sharing whose shares are themselves short, which makes each partial signature verifiable on its own. The price is a committee limited to 64 signers, key material that grows with the committee, and communication that our own measurements put at a factor 4.4 above ours. Where accountability matters more than scale and bandwidth, Hermine is the better fit; where the reverse holds, Thimza is a better fit.

Quorus [BCEPT26] focuses on compatibility with ML-DSA, rather than with Raccoon. That is a different goal, and it carries the difficulties of thresholdizing rejection sampling and hint computation, which push it towards smaller committees or higher round complexity. A deployment that must keep FIPS 204 verifiers will prefer it; one that can adopt a Raccoon verifier will find our sizes and speeds better at large committees.

Although our focus is on threshold signatures, our secret sharing and masking modules are self-contained and could be reused as gadgets by other teams.

References

- [AA26] Navid Abapour and Hamidreza Amini Khorasgani. *Thimza: Stateless Threshold Signatures at Optimal Bandwidth*. Preliminary draft. 2026. URL: <https://github.com/navid-divan/thimza/blob/main/docs/Draft.pdf>.
- [APS15] Martin R. Albrecht, Rachel Player, and Sam Scott. “On the Concrete Hardness of Learning with Errors”. In: *Journal of Mathematical Cryptology* 9.3 (2015), pp. 169–203. DOI: [10.1515/jmc-2015-0016](https://doi.org/10.1515/jmc-2015-0016). Also at ia.cr/2015/046.
- [BCEPT26] Alexander Bienstock, Leo de Castro, Daniel Escudero, Antigoni Polychroniadou, and Akira Takahashi. *Quorus: Scalable Threshold ML-DSA from MPC*. Preview Writeup, NIST Threshold Call. 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Quorus-PW01.pdf>.
- [BCKMTZ22] Mihir Bellare, Elizabeth C. Crites, Chelsea Komlo, Mary Maller, Stefano Tessaro, and Chenzhi Zhu. “Better than Advertised Security for Non-interactive Threshold Signatures”. In: *CRYPTO 2022, Part IV*. Vol. 13510. LNCS. Springer, 2022, pp. 517–550. DOI: [10.1007/978-3-031-15985-5_18](https://doi.org/10.1007/978-3-031-15985-5_18). Also at ia.cr/2022/833.
- [BCPEKNPT26] Giacomo Borin, Sofía Celi, Rafael del Pino, Thomas Espitau, Shuichi Katsumata, Guilhem Niot, Thomas Prest, and Kaoru Takemure. *Hermine: An Efficient Raccoon-Style Non-Interactive Threshold Signature with Advanced Properties*. Preview Writeup, NIST Threshold Call. 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Hermine-PW01.pdf>.
- [BEKKK+26] Cecilia Boschini, Thomas Espitau, Aaron Kaiser, Shuichi Katsumata, Darya Kaviani, Russell W. F. Lai, Giulio Malavolta, Thomas Prest, Peter Schwabe, Akira Takahashi, Kaoru Takemure, and Mehdi Tibouchi. *Tanuki: Two-Round Threshold Signatures from Lattices*. Preview Writeup, NIST Threshold Call. 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Tanuki-PW01.pdf>.
- [BKLMTT25] Cecilia Boschini, Darya Kaviani, Russell W. F. Lai, Giulio Malavolta, Akira Takahashi, and Mehdi Tibouchi. “Ringtail: Practical Two-Round Threshold Signatures from Learning with Errors”. In: *2025 IEEE S&P*. 2025, pp. 149–164. DOI: [10.1109/SP61157.2025.00070](https://doi.org/10.1109/SP61157.2025.00070). Also at ia.cr/2024/1113.
- [EKT24] Thomas Espitau, Shuichi Katsumata, and Kaoru Takemure. “Two-Round Threshold Signature from Algebraic One-More Learning with Errors”. In: *CRYPTO 2024, Part VII*. Vol. 14926. LNCS. Springer, 2024, pp. 387–424. DOI: [10.1007/978-3-031-68394-7_13](https://doi.org/10.1007/978-3-031-68394-7_13). Also at ia.cr/2024/496.
- [GTJLLWY26] Liming Gao, Guofeng Tang, Dingding Jia, Yijian Liu, Bingqian Liu, Xianhui Lu, Kunpeng Wang, and Yongjian Yin. “TalonG: Bandwidth-Efficient Two-Round Threshold Signatures from Lattices”. In: *EUROCRYPT 2026*. LNCS. Springer, 2026. DOI: [10.1007/978-3-032-25291-3_4](https://doi.org/10.1007/978-3-032-25291-3_4). Also at ia.cr/2026/303.
- [Lyu12] Vadim Lyubashevsky. “Lattice Signatures without Trapdoors”. In: *EUROCRYPT 2012*. Vol. 7237. LNCS. Springer, 2012, pp. 738–755. DOI: [10.1007/978-3-642-29011-4_43](https://doi.org/10.1007/978-3-642-29011-4_43). Also at ia.cr/2011/537.
- [Nat15] National Institute of Standards and Technology. *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. Federal Information Processing Standards Publication FIPS 202. 2015. DOI: [10.6028/NIST.FIPS.202](https://doi.org/10.6028/NIST.FIPS.202).
- [PKMMPS24] Rafael del Pino, Shuichi Katsumata, Mary Maller, Fabrice Mouhartem, Thomas Prest, and Markku Juhani O. Saarinen. “Threshold Raccoon: Practical Threshold Signatures from Standard Lattice Assumptions”. In: *EUROCRYPT 2024, Part II*. Vol. 14652. LNCS. Springer, 2024, pp. 219–248. DOI: [10.1007/978-3-031-58723-8_8](https://doi.org/10.1007/978-3-031-58723-8_8). Also at ia.cr/2024/184.
- [PKPR24] Rafael del Pino, Shuichi Katsumata, Thomas Prest, and Mélissa Rossi. “Raccoon: A Masking-Friendly Signature Proven in the Probing Model”. In: *CRYPTO 2024, Part I*. Vol. 14920. LNCS. Springer, 2024, pp. 409–444. DOI: [10.1007/978-3-031-68376-3_13](https://doi.org/10.1007/978-3-031-68376-3_13). Also at ia.cr/2024/1291.
- [NIST-IR8214C] Luís T. A. N. Brandão and René Peralta. *NIST First Call for Multi-Party Threshold Schemes*. (National Institute of Standards and Technology) NIST Internal Report (NISTIR) 8214C. 2026. DOI: [10.6028/NIST.IR.8214C](https://doi.org/10.6028/NIST.IR.8214C).