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

Title: LEAST

Subtitle: Linear Equivalence Action Threshold Signature

Version: 0.1 (2026-01-20)[1]

Team name: LEAST -project

Team members: Baldi Marco, Battagliola Michele, Borin Giacomo, Di Crescenzo Giovanni, El Mechri Rahmi, Meneghetti Alessio, Persichetti Edoardo, Santini Paolo, Floyd Zweydinger

Abstract: This document preview the LEAST submission to the NIST Call for Multi-Party Threshold Schemes. LEAST relies on the framework of cryptographic group actions, a fundamental mathematical tool with a long history of use in cryptography. In particular, LEAST focuses on the Linear Code Equivalence group action, which is at the basis of the post-quantum LESS signature scheme. LEAST exploits a multiplicative secret sharings for non-commutative group actions to distribute in a Round Robin fashion the signing primitive of the LESS signature scheme. The LEAST submission also contains a distributed key generation procedure.

Proposed crypto-systems: (I) LEAST (Category S1); LEAST Distributed Key Generation (Category S4).

Keywords: Threshold Signature; Secret Sharing; Linear Code Equivalence; LESS

[1] 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: Marco Baldi [i1,a1], Michele Battagliola [i2,a1], Giacomo Borin [i3,a2,a6], Giovanni Di Crescenzo [i4,a7], Rahmi El Mechri [i5,a1,a3], Alessio Meneghetti [i6,a4], Edoardo Persichetti [i7,a5], Paolo Santini [i8,a1], Floyd Zweydinger [i9,a8]

Open Researcher and Contributor Identifiers (ORCID):

i1 (0000-0002-8754-5526); i2 (0000-0002-8269-2148); i3 (0009-0001-7311-3802); i4 (0000-0002-5138-1144); i5 (0009-0007-3739-9106); i6 (0000-0002-5159-7252); i7 (0000-0002-1895-377X); i8 (0000-0003-0631-3668); i9 (0009-0006-7610-9143)

Affiliations:

[a1] Department of Information Engineering, Marche Polytechnic University, Ancona, Italy

[a2] Department of Mathematics, University of Zurich, Zurich, Switzlerand

[a3] IMT School for Advanced Studies Lucca, Lucca, Italy

[a4] Department of Mathematics, University of Bari Aldo Moro, Bari, Italy

[a5] Department of Mathematics, Florida Atlantic University, Boca Raton, USA

[a6] Foundations of Cryptography Group, IBM Research, Zurich, Switzerland

[a7] Peraton Labs, USA

[a8] Technology Innovation Institute, UAE

Main contacts:

- **Team mailing list:** info@least-project.org
- **Primary technical contact person:** Michele Battagliola, battagliola.michele@proton.me
- **Secondary contact person 1:** Giacomo Borin, crypto@gbor.in
- **Secondary contact person 2:** Edoardo Persichetti, epersichetti@fau.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

The LESS signature scheme [BBBBC+24] is a promising post-quantum digital signature scheme relying on the hardness of the *Linear Code Equivalence* problem. The scheme fits the *Cryptographic group actions* framework [ADMP20] and presents similar characteristics to isogeny-based signature schemes [DG19; BKV19]. This because both for the latter schemes and LESS the core subroutine can be abstracted as a group action $\star : G \times X \to X$ of a group G onto a set X. In both cases, the hardness assumption is the Group Action Discret Logarithm Problem **gaDLOG** problem, which states that given elements $x, y \in X$ such that $y = g \star x$ it is hard to recover g. In LESS the public key is a **gaDLOG** instance x, y and the secret key is the witness g. The core difference between isogeny-based and code-based group actions stems from the structure of the group G. In the former the group is commutative, allowing more complex constructions like public key encryption, but also sub-exponential quantum attacks to **gaDLOG** [Kup05; Reg04; Kup13; BS20; Pei20]. In code-based group actions, instead, the group is not commutative. However, it is still possible to instantiate (threshold) digital signatures with them. In fact, the LEAST submission leverages post-quantum *code-based group actions* [BMPS20] to instantiate the following threshold primitives:

- A Threshold Signature Scheme (category S1) with Dishonest majority.
- A Distributed Key Generation (DKG) mechanism (category S4), compatible with the signature.

Both protocols are actively secure against static adversaries.

We argue that LEAST is relevant to the NIST call for the following reasons:

1. **Solid Security Foundations:** Linear Equivalence is a traditional problem from coding theory, which is well-known and has been studied for decades. Weak instances have been extensively studied and identified in literature, and they are easy to avoid. The currently best known solvers boil down to codeword searching and can therefore rely on the established security track of the Syndrome Decoding Problem (SDP).
2. **Quantum-safety, Diversity:** the schemes are plausibly quantum-safe, based on different assumptions compared to lattice schemes. Also, the non-commutative structure of the group thwarts sub-exponential quantum attacks, allowing for efficient scalability to higher security levels.
3. **Compatibility with LESS :** with minimal modifications to the LESS scheme [BBBBC+24] (currently a candidate for standardization) parameters, the LEAST threshold signing procedure is compatible with the LESS verification procedure.
4. **Compactness:** as in the case of centralized LESS , our signatures are small when compared to other ZK-based signatures.

We also want to highlight the limitations of LEAST , some of them being shared with LESS :

1. Inherently **sequential structure** of the threshold primitives (aka *Round Robin* communication structure), leading to a high number of rounds and increased protocol latency.

2. **Computational bottleneck:** as in the case of LESS , our protocol is relatively slow due to the cost of the Gaussian elimination algorithm and the computation of the canonical forms. In conjunction with the aforementioned round robin structure, this may limit its usability for applications where speed is the paramount priority.
3. **Large public keys:** LESS features relatively large public keys, in the order of at least a few kilobytes, and this also holds for our protocol.
4. **Limited number of parties:** the number of shares that each party holds is not constant and becomes exponentially large. For this reason, LEAST is suitable when N, T are close (e.g. the full threshold case with $N = T$) or low enough.

2. Specification

We first provide an high-level group-agnostic analysis of the threshold schemes (S1, S4) and their security properties. The security of these schemes will consider active static adversaries and rely on assumptions from the group action literature. Then we describe the group action used, and specific code-based optimizations [CPS25; PS23] that can be integrated within the threshold signing procedure.

Let T be the threshold and N the number of parties. A LESS key pair is composed by a secret group element g and a public pair of set elements x, y with $y = g \star x$.

2.1. Secret Sharing

As for LESS , LEAST secret keys belongs to a non-commutative group and thus many of the secret sharing schemes in the literature are not directly applicable. Instead, we consider multiplicative secret sharings [Des93], where, informally speaking, the secret can be written as a sequential product of shares – more precisely group elements – of qualified parties (thus, working well with secrets and shares in non-commutative groups).

Case $T = N$. In this case, the secret key g as the *ordered* product of N group elements $g_1 \cdots g_N$.

Case $2 \leq T < N$. To address this case, we consider the classical, state-of-the-art, Replicated Secret Sharing scheme from [LMC16], as well as previous and recent Recursive Secret Sharing schemes [DDB95; BBDMP25], as we now briefly describe.

- **Replicated Secret Sharing:** this technique splits a secret group value g into multiple group values g_i using the classical replicated secret sharing. The main problem is that the number of shares grows exponentially and becomes quickly unusable as T or N increases.
- **Recursive Secret Sharing:** The main idea is to split the set of parties into two sides, write the secret as the product of two random group values, and recursively share each of these two values, with all possible (and likely smaller) threshold values, over each side.
  - To do a 1-out-of-N sharing the dealer just distributes the secret group value g to all of the n parties.

- To achieve an N-out-of-N sharing, sharing and recovery proceed as explained in the above case $T = N$.
- Otherwise, the user set is split in two sides of approximately the same size. For any ℓ such that $\max(0, T - \lceil \frac{N}{2} \rceil) \leq \ell \leq \min(T, \lfloor \frac{N}{2} \rfloor)$, split g as $g_2 \cdot g_1$, then recursively perform a sharing with threshold ℓ of g_1 on the first set and a sharing with threshold $t - \ell$ of g_2 on the second set. To allow each user to correctly combine their secret shares, the dealer also labels each shared group value with a list containing all the recursive steps performed to arrive at the sharing.

Relying on Recursive Secret Sharing allow for concretely reducing the number of shares needed, even though the dependence on T and N is still super-polynomial, and reducing the number of rounds of interaction from $\binom{N}{T}$ to N.

We are currently working on several variants of the above recursive secret sharing approach, focusing on improving the scheme's performance.

2.2. (Distributed) Key Generation

Given as input any pair of LESS secret and public key $g, (x, g \star x)$, a trusted party can effectively share it using the aforementioned schemes, also providing parties with public verification shares.

For $T = N$, as explained in [BBMP24], a distributed key generation can be obtained in the following way. First, the users agree on an order of execution and a set origin element $x_0 \in X$. Then, in order, each user samples its own secret share $g_i \in G$ and publishes $x_i = g_i \star x_{i-1}$, which also serves as public verification share for the i-th user. Each user also needs to publish a non-interactive proof of knowledge for g_i, to be verified by other users. These proofs are as expensive as one centralized LESS signature. This distributed procedure can also be applied easily to Replicated Secret Sharing, as shown in [BBMP24].

The centralized version of the key generation for Recursive Secret Sharing is illustrated in [BBDMP25] and follows the same principle of the one based on replicated secret sharing: the dealer performs the secret sharing of g and publishes all the intermediate set values as verification shares. The distributed key generation for the Recursive version is currently under analysis.

2.3. Threshold Signature Scheme

As a base case, we show how distributed signing works for $T = N$ users. Since we are using a multiplicative secret sharing, g is shared among the N users as $g = g_1 \cdot \cdots \cdot g_N$. Also, users know public verification shares $x_i = g_i \star x_{i-1}$, with $x_0 = x$ and $x + N = y$. The signature algorithm is divided in two phases, both having a Round Robin structure:

1. The first phase is message-less and can be preprocessed. The first player computes a random group action $x_1 = g_{r_1} \star x$ and sends it to the next player. Then, the other players continue following the Round Robin and computing each random group action g_{r_i} on the set element received from the previous player. The final set element $\tilde{x}$ is the commitment.
2. A random binary challenge is computed using some secure randomness and $\tilde{x}$.

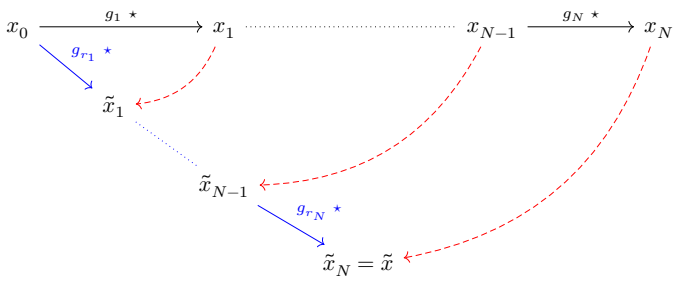

Figure 1: Schematic representation of the protocol idea, assuming $T = N$ and $g = g_1 \cdot \cdots \cdot g_N$. The elements $x_1, \ldots, x_N$ are the public verification shares. In blue are the ephemeral group elements revealed on challenge 0, while in red the map reconstructed for challenge 1.

3. If the challenge is 0, all the players reveal g_{r_i} so that it is possible to compute the action linking x to $\tilde{x}$, otherwise the players perform a second Round Robin to compute the action from y to $\tilde{x}$ (which is obtained from reconstructing g and all the g_{r_i}). During the Round Robin stage, parties can always verify the correctness of other parties computation using public verification shares, effectively identifying misbehaving parties.

The above protocol is repeated in parallel multiple times to achieve the desired security level (i.e. λ repetitions ensure a security level of 2^λ as long as all the parameters are suitably chosen).

2.4. Group Action Specification and Optimizations

Let p be a prime and $\mathbb{F}_p$ be the field with p elements. In our scheme, the set X will be the set of $[n, k]$ codes over $\mathbb{F}_p$, while G will be the group M_n of $n \times n$ monomial matrices with coefficients in $\mathbb{F}_p$. The action considered is the following

$$\begin{aligned} \star \colon M_n \times X &\longrightarrow X \\ (\mathbf{Q}, \mathbf{G}) &\longmapsto \mathsf{SF}(\mathbf{GQ}), \end{aligned}$$

where $\mathbf{G}$ represents the generator matrix of the code and SF represents the reduction to standard form. More details on the group actions can be found in [BBBBC+24]. To achieve more compact signatures we rely on the usage of canonical forms, as explained in [CPS25]. The canonical form is computed only on the final commitment $\tilde{x}$ and not on all the intermediate values.

As in the case of LESS , we can use the multi-key optimization to decrease the soundness error of the protocol, that allows reducing the number of parallel executions.

3. Open-Source Implementation

All source code will be released under open-source software licenses compatible with those of possible bundled dependencies. We plan to present a high-level Rust implementation of the LEAST threshold protocol and the associated networking model, leveraging efficient Rust-based group action routines provided by LESS . Since the primary computational bottlenecks lie in Gaussian elimination and canonical form computation, we adopt the same approach as LESS by introducing optimized implementations using AVX2, AVX512, and NEON instruction sets.

4. Experimental Performance Evaluation

A public key comprises s monomial matrices $\mathbf{G}_0, ..., \mathbf{G}_{s-1}$, so that $\mathbf{G}_i = \mathsf{SF}(\mathbf{G}_0\mathbf{Q}_i)$ for some (shared) monomial matrix $\mathbf{Q}_i$. The total number of repetitions needed to achieve λ bits of security is $\tau = \lceil \frac{\lambda}{\log_2(s)} \rceil$. The signature size is computed as $\ell_{\mathsf{salt}} + \tau\ell_{\mathsf{monomial}} + \ell_{\mathsf{digest}} + 1$, where:

- ℓ_{salt} is the size of the salt (resp. 32, 48 and 64 bytes);
- ℓ_{monomial} is the size of a monomial matrix ($\frac{n}{8}$ bytes);
- ℓ_{digest} is the size of the digest (resp. 32, 48 and 64 bytes).

Table 1 shows the expected public key and signature sizes for various parameter choices. Table 2 shows the total number of shares each user has and the total number of public intermediate generator matrices used for verifiability (considering only Recursive Secret Sharing). Notice that the table refers to a single couple in the public key (i.e. $s = 1$).

4.1. Timing

While concrete benchmarks are not available yet, due to the lack of a full implementation, we provide a very rough estimate of timing for LEAST . Let n_{shares} be the number of shares required to reconstruct the secret key, that depends on the secret sharing scheme used.

- During the Round Robin stage required for the computation of $\tilde{x}$, each user performs the following steps[2] one matrix multiplication by a monomial matrix and one Gaussian elimination;
- During the computation of the response, each user performs either the multiplication of n_{round} monomial matrices (challenge 0) or one multiplication of two monomial matrices and one correctness check of the received group element. This check is a simplified LESS verification (challenge 1).

An upper bound for the computational effort of each user is thus: 2 Gaussian eliminations, one monomial matrix generation and 3 products by a monomial matrix. This cost is dominated by the Gaussian elimination (roughly 0.50 MCycles for NIST Category 1).

At the end, a single canonical form computation (roughly 0.10 MCycles for NIST Category 1) and a single hash (KECCAK) of $\tilde{x}$ (roughly 0.10 MCycles for NIST Category 1) are required.

Overall, using as example LEAST -252-2 with $T = 2$ and recursive secret sharing, we expect around $128 \cdot 2 \cdot 0.5 + 128 \cdot 0.2 = 153.6$ MCycles, without considering any additional delay due to the communication between the parties.

The estimation of clock cycles is based on a Ryzen 5 7600X processor, which is the same platform used in LESS .

[2] for a single parallel repetition

Table 1: Performance and Parameters for LEAST Signature Schemes.

NIST Cat.	Parameter Set	Prot. Params n	k	q	Keys s	Rep. τ	pk size (B)	Signature size (B)
1	LEAST -252-2	252	126	127	2	128	13940	4160
	LEAST -252-4				4	64	41788	2112
	LEAST -252-8				8	43	97484	1440
3	LEAST -400-2	400	200	127	2	192	35074	9696
	LEAST -400-4				4	96	105174	4896
5	LEAST -548-2	548	274	127	2	256	65793	17792
	LEAST -548-4				4	128	197315	8960

Table 2: Number and size of shares, both per users and in total. The size refers to the case $s = 2$, i.e., the public key is a single couple $(\mathbf{G}_0, \mathbf{G}_1)$.

NIST Cat.	N	T	Per User # Shares	Per User Size (B)	Total # Shares	Total Size (B)
1	3	2	2	64	4	55696
	5	3	4	128	11	153164
	8	5	6	192	30	417720
3	3	2	2	100	4	140200
	5	3	4	200	11	385550
	8	5	6	300	30	417720
5	3	2	2	138	4	263044
	5	3	4	276	11	1051500
	8	5	6	414	30	1972830

5. Licensing, Patent Claims, and Funding

We are not aware of any patent claims covering the contents of the submission. Marco Baldi, Michele Battagliola, Rahmi El Mechri and Paolo Santini are supported by the Italian Ministry of University and Research (MUR) under the PRIN 2022 program with projects "Mathematical Primitives for Post Quantum Digital Signatures" (CUP I53D23006580001) and "Post quantum Identification and eNcryption primiTives: dEsign and Realization (POINTER)" (CUP I53D23003670006), by MUR under the Italian Fund for Applied Science (FISA 2022), project "Quantum-safe cryptographic tools for the protection of national data and information technology assets" (QSAFEIT) - No. FISA 2022-00618 (CUP I33C24000520001).

Giacomo Borin is supported by *CryptonIs, SNSF Consolidator Grant 213766*, (https://data.snf.ch/grants/grant/213766).

Alessio Meneghetti is supported by the project PRIN 2022SC, title "Algebraic Methods in Cryptanalysis", Grant Ref. 2022RFAZCJ, CUP H53C24000830006.

References

[ADMP20] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. "Cryptographic Group Actions and Applications". In: *Advances in Cryptology – ASIACRYPT 2020, Part II*. Ed. by Shiho Moriai and Huaxiong Wang. Vol. 12492. Lecture Notes in Computer Science. Daejeon, South Korea: Springer, Cham, Switzerland, December 2020, pp. 411–439. DOI: 10.1007/978-3-030-64834-3_14. Also at ia.cr/2020/1188.

[BBBBC+24] Marco Baldi, Alessandro Barenghi, Luke Beckwith, Jean-François Biasse, Tung Chou, Andre Esser, Kris Gaj, Patrick Karl, Kamyar Mohajerani, Gerardo Pelosi, Edoardo Persichetti, Markku-Juhani O. Saarinen, Paolo Santini, Robert Wallace, and Floyd Zweydinger. *LESS — Linear Equivalence Signature Scheme*. Tech. rep. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures. National Institute of Standards and Technology, 2024.

[BBDMP25] Michele Battagliola, Giacomo Borin, Giovanni Di Crescenzo, Alessio Meneghetti, and Edoardo Persichetti. "Enhancing Threshold Group Action Signature Schemes: Adaptive Security and Scalability Improvements". In: *Post-Quantum Cryptography - 16th International Workshop, PQCrypto 2025, Part I*. Ed. by Ruben Niederhagen and Markku-Juhani O. Saarinen. Taipei, Taiwan: Springer, Cham, Switzerland, April 2025, pp. 129–161. DOI: 10.1007/978-3-031-86599-2_5. Also at ia.cr/2025/085.

[BBMP24] Michele Battagliola, Giacomo Borin, Alessio Meneghetti, and Edoardo Persichetti. "Cutting the GRASS: Threshold GRoup Action Signature Schemes". In: *Topics in Cryptology – CT-RSA 2024*. Ed. by Elisabeth Oswald. Vol. 14643. Lecture Notes in Computer Science. San Francisco, CA, USA: Springer, Cham, Switzerland, May 2024, pp. 460–489. DOI: 10.1007/978-3-031-58868-6_18. Also at ia.cr/2023/859.

[BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. "CSI-FiSh: Efficient Isogeny Based Signatures Through Class Group Computations". In: *Advances in Cryptology – ASIACRYPT 2019, Part I*. Ed. by Steven D. Galbraith and Shiho Moriai. Vol. 11921. Lecture Notes in Computer Science. Kobe, Japan: Springer, Cham, Switzerland, December 2019, pp. 227–247. DOI: 10.1007/978-3-030-34578-5_9. Also at ia.cr/2019/498.

[BMPS20] Jean-François Biasse, Giacomo Micheli, Edoardo Persichetti, and Paolo Santini. "LESS is More: Code-Based Signatures Without Syndromes". In: *AFRICACRYPT 20: 12th International Conference on Cryptology in Africa*. Ed. by Abderrahmane Nitaj and Amr M. Youssef. Vol. 12174. Lecture Notes in Computer Science. Cairo, Egypt: Springer, Cham, Switzerland, July 2020, pp. 45–65. DOI: 10.1007/978-3-030-51938-4_3. Also at ia.cr/2020/594.

[BS20] Xavier Bonnetain and André Schrottenloher. "Quantum Security Analysis of CSIDH". In: *Advances in Cryptology – EUROCRYPT 2020, Part II*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12106. Lecture Notes in Computer Science. Zagreb, Croatia: Springer,

Cham, Switzerland, May 2020, pp. 493–522. DOI: 10.1007/978-3-030-45724-2_17. Also at ia.cr/2018/537.

[CPS25] Tung Chou, Edoardo Persichetti, and Paolo Santini. "On linear equivalence, canonical forms, and digital signatures". In: 93.7 (2025), pp. 2415–2457. DOI: 10.1007/s10623-025-01576-1. Also at ia.cr/2023/1533.

[DDB95] Yvo Desmedt, Giovanni Di Crescenzo, and Mike Burmester. "Multiplicative Non-abelian Sharing Schemes and their Application to Threshold Cryptography". In: *Advances in Cryptology – ASIACRYPT'94*. Ed. by Josef Pieprzyk and Reihaneh Safavi-Naini. Vol. 917. Lecture Notes in Computer Science. Wollongong, Australia: Springer Berlin Heidelberg, Germany, November 1995, pp. 21–32. DOI: 10.1007/BFb0000421.

[Des93] Yvo Desmedt. "Treshold Cryptosystems (invited talk)". In: *Advances in Cryptology – AUSCRYPT'92*. Ed. by Jennifer Seberry and Yuliang Zheng. Vol. 718. Lecture Notes in Computer Science. Gold Coast, Queensland, Australia: Springer Berlin Heidelberg, Germany, December 1993, pp. 3–14. DOI: 10.1007/3-540-57220-1_47.

[DG19] Luca De Feo and Steven D. Galbraith. "SeaSign: Compact Isogeny Signatures from Class Group Actions". In: *Advances in Cryptology – EUROCRYPT 2019, Part III*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. Lecture Notes in Computer Science. Darmstadt, Germany: Springer, Cham, Switzerland, May 2019, pp. 759–789. DOI: 10.1007/978-3-030-17659-4_26. Also at ia.cr/2018/824.

[Kup05] Greg Kuperberg. "A subexponential-time quantum algorithm for the dihedral hidden subgroup problem". In: *SIAM Journal on Computing* 35.1 (2005), pp. 170–188. DOI: https://doi.org/10.1137/S0097539703436345. Also at arXiv:quant-ph/0302112.

[Kup13] Greg Kuperberg. "Another Subexponential-time Quantum Algorithm for the Dihedral Hidden Subgroup Problem". In: *8th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2013)*. Ed. by Simone Severini and Fernando Brandao. Vol. 22. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013, pp. 20–34. DOI: 10.4230/LIPIcs.TQC.2013.20.

[LMC16] Jian Liu, Sihem Mesnager, and Lusheng Chen. "Secret Sharing Schemes with General Access Structures". In: *Information Security and Cryptology*. Ed. by Dongdai Lin, XiaoFeng Wang, and Moti Yung. Cham: Springer International Publishing, 2016, pp. 341–360. DOI: 10.1007/978-3-319-38898-4_20. Also at ia.cr/2015/1139.

[Pei20] Chris Peikert. "He Gives C-Sieves on the CSIDH". In: *Advances in Cryptology – EUROCRYPT 2020, Part II*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12106. Lecture Notes in Computer Science. Zagreb, Croatia: Springer, Cham, Switzerland, May 2020, pp. 463–492. DOI: 10.1007/978-3-030-45724-2_16. Also at ia.cr/2019/725.

[PS23] Edoardo Persichetti and Paolo Santini. "A New Formulation of the Linear Equivalence Problem and Shorter LESS Signatures". In: *Advances in Cryptology – ASI-*

ACRYPT 2023, Part VII. Ed. by Jian Guo and Ron Steinfeld. Vol. 14444. Lecture Notes in Computer Science. Guangzhou, China: Springer, Singapore, Singapore, December 2023, pp. 351–378. DOI: 10.1007/978-981-99-8739-9_12. Also at ia.cr/2023/847.

[Reg04] Oded Regev. *A Subexponential Time Algorithm for the Dihedral Hidden Subgroup Problem with Polynomial Space*. 2004. arXiv: quant-ph/0406151 [quant-ph].

[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.