

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

Title: DKLs

Subtitle: Threshold ECDSA from VOLE and Commitments

Version: Preview Writeup 0.1 (2026-08-07)¹

Team name: ABCDKLs: Associated Builders of and Contributors to the DKLs protocols

Team members: Jack Doerner, Yashvanth Kondi, Eysa Lee, Peter Rindal, Lawrence Roy, abhi shelat

Abstract: We propose the DKLs threshold ECDSA key-generation and signing schemes for NIST's MPTS call. Our protocols achieve statistical security against a dishonest majority in the hybrid model of vector oblivious linear evaluation (VOLE) and commitments. Our package will also include proposals for a VOLE protocol, an oblivious transfer (OT) extension protocol, a base OT protocol, and a commitment scheme, which are secure under the Diffie-Hellman assumption over the signing curve, in the random oracle model. Both key generation and signing require only three rounds of point-to-point communication.

Proposed crypto-systems:

- (I) DKLs Threshold ECDSA Key Generation (Categories N4.1)
- (II) DKLs Threshold ECDSA Signing (Categories N1.2)
- (III) DKLs VOLE (Categories S7)
- (IV) Softspoken OT Extension (Categories S7)
- (V) Endemic OT (Categories S7)
- (VI) Salted-Hash Commitments (Categories S7)

Keywords: Threshold Cryptography; Threshold ECDSA; VOLE; OT Extension; OT



Document License: [CC BY 4.0 International](https://creativecommons.org/licenses/by/4.0/)

¹Preliminary 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: Jack Doerner^{i1,a1}, Yashvanth Kondi^{i2,a2}, Eysa Lee^{i3,a3}, Peter Rindal^{i4,a4}, Lawrence Roy^{i5,a5}, abhi shelat^{i6,a6,a7}

Open Researcher and Contributor Identifiers (ORCID):

i1 (0009-0007-0409-2132); i2 (0009-0008-1848-9234); i3 (0009-0003-2860-2417); i4 (0000-0003-2402-7411); i5 (0009-0003-8436-0029); i6 (0009-0008-5362-469X)

Affiliations:

^{a1} Department of Computer Science, University of Virginia @ Charlottesville, VA, USA

^{a2} Silence Laboratories @ Singapore

^{a3} Department of Computer Science, Barnard College of Columbia University @ New York City, NY, USA

^{a4} Category Labs @ New York, NY, USA

^{a5} IBM Research Zurich @ Zurich, CH

^{a6} Khoury College of Computer Sciences, Northeastern University @ Boston, MA, USA

^{a7} Google @ Cambridge, MA, USA

Main contacts:

- **Team mailing list:** abc@dkls.info
- **Primary technical contact person:** Jack Doerner, j@ckdoerner.net
- **Secondary contact person 1:** Yashvanth Kondi, yash@ykondi.net
- **Secondary contact person 2:** Eysa Lee, eylee@barnard.edu
- **Secondary contact person 3:** abhi shelat, abhi@neu.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

We plan to submit a package to the NIST MPTC call describing the DKLs threshold ECDSA key generation and signing protocols [DKLs24], which fit within categories N4.1 and N1.2, and offer the following features:

- Three-round signing with *optional* preprocessing. The first round can always be preprocessed. The first two rounds can be preprocessed if the signing of nonce-dependent messages is permitted. This structure is optimal for any protocol that guarantees unbiased nonces and that is black-box in the signing curve. Although preprocessing is an option, it adds implementation complexity, and so we make no recommendation that it be used.
- Scalability: DKLs is designed for and has been verified to achieve concrete efficiency in a wide range of deployment scenarios. These include deployments to hundreds of parties scattered across the globe, and deployments to power-constrained devices such as embedded hardware and low-end mobile phones.
- Security: DKLs is proven to compositably realize an ideal functionality that computes ECDSA signatures exactly as specified by NIST. This security guarantee is easy to reason about and allows the signatures our protocol produces to be used anywhere the signatures of a trusted signer could be.
- Modularity and Conservative Design: DKLs is constructed and proven secure in a modular fashion, using building blocks that are standard to the literature of multiparty computation. In particular, oblivious transfer (OT) and OT extension, vector oblivious linear evaluation (VOLE), and commitments. The security of the top-level protocols and their building blocks follows exclusively from the security of NIST-standardized symmetric-key primitives, plus, in the case of OT, the Diffie-Hellman assumption on the signing curve.

This submission updates the original DKLs signing protocols [DKLs24] with improvements and corrections subsequently introduced by Asharov [Ash26], Segev [Seg26], and Doerner and Roy [DR26], and includes recommended instantiations for all necessary building-blocks. At least three independent open-source implementations of the DKLs protocols exist at present, in addition to a number of commercial implementations.

2. Specification

- **Organization:**

The document begins by specifying the interfaces for a number of cryptographic objects which are either used to construct or satisfied by the concrete cryptosystems enumerated below. These interfaces will be consistent between each object's ideal functionality (which defines its security) and its concrete instantiation. In addition to the interface, the code of each object's ideal functionality will be specified.

Next, the document specifies the following concrete cryptosystems, which are recommended for use as sub-protocols in the main key generation and signing protocols.

1. Instantiations for the pseudorandom function (PRF) and random oracle (RO) utilized by the other cryptosystems.
2. A simple commitment scheme which depends upon the random oracle [CDGLN18].
3. The Endemic OT protocol, which depends upon the random oracle [MR19].
4. The Softspoken oblivious transfer extension (OTE) protocol [Roy22], which depends internally upon oblivious transfer and the random oracle.
5. The DKLs vector oblivious linear evaluation (VOLE) protocol [DKLs24; DR26], which depends internally upon oblivious transfer (extension) and the random oracle. In addition to the specification of the protocol, we will include a table of parameterizations to match all standardized signing curves and several statistical security levels, and proof that the parameter choices achieve the desired level of security.

Finally, the document specifies the main protocols of interest, which make use of the components enumerated above.

6. The DKLs key generation protocol [DKLs24], which depends upon commitments and additionally uses Shamir sharing.
 7. The DKLs threshold ECDSA signing protocol [DKLs24], which depends upon commitments, VOLE, and key generation, and additionally uses Shamir sharing.
 8. Optional proactive state-refresh protocols.
- **System model:** The protocol is designed to tolerate a dishonest majority of signers. Thus, for n total system participants, the number of signers t can be chosen freely in $[2, n]$, and the number of corruptions tolerated is $t - 1$. In the notation of [NIST-IR8214C, Appendix C.3], DKLs is appropriate for (and has been tested against) profiles $n2$, $n3fD$, $nSfD$, $nMfD$, and $nLfD$. DKLs is also appropriate for the honest majority (h) and honest supermajority (H) settings. We expect signing to require more than one second (but less than one minute) in the “enormous” (E) setting, which may be tolerable in some applications.

The trusted setup assumption consists only of a global random oracle, which will be realized concretely using the same SHA2 hash function specified by ECDSA itself. The signing protocol uses only authenticated, asynchronous point-to-point channels. The key generation protocol additionally uses private authenticated asynchronous point-to-point channels.

The only mandatory long-term state that participants must hold is their shares of the secret key. Any pair of parties who can keep additional pairwise state can improve the efficiency of their interactions for the duration of time that they can keep such state. This pairwise state can be deleted without compromising security.

- **Security:** Security for the protocol is formulated in the Universal Composability (UC) model [Can01]. The protocol is proven to realize an ideal functionality that outputs ECDSA signatures exactly according to the NIST specification, in the presence of a malicious adversary that statically corrupts any number of participants that is less than the threshold. The functionality does not permit nonce bias, nor does it permit the message to depend upon the nonce. The functionality does permit the adversary to induce selective aborts, but

the key-generation component of the functionality guarantees that if any honest party does not receive a well-formed share of the secret key, then the honest parties abort unanimously.

The security proofs for each of the sub-protocols are independent of one another. The team will produce machine-checkable proofs of the major components, in addition to the extant paper proofs. When all of the sub-protocols are considered, security holds under the Diffie-Hellman assumption on the signing curve, regardless of whether ECDSA is a secure signature scheme. If it is assumed that ECDSA is existentially unforgeable, then the protocol additionally realizes a generic threshold signature functionality and satisfies game-based unforgeability [CDLLR25].

The protocol supports several optional features with impacts upon security:

- Proactive refresh is supported with the addition of a refresh protocol. The basic key generation and signing protocols need not be altered, and pairwise state (if kept) can be refreshed independently of key-shares [KMOS21].
- The signing protocol can be configured to support *presignatures* (i.e. the preprocessing of the first two rounds), in which case it realizes a functionality that allows the parties to supply the message to be signed *after* the nonce is revealed. We stress that in this case, unforgeability is *not* known to follow from the standard unforgeability of single-party ECDSA signatures. Nevertheless, this mode may be desirable since only one highly-efficient round of communication must occur after the message is supplied. We make no recommendation that presignatures be used.

3. Open-Source Implementation

1. **Code structure:** Our reference implementation will be written in the rust language, and will include modules for each of the interactive protocols specified in the document: OT, OTe, VOLE, key generation, and signing. Code will also be included to manage session identification, random oracle querying, and commitments. As a bundled dependency, we will supply a rust-based implementation of the necessary field arithmetic and elliptic curve operations, and our implementation will include a testing harness which allows for the easy benchmarking of our protocol on many machines over the internet.

In addition to the rust-based reference implementation of our protocol, we will include easycrypt-based machine checkable proofs of security for the signing and VOLE protocols, and a lean-based machine-checkable proof for the parameterization of the VOLE protocol.

2. **Code progress and availability:** We have a fully-functional in-house implementation, but it requires minor updates before public release. Additionally, several independently-developed open-source implementations are available via public git repositories. A preliminary version of our machine-checkable proof for the VOLE parameterization is already available via git.²

²See <https://codeberg.org/abhvious/scholia/src/branch/main/lhl>. Note that the lean code of this proof was developed with LLM assistance.

Parties	2	3	4	8	16	32	64	128	256
LAN	4.0	6.1	8.4	19.0	39.6	81.3	169.5	334.2	686.2
US WAN			105.3	108.2	113.7	125.8	183.0	348.6	749.8
Global WAN					494.9	503.6	532.0	623.1	1,326.4

Table 1: **Average Wall-clock Times in Milliseconds for Signing.**

3. **Implementation of the networking model:** Our signing protocol requires only authenticated point-to-point channels, and our key generation protocol requires only private and authenticated point-to-point channels. Consequently, our code generates the logical messages of our protocol and their addressees, and any secure point-to-point channel implementation can be used to deliver them. We provide a testing harness that performs message delivery using a standard rust library for TLS channels.
4. **Testing:** Our protocols are simulation-secure, and the simulators contain precise descriptions of the conditions under which malicious behavior should trigger an abort (or otherwise noticeable deviation the distribution of an honest protocol execution) in the ideal world. We can generate comprehensive³ test cases for malicious behavior by enumerating these conditions and explicitly triggering each one. The modularity of our protocol allows this process to be performed for each individual building block. Test vectors can similarly be specified for each individual building block.

4. Experimental Performance Evaluation

1. Performance:

In 2023, benchmarked our in-house implementation in three network settings, which we will refer to as *LAN*, *US WAN*, and *Global WAN*. Each party was allocated a single Google Cloud Platform n2d-highcpu-8 node, running Debian 11. In the LAN setting, latency was measured at 0.4ms. In the US WAN setting, the parties were divided equally among four datacenters in South Carolina, Nevada, Iowa, and Virginia, and the longest observed latency among these four was 60ms. In the Global WAN setting, the parties were divided equally among the four aforementioned US datacenters, plus datacenters in Montréal, São Paulo, London, the Netherlands, Zurich, Israel, Mumbai, Singapore, Taiwan, Seoul, Tokyo, and Sydney. The longest observed latency among these sixteen was 329ms. In each setting, we varied the number of parties (keeping an equal number in each datacenter) and collected at least 60 samples. Average wall-clock times for signing are reported in Table 1.⁴

2. Platform:

³Comprehensive in the sense that we cover a constant number of instances of each individual condition that should trigger an abort; we note that this is a sanity check only and by no means a guarantee in the same sense that formal verification of the code base would be.

⁴Note that these numbers are provisional, and the performance of our final reference implementation may differ.

Due to its reliance primarily on symmetric cryptography, plus a small number of operations in the signing curve, DKLs is appropriate for use on weak or power-constrained, including phones. No challenges are expected with regard to the baseline platform.

5. Licensing, Patent Claims, and Funding

1. The core code of our reference implementation, which includes all protocols described in this document, will be available under the Apache license. It depends upon a number of rust crates for features such as multi-threading; all of these crates are available under the Apache, BSD, or MIT licenses.
2. The team is not aware of any patents with claims covering the content of the planned submission. The team commits to avoid the pursuit of such patents in the future.

6. Comparison with Other Submissions

6.1. Differentiating Philosophy of Design

In order to minimize the surface area of potential specification, analysis, and implementation errors, this submission strives for minimal complexity while retaining strong security and concrete efficiency guarantees. Accordingly, it avoids the use of broadcast channels and synchrony entirely, restricts the use of private channels to one specific round during key-generation, and uses only symmetric cryptographic primitives previously standardized by NIST, along with operations on the NIST-standardized signing curve. In order to minimize the risk of insecurity due to advances in cryptanalysis, the security of this submission relies only upon the Diffie-Hellman assumption in the signing curve, and upon the security of symmetric-key primitives. In other words, this submission prioritizes a conservative approach to security.

DKLs is carefully constructed to avoid the use of sigma protocols or other zero-knowledge proofs, which enables it to achieve composable security without the use of straight-line extraction compilers and their associated overhead. It does not require extraction of secret key shares during the key generation protocol, which guarantees that the signing protocol remains secure even if combined with key generation protocols other than the one specified in our submission.

Notably, this submission forgoes advanced security properties such as identifiable abort and guaranteed output delivery, although routes toward enhancing the DKLs protocol to achieve these are known [CDKS24]. In the dishonest majority setting, these properties imply an authenticated broadcast channel, with all of the complexity such a broadcast channel entails in practice.

6.2. Specific Comparison

- **CGGMP** [CGGMP20]. This scheme addresses an identical threshold profile to our own, but to the best of our knowledge it only supports n -out-of- n signing under the security proof provided in the paper, whereas DKLs supports t -out-of- n signing for any $t \in [2, n]$. It

also contains one major building block (VOLE) which overlaps with a building block we specify, although the authors of CGGMP recommend a different instantiation of VOLE than we do. CGGMP requires a reliable, authenticated broadcast channel.⁵ It makes use of number theoretic components, including integer commitments and homomorphic encryption, and requires the acceptance of number theoretic assumptions. It also requires several zero-knowledge proofs during setup and pre-signing.

Both CGGMP and DKLs are proven secure in the UC model. Using the taxonomy of [CDLLR25], CGGMP realizes a type-two (or possibly type-four) ideal signature functionality with adaptive security. DKLs simultaneously realizes a type-three functionality and a type-four functionality, both with static security. CGGMP achieves identifiable abort, whereas DKLs does not attempt to do so. However, because the CGGMP security proof only covers n -out-of- n signing at the time of writing, identifiability does not necessarily imply that CGGMP is able to recover from failures.

A recent update to CGGMP adopts the same three-round structure originally introduced by DKLs. Therefore, both protocols are capable of signing in three rounds of communication. If configured for presignatures, both require one round to sign, and CGGMP requires three rounds to pre-sign, whereas DKLs requires two. We expect CGGMP to outperform DKLs in terms of bandwidth, and DKLs to outperform CGGMP in terms of computational costs.

- **BAM** [ABCGJM26]. For assumptions and building blocks, see our notes on CGGMP, but note that as a two-party-only protocol, BAM does not require broadcast. BAM satisfies only a game-based notion of security, and only in the generic group model. The security game defined by BAM permits adversarial nonce bias and nonce-dependent signatures, but in compensation for this, BAM requires only two rounds of interaction, whereas DKLs requires three if it is not configured for presignatures or pipelining.
- **THE CLASH**. This scheme addresses an identical threshold profile to our own. THE CLASH requires a reliable, authenticated broadcast channel.⁵ It makes use of class-group-based homomorphic encryption, and requires the acceptance of related class-group assumptions. It also requires several zero-knowledge proofs (including during signing).

THE CLASH satisfies only a game-based notion of security, and requires five rounds to produce a signature.⁶ We expect THE CLASH to outperform DKLs in terms of bandwidth, and DKLs to outperform THE CLASH in terms of computational and latency-induced costs.

- **TECLA** [CCLST19]. For assumptions and building blocks, see our notes on THE CLASH, but note that as a two-party-only protocol, TECLA does not require broadcast. Like DKLs, TECLA realizes a type-three⁷ threshold signature functionality; however, its security proof is only known to hold in the standalone model. It is not known to be concurrently composable.

⁵ We do not know of a proposed or extant standard for realizing a broadcast channel at the time of writing.

⁶We are unsure whether THE CLASH supports presignatures or identifiable abort, but we find both likely.

⁷See the taxonomy of [CDLLR25].

References

[ABCGJM26] Michael Adjedj, Constantin Blokh, Geoffroy Couteau, Arik Galansky, Antoine Joux, and Nikolaos Makriyannis. “Two-Round 2PC ECDSA at the Cost of 1 OLE: Applications to Embedded Cryptocurrency Wallets”. In: *Public-Key Cryptography - PKC 2026 - 29th IACR International Conference on Practice and Theory of Public-Key Cryptography, West Palm Beach, FL, USA, May 25-28, 2026, Proceedings, Part II*. Ed. by Shi Bai and Edoardo Persichetti. Vol. 16552. Lecture Notes in Computer Science. Springer, 2026, pp. 355–386. DOI: [10.1007/978-3-032-26734-4_12](https://doi.org/10.1007/978-3-032-26734-4_12). URL: https://doi.org/10.1007/978-3-032-26734-4_12. Available at <https://eprint.iacr.org/2024/1950>.

[Ash26] Gilad Asharov. *Revisiting DKLs Threshold ECDSA: Enhanced OT-based VOLE and Two-Party Signing*. Cryptology ePrint Archive, Paper 2026/976. 2026. URL: <https://eprint.iacr.org/2026/976>.

[Can01] Ran Canetti. “Universally Composable Security: A New Paradigm for Cryptographic Protocols”. In: *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, Las Vegas, Nevada, USA, October 14-17, 2001*. IEEE Computer Society, 2001, pp. 136–145. DOI: [10.1109/SFCS.2001.959888](https://doi.org/10.1109/SFCS.2001.959888). URL: <https://doi.org/10.1109/SFCS.2001.959888>. Available at <https://eprint.iacr.org/2000/067>.

[CCLST19] Guilhem Castagnos, Dario Catalano, Fabien Laguillaumie, Federico Savasta, and Ida Tucker. “Two-Party ECDSA from Hash Proof Systems and Efficient Instantiations”. In: *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11694. Lecture Notes in Computer Science. Springer, 2019, pp. 191–221. DOI: [10.1007/978-3-030-26954-8_7](https://doi.org/10.1007/978-3-030-26954-8_7). URL: https://doi.org/10.1007/978-3-030-26954-8_7. Available at <https://eprint.iacr.org/2019/503>.

[CDGLN18] Jan Camenisch, Manu Drijvers, Tommaso Gagliardoni, Anja Lehmann, and Gregory Neven. “The Wonderful World of Global Random Oracles”. In: *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. Lecture Notes in Computer Science. Springer, 2018, pp. 280–312. DOI: [10.1007/978-3-319-78381-9_11](https://doi.org/10.1007/978-3-319-78381-9_11). URL: https://doi.org/10.1007/978-3-319-78381-9_11. Available at <https://eprint.iacr.org/2018/165>.

[CDKS24] Ran Cohen, Jack Doerner, Yashvanth Kondi, and Abhi Shelat. “Secure Multiparty Computation with Identifiable Abort via Vindicating Release”. In: *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part VIII*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14927. Lecture Notes in Computer Science. Springer, 2024, pp. 36–73. DOI: [10.1007/978-3-031-68397-8_2](https://doi.org/10.1007/978-3-031-68397-8_2). URL: https://doi.org/10.1007/978-3-031-68397-8_2. Available at <https://eprint.iacr.org/2023/1136>.

[CDLLR25] Ran Cohen, Jack Doerner, Eysa Lee, Anna Lysyanskaya, and Lawrence Roy. “An Unstoppable Ideal Functionality for Signatures and a Modular Analysis of the Dolev-Strong Broadcast”. In: *Theory of Cryptography - 23rd International Conference, TCC 2025, Aarhus, Denmark, December 1-5,*

2025, *Proceedings, Part IV*. Ed. by Benny Applebaum and Huijia (Rachel) Lin. Vol. 16271. Lecture Notes in Computer Science. Springer, 2025, pp. 675–708. DOI: [10.1007/978-3-032-12290-2_22](https://doi.org/10.1007/978-3-032-12290-2_22). URL: https://doi.org/10.1007/978-3-032-12290-2_22. Available at <https://eprint.iacr.org/2024/1807>.

[CGGMP20] Ran Canetti, Rosario Gennaro, Steven Goldfeder, Nikolaos Makriyannis, and Udi Peled. “UC Non-Interactive, Proactive, Threshold ECDSA with Identifiable Aborts”. In: *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*. Ed. by Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna. ACM, 2020, pp. 1769–1787. DOI: [10.1145/3372297.3423367](https://doi.org/10.1145/3372297.3423367). URL: <https://doi.org/10.1145/3372297.3423367>. Available at <https://eprint.iacr.org/2021/060>.

[DKLs24] Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. “Threshold ECDSA in Three Rounds”. In: *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*. IEEE, 2024, pp. 3053–3071. DOI: [10.1109/SP54263.2024.00178](https://doi.org/10.1109/SP54263.2024.00178). URL: <https://doi.org/10.1109/SP54263.2024.00178>. Available at <https://eprint.iacr.org/2023/765>.

[DR26] Jack Doerner and Lawrence Roy. “Tighter Bounds for the Oblivious Bit-Fixing Inner Product Extractor on Biased Seeds”. In: *7th Conference on Information-Theoretic Cryptography, ITC 2026, University of California, Santa Barbara, CA, USA, August 14-15, 2026*. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2026. Available at <https://eprint.iacr.org/2026/654>.

[KMOS21] Yashvanth Kondi, Bernardo Magri, Claudio Orlandi, and Omer Shlomovits. “Refresh When You Wake Up: Proactive Threshold Wallets with Offline Devices”. In: *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*. IEEE, 2021, pp. 608–625. DOI: [10.1109/SP40001.2021.00067](https://doi.org/10.1109/SP40001.2021.00067). URL: <https://doi.org/10.1109/SP40001.2021.00067>. Available at <https://eprint.iacr.org/2019/1328>.

[MR19] Daniel Masny and Peter Rindal. “Endemic Oblivious Transfer”. In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*. Ed. by Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz. ACM, 2019, pp. 309–326. DOI: [10.1145/3319535.3354210](https://doi.org/10.1145/3319535.3354210). URL: <https://doi.org/10.1145/3319535.3354210>. Available at <https://eprint.iacr.org/2019/706>.

[Roy22] Lawrence Roy. “SoftSpokenOT: Quieter OT Extension from Small-Field Silent VOLE in the Minicrypt Model”. In: *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part I*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13507. Lecture Notes in Computer Science. Springer, 2022, pp. 657–687. DOI: [10.1007/978-3-031-15802-5_23](https://doi.org/10.1007/978-3-031-15802-5_23). URL: https://doi.org/10.1007/978-3-031-15802-5_23. Available at <https://eprint.iacr.org/2022/192>.

[Seg26] Gil Segev. *On the Statistical vs. Computational Security of the DKLs23 Multiparty ECDSA Protocol*. Cryptology ePrint Archive, Paper 2026/929. 2026. URL: <https://eprint.iacr.org/2026/929>.

[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).