

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

Title: CL and Threshold-CL

Subtitle: (Threshold) Linearly Homomorphic PKE from Class Group

Version: Preview Writeup 1.0 (2026-07-31)¹

Team name: CL team

Team members: Cyril Bouvier, Lennart Braun, Guilhem Castagnos, Quentin Combal, Fabien Laguillaumie, Kelsey Melissaris, Ida Tucker

Abstract:

This Preview Writeup defines the starting plan for submitting CL and Threshold-CL to the NIST First Call for Multi-Party Threshold Schemes. These are respectively a linearly homomorphic encryption scheme and its threshold variant whose security relies on problems in class groups. (Threshold) linearly homomorphic encryption is a key tool to design secure multiparty protocols. Our primitives and protocols are accompanied by a benchmarked implementation using the BICYCL library, an optimized and specialized library for class group cryptography. We show that the CL encryption scheme is faster and more compact than Paillier encryption. Besides, the unique class-group setting offers an original solution to design MPC protocols without trusted setup. A first variant of the distributed key generation and threshold decryption protocol is proven actively secure in the UC framework with guaranteed output delivery in the honest majority setting against static adversaries. A second variant is adaptively secure in the single inconsistent player (SIP) model.

Proposed crypto-systems:

- (I) CL Encryption (Categories S7)
- (II) Threshold-CL Encryption (Static Security) (Categories S7)
- (III) Threshold-CL Encryption (Adaptive Security) (Categories S7)

Keywords: Threshold Cryptography; NIST Threshold Call; (Threshold) Linearly Homomorphic Public Key Encryption; Class Groups

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

¹Version submitted to NIST-MPTC for publication

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: Cyril Bouvier^{i1,a1}, Lennart Braun^{i2,a3}, Guilhem Castagnos^{i3,a2}, Quentin Combal^{i4,a1}, Fabien Laguillaumie^{i5,a1}, Kelsey Melissaris^{i6,a3}, Ida Tucker^{i7,a1}

Open Researcher and Contributor Identifiers (ORCID):

i1 (0009-0006-6722-4887); i2 (0000-0001-9164-305X); i3 (0009-0004-5857-7889); i4 (0009-0003-8653-9444); i5 (0000-0001-6464-1139); i6 (0000-0001-6840-9863); i7 (0000-0003-4895-5896)

Affiliations: ^{a1} Université de Montpellier, CNRS, LIRMM @ Montpellier, France. ^{a2} Université de Bordeaux, CNRS, INRIA, IMB @ Talence, France. ^{a3} Université Paris Cité, CNRS, IRIF @ Paris, France.

Main contacts:

- **Team mailing list:** cl-team@lirmm.fr
- **Primary technical contact person:** Guilhem Castagnos, guilhem.castagnos@math.u-bordeaux.fr
- **Secondary contact person 1:** Fabien Laguillaumie, fabien.laguillaumie@lirmm.fr
- **Secondary contact person 2:** Lennart Braun, lbraun@irif.fr

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

Secure multiparty computation (MPC) allows a set of N parties to jointly compute an agreed function on their private inputs. During this interactive protocol, no other information than the result of the evaluation is revealed even if up to t parties are corrupted by an adversary, for some threshold $1 \leq t < N$.

Two fundamental tools that are widely used to make MPC protocols efficient and secure are *Linearly Homomorphic Encryption* (LHE) and *Threshold Linearly Homomorphic Encryption* (TLHE). Such encryption schemes allow to compute linear functions of encrypted messages. The latter is additionally equipped with (interactive) distributed key generation and distributed decryption protocols, where each party owns a secret share of the secret decryption key.

LHE is the heart of many secure distributed protocols, for instance to thresholdize signatures, to securely compute on private sets, to privately retrieve distributed information, or in federated learning or electronic voting. On the other hand, a general framework was proposed in [CDN01] to implement MPC for general functions relying on any TLHE and appropriate zero-knowledge proofs. This CDN paradigm is also well-adapted to the You Only Speak Once (YOSO) model for MPC [GHKMNR21]. TLHE is a building block of Encryption Switching Protocols (ESP) that imply general two-party computation [CPP16].

This preview document presents a linearly encryption scheme that relies on class groups and its threshold variants, and that will be submitted in the S7 category, as gadgets. It is the Castagnos-Laguillaumie (CL) encryption [CL15] described in [CLT18] and its threshold variant presented in [BCDLMOT24; BCDLMOT25]. Algorithmic problems in class groups are generally harder than those in finite fields or related to integer factorization, which makes it possible to shorten cryptographic parameters at a given security level. For this reason, CL encryption scheme is faster and more compact than Paillier's encryption. Besides, the unique class-group setting offers an original solution to design MPC protocols without trusted setup.

This is the reason why both the threshold and the non-threshold schemes are building blocks for many multiparty computation protocols. For instance, the general CDN framework has been adapted to class groups [BDO23] as well as ESP [CIL17]. The regular CL encryption has also been proved very useful in many MPC scenarios like threshold ECDSA [CCLST20][LLZD25][DNP25] and random beacons [TCLM21].

The proposal is accompanied by an implementation using the BICYCL library [BCIL23; BIC]

2. Specification

2.1. Organization

The specification document will include three cryptosystems, the CL encryption scheme and two threshold variants for static and adaptive security.

2.2. System model

Protocol overview:

The CL encryption scheme that we implement uses the framework of a cyclic group with an easy discrete logarithm subgroup as introduced in [CL15]: it consists of a group G of unknown order which is a direct product $F \times G^q$ of a subgroup F of prime order q and the subgroup G^q of q -th powers of G . If we denote $s = |G|/q$, then we require that $\gcd(q, s) = 1$. Computing discrete logarithm (DL) in F can be done efficiently. This original scheme is similar to ElGamal in the exponent, where the plaintext message is encoded in the subgroup F where the DL problem is easy. Therefore, during decryption we can recover the message whatever its size, and we get fully linear homomorphic computation modulo q . The scheme we present here is the variant introduced in [CLT18]. The idea consists of using a subgroup decomposition problem to hide the message in the second component of the ciphertext. This scheme is actually a generalization of the variant of Paillier, proposed by Camenisch and Shoup in [CS03].

In [BDO23], a statically secure threshold version of the CL cryptosystem is described, including a distributed key generation protocol. To get malicious security, [BDO23] uses efficient Σ -protocols for parties to prove correct behaviour. The versions that will be submitted were originally described in [BCDLMOT24]. They improve upon the previous construction, by the use of a weaker reconstruction property of the underlying VSSs allowing the removal of a costly proof of knowledge, and include a protocol variant with adaptive (SIP) security. Compared to [BCDLMOT24], we also use a more efficient batching technique from [BCL25].

Networking aspects:

The threshold version of the CL encryption scheme requires reliable broadcast and secure channels.

2.3. Security

The linearly homomorphic encryption that will be submitted achieves indistinguishability under chosen plaintext attacks assuming a hard subgroup problem in G .

A first variant of the threshold protocol is actively secure in the UC framework with guaranteed output delivery, tolerating up to $t < N/2$ static corruptions under the Unknown Order Assumption and the Rough Order Assumption. The second one tolerates up to $t < N/2$ adaptive corruptions in the single inconsistent played (SIP) model and provides guaranteed output delivery under the same assumptions.

3. Open-Source Implementation

Code structure: Our core code will mainly consist of the BICYCL library ([BCIL23],[BIC]), our open source C++ library for class groups based cryptography. BICYCL implements arithmetic in

the ideal class groups of imaginary quadratic fields, along with a set of cryptographic primitives based on class groups. It started as an efficient implementation of the CL encryption scheme, and now also implements its threshold variant for static security, as well as other multi-party schemes based on CL. BICYCL relies on the GNU Multiple Precision Arithmetic Library (GMP) for arithmetic operations over big integers, as well as the OpenSSL library for arithmetic on elliptic curves. Both libraries are written in C. For simplicity, we plan to include GMP and OpenSSL as external dependencies, installed through the distribution's package manager. Compilation will be performed using GCC, GNU make, and Cmake, included as external dependencies as well. As our core code does not rely on architecture-specific features, we only select the compiler options for general optimization (e.g. `-O3`). Still, we will consider architecture-specific options if we notice significant performance improvements.

Code progress and availability: Currently BICYCL holds implementations for the CL encryption scheme and for its threshold variant with static security (Threshold-CL static). We plan on implementing Threshold-CL with adaptive security in the same library. Additionally some rework of the implementations may be needed to make the code clearer and ease the auditing process. The git repository for BICYCL is publicly available and hosted on the LIRMM's GitLab instance [BIC]. To comply with the submission requirements, we will develop additional C++ code to take care of the networking between parties. This code will rely on the BICYCL library to perform protocol operations for each party.

Implementation of the networking model: Our protocol relies on broadcast and point-to-point communication. For the latter, the protocol assumes secure channels between each pair of players, which can be implemented using the Transport Layer Security (TLS) protocol. The broadcast channel does not need to be encrypted, but requires authentication nonetheless. Our implementation will use a best-effort broadcast based on the point-to-point links. Again, TLS can be used for providing authentication. We note that our protocol requires a reliable broadcast channel, but we consider the implementation of such a construction to be out of our scope for now. We will keep our implementation modular so that a reliable broadcast protocol can be plugged-in at a later point, or for deployment. To run the benchmark on the baseline platform, we plan to instantiate our channels using unauthenticated UDP or TCP sockets for simplicity, but will aim to integrate TLS, depending on the available time. Each party will be run as a separate process, and parties will communicate through the loopback interface available on GNU/Linux systems.

Testing: We plan to have simple tests where at least one party is malicious and attempts to "cheat" at different points in the protocol. The implementation will check that in each case, the protocol is aborted by honest parties.

4. Experimental Performance Evaluation

Performance: We give a comparison of Pailler and CL in Table 1. The values are taken from BICYCL (commit 797718e) in which both protocols are implemented. The machine we used has

16 x64 CPU cores (up to 5 GHz) and 16 GB of RAM. For security levels of 128 bits and higher, CL provides both faster computation and smaller ciphertexts. CL also scales better at higher security levels, being almost 3 times as fast as Paillier for 192-bit security. Having smaller ciphertexts is a very desirable property when building multi-party protocols, as it reduces the bandwidth needed for communication between parties. This makes CL a competitive linearly homomorphic encryption scheme.

In Table 2 we give timings and communication sizes for Threshold-CL (static security), also taken from BICYCL from the same commit. Threshold decryption using Threshold-CL is both efficient and light on bandwidth usage. Moreover, the decryption phase scales very well with larger numbers of participants, both in terms of execution time and communication size. We believe Threshold-CL can be competitive for all sizes of set of participants, starting at 3; that is, using the Threshold Call labels, Three, Small ($N > 3$), Medium ($N > 8$), Large ($N > 64$) or Enormous ($N > 1024$).

κ	Parameters		Encryption (ms)		Decryption (ms)		Ct size (kbit)	
	Paillier (RSA mod.)	CL (Δ_K)	Paillier	CL	Paillier	CL	Paillier	CL
112	2048 bits	1348 bits	6	6	6	7	4.1	3.7
128	3072 bits	1827 bits	14	11	14	13	6.1	4.7
192	7680 bits	3598 bits	148	42	148	57	15.4	8.7
256	15360 bits	5971 bits	870	120	870	150	30.7	14.0

Table 1: Comparison between Paillier and CL from BICYCL benchmarks. κ refers to the computational security parameter. The RSA modulus size is fixed according to the most recent NIST recommendations [NIST-SP80057p1].

N	3	5	7	10	20	50	100	200
Execution (ms)								
Keygen	77	129	189	280	675	2 776	9 500	37 000
Decrypt	100	121	122	127	147	221	373	862
Communication per party (kB)								
Keygen	1.1	1.6	2.1	2.8	5.5	14.7	35	96
Decrypt	0.41	0.55	0.61	0.66	0.74	0.82	0.89	1.02

Table 2: Threshold-CL (static security) timing and communication from BICYCL benchmarks. With $T = N/2 - 1$, computational security $\kappa = 128$ bits, statistical security $\sigma = 40$ bits. Operations are performed on a single thread.

Platform: The current benchmark for Threshold-CL runs all parties consecutively in a single process. Communication is simulated, their execution time is not taken into account; therefore, benchmarking Threshold-CL with parties running as separate processes and communicating over a network, as recommended by the NIST Threshold Call, is expected to give longer timings. Moreover, with a one-process-per-party benchmark, resources have to be shared. We estimate

that with 64 GB of RAM, we can go up to $N = 500$ concurrent parties for Threshold-CL with our current implementation. But to measure performance accurately, each party should have access to at least one CPU core. Therefore, for the one-process-per-party benchmark of Threshold-CL, we plan to limit our benchmark cases to have one CPU core per party, which would give $N \leq 16$ if we don't consider hyper-threading and other system processes that run on the same machine. For cases with larger N , we will use a single-process benchmark. Regarding CL (non-threshold), the baseline platform will be more than adequate.

5. Licensing, Patent Claims, and Funding

Licenses: Our main library, BICYCL, relies on the GMP and OpenSSL libraries. Both libraries are free software and licensed under OSI approved licenses (LGPL version 3.0 for GMP, Apache license version 2.0 for OpenSSL). BICYCL itself is licensed under GPL version 3.0 or later, a license compatible with both LGPL v3.0 and Apache v2.0. Regarding the additional code needed for the submission, we will release it under GPLv3 as well. Building uses tools from the GNU toolchain (gcc, make, ...) that are licensed under GPL, as well as CMake, licensed under the BSD 3-clause license. All the aforementioned licenses are approved by the Open-Source Initiative.

Patent claims: We declare that, to the best of our knowledge, no patents cover the contents of our submission. The copyright holders of our core code are the team members who contributed to its development.

Funding: The research associated with our submission is supported by the French Agence Nationale de la Recherche (ANR) project ANR-21-CE39-0006 SANGRIA, the France 2030 ANR project ANR-22-PECY-003 SecureCompute, the ANR ASTRID program under the national project AMIRAL with reference ANR-21-ASTR-0016, and by ICO, Institut Cybersecrit  Occitanie, funded by R gion Occitanie, France.

References

[BCDLMOT24] Lennart Braun, Guilhem Castagnos, Ivan Damgård, Fabien Laguillaumie, Kelsey Melissaris, Claudio Orlandi, and Ida Tucker. “An Improved Threshold Homomorphic Cryptosystem Based on Class Groups”. In: *SCN 24, Part II*. Ed. by Clemente Galdi and Duong Hieu Phan. Vol. 14974. LNCS. Springer, Cham, September 2024, pp. 24–46. DOI: [10.1007/978-3-031-71073-5_2](https://doi.org/10.1007/978-3-031-71073-5_2).

[BCDLMOT25] Lennart Braun, Guilhem Castagnos, Ivan Damgård, Fabien Laguillaumie, Kelsey Melissaris, Claudio Orlandi, and Ida Tucker. “An Improved Threshold Homomorphic Cryptosystem Based on Class Groups”. In: *Cryptogr. Commun.* 17.5 (2025), pp. 1367–1406. DOI: [10.1007/s12095-025-00826-2](https://doi.org/10.1007/s12095-025-00826-2). URL: <https://doi.org/10.1007/s12095-025-00826-2>.

[BCIL23] Cyril Bouvier, Guilhem Castagnos, Laurent Imbert, and Fabien Laguillaumie. “I Want to Ride My BICYCL : BICYCL Implements CryptographY in CLass Groups”. In: *Journal of Cryptology* 36.3 (July 2023), p. 17. DOI: [10.1007/s00145-023-09459-1](https://doi.org/10.1007/s00145-023-09459-1).

[BCL25] Agathe Beaugrand, Guilhem Castagnos, and Fabien Laguillaumie. “Efficient Succinct Zero-Knowledge Arguments in the CL Framework”. In: *Journal of Cryptology* 38.1 (January 2025), p. 13. DOI: [10.1007/s00145-024-09534-1](https://doi.org/10.1007/s00145-024-09534-1).

[BDO23] Lennart Braun, Ivan Damgård, and Claudio Orlandi. “Secure Multiparty Computation from Threshold Encryption Based on Class Groups”. In: *CRYPTO 2023, Part I*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14081. LNCS. Springer, Cham, August 2023, pp. 613–645. DOI: [10.1007/978-3-031-38557-5_20](https://doi.org/10.1007/978-3-031-38557-5_20).

[BIC] *BICYCL public git repository*. URL: <https://gite.lirmm.fr/crypto/bicycl>.

[CCLST20] Guilhem Castagnos, Dario Catalano, Fabien Laguillaumie, Federico Savasta, and Ida Tucker. *Bandwidth-Efficient Threshold EC-DSA*. Ed. by Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas. May 2020. DOI: [10.1007/978-3-030-45388-6_10](https://doi.org/10.1007/978-3-030-45388-6_10).

[CDN01] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. “Multiparty Computation from Threshold Homomorphic Encryption”. In: *EUROCRYPT 2001*. Ed. by Birgit Pfitzmann. Vol. 2045. LNCS. Springer, Berlin, Heidelberg, May 2001, pp. 280–299. DOI: [10.1007/3-540-44987-6_18](https://doi.org/10.1007/3-540-44987-6_18).

[CIL17] Guilhem Castagnos, Laurent Imbert, and Fabien Laguillaumie. “Encryption Switching Protocols Revisited: Switching Modulo p ”. In: *CRYPTO 2017, Part I*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10401. LNCS. Springer, Cham, August 2017, pp. 255–287. DOI: [10.1007/978-3-319-63688-7_9](https://doi.org/10.1007/978-3-319-63688-7_9).

[CL15] Guilhem Castagnos and Fabien Laguillaumie. “Linearly Homomorphic Encryption from DDH”. In: *CT-RSA 2015*. Ed. by Kaisa Nyberg. Vol. 9048. LNCS. Springer, Cham, April 2015, pp. 487–505. DOI: [10.1007/978-3-319-16715-2_26](https://doi.org/10.1007/978-3-319-16715-2_26).

[CLT18] Guilhem Castagnos, Fabien Laguillaumie, and Ida Tucker. “Practical Fully Secure Unrestricted Inner Product Functional Encryption Modulo p ”. In: *ASIACRYPT 2018, Part II*. Ed. by Thomas Peyrin and

Steven Galbraith. Vol. 11273. LNCS. Springer, Cham, December 2018, pp. 733–764. DOI: [10.1007/978-3-030-03329-3_25](https://doi.org/10.1007/978-3-030-03329-3_25).

[CPP16] Geoffroy Couteau, Thomas Peters, and David Pointcheval. “Encryption Switching Protocols”. In: *CRYPTO 2016, Part I*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9814. LNCS. Springer, Berlin, Heidelberg, August 2016, pp. 308–338. DOI: [10.1007/978-3-662-53018-4_12](https://doi.org/10.1007/978-3-662-53018-4_12).

[CS03] Jan Camenisch and Victor Shoup. “Practical Verifiable Encryption and Decryption of Discrete Logarithms”. In: *CRYPTO 2003*. Ed. by Dan Boneh. Vol. 2729. LNCS. Springer, Berlin, Heidelberg, August 2003, pp. 126–144. DOI: [10.1007/978-3-540-45146-4_8](https://doi.org/10.1007/978-3-540-45146-4_8).

[DNP25] Hila Dahari-Garbian, Ariel Nof, and Luke Parker. “Trout: Two-Round Threshold ECDSA from Class Groups”. In: *ACM CCS 2025*. Ed. by Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier. ACM Press, October 2025, pp. 380–393. DOI: [10.1145/3719027.3765192](https://doi.org/10.1145/3719027.3765192).

[GHKMNR21] Craig Gentry, Shai Halevi, Hugo Krawczyk, Bernardo Magri, Jesper Buus Nielsen, Tal Rabin, and Sophia Yakubov. “YOSO: You Only Speak Once - Secure MPC with Stateless Ephemeral Roles”. In: *CRYPTO 2021, Part II*. Ed. by Tal Malkin and Chris Peikert. Vol. 12826. LNCS. Virtual Event: Springer, Cham, August 2021, pp. 64–93. DOI: [10.1007/978-3-030-84245-1_3](https://doi.org/10.1007/978-3-030-84245-1_3).

[LLZD25] Yingjie Lyu, Zengpeng Li, Hong-Sheng Zhou, and Xudong Deng. “Threshold ECDSA in Two Rounds”. In: *ACM CCS 2025*. Ed. by Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier. ACM Press, October 2025, pp. 2937–2950. DOI: [10.1145/3719027.3765176](https://doi.org/10.1145/3719027.3765176).

[NIST-SP80057p1] Elaine Barker. *Recommendation for Key Management: Part 1 - General*. National Institute of Standards and Technology (NIST). 2020. DOI: [10.6028/NIST.SP.800-57pt1r5](https://doi.org/10.6028/NIST.SP.800-57pt1r5). URL: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf>.

[TCLM21] Sri Aravinda Krishnan Thyagarajan, Guilhem Castagnos, Fabien Laguillaumie, and Giulio Malavolta. “Efficient CCA Timed Commitments in Class Groups”. In: *ACM CCS 2021*. Ed. by Giovanni Vigna and Elaine Shi. ACM Press, November 2021, pp. 2663–2684. DOI: [10.1145/3460120.3484773](https://doi.org/10.1145/3460120.3484773).

[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).