

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

Title: ark-pq: post-quantum zero-knowledge arguments of knowledge

Version: Preview Writeup 1.0 (2026-09-28)¹

Team name: ark-pq: Post-quantum arkworks

Team members: Alessandro Chiesa, Giacomo Fenzi, Christian Knabenhans, Andrew Zitek-Estrada

Abstract: ark-pq is a library for post-quantum zero-knowledge succinct non-interactive arguments of knowledge (zkSNARKs). ark-pq can prove sensitive claims about threshold cryptographic primitives without exposing sensitive data (e.g., knowledge of a secret key, or the validity of a signature), as well as an extensive range of other complex well-formedness conditions.

ark-pq is built from two very well-understood primitives: interactive oracle proofs (which are information-theoretically secure) and Merkle tree vector commitments (which are built from hash functions). ark-pq is designed to be modular and configurable, allowing for the construction of a wide range of zkSNARKs with different trade-offs between proof size, prover time, and verifier time.

Proposed crypto-systems:

- (I) arguments (Categories S6)

Keywords: Threshold Cryptography; NIST Threshold Call; Zero-knowledge Proof; zkSNARK

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: Alessandro Chiesa^{i1,a1†}, Giacomo Fenzi^{i2,a1,*}, Christian Knabenhans^{i3,a1*}, Andrew Zitek-Estrada^{i4,a1‡}

Open Researcher and Contributor Identifiers (ORCID):

i1 (0000-0002-3029-2353); i2 (0000-0003-3702-1780); i3 (0000-0003-4645-7295); i4 (0009-0001-2974-3065)

Affiliations:

^{a1} COMPSEC lab, School of Computer and Communication Sciences, EPFL, Switzerland

Associateship clarifications:

* Ph.D. student. † Associate professor. ‡ Research engineer.

Main contacts:

- **Team mailing list:** ark-pq@groupes.epfl.ch
- **Primary technical contact person:** Alessandro Chiesa, alessandro.chiesa@epfl.ch

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. Overview

Zero-knowledge proofs can verify sensitive claims without exposing sensitive data. For example, they can prove knowledge of a secret key or the validity of a signature. Since they support arbitrary NP statements, they can be used to prove an extensive range of complex well-formedness conditions. At the same time, post-quantum security is no longer a distant planning exercise. It must be addressed when constructing zero-knowledge proof systems.

We propose a software stack for post-quantum zero knowledge focused on interactive oracle proofs (IOPs). IOPs are a family of information-theoretic proof systems that allow a prover to convince a verifier about the validity of a statement through a sequence of proof strings to which the verifier has oracle access. Our constructions require no trusted setup (i.e., they are *transparent*), are post-quantum secure, and support arbitrary NP statements.

The goal is to make post-quantum zero knowledge easier to develop by providing libraries that are modular, extensible, and ergonomic. We are developing the fundamental building blocks for IOPs, including but not limited to the sumcheck protocol and hash-based vector commitments. We provide theory-informed interfaces and abstractions needed to reason about IOPs. We additionally provide methodology for security analysis, including feedback about how configurations achieve different efficiency trade-offs (proving time, proof size, verifier time).

1.1. Interactive oracle proofs

Interactive oracle proofs (IOPs) are information-theoretic proof systems that allow a prover to convince a verifier of the validity of a statement by sending a sequence of proof strings, to which the verifier has oracle access.

IOPs are configurable in multiple ways (e.g., through parameters of an underlying error-correcting code), which allows for different trade-offs between prover time, proof size, and verifier time.

Our submission will include implementations of multiple IOPs. As part of this implementation, we will provide an optimized and flexible implementation of the sumcheck protocol [LFKN90; BCFFMMZ25], which is a fundamental building block for many IOPs.²

1.2. Merkle vector commitments

A vector commitment is a cryptographic primitive that allows one to commit to a vector of values and later prove the value at a specific index without revealing the entire vector. A common construction of vector commitments is based on Merkle trees, where each leaf node represents a value in the vector and each non-leaf node represents the hash of its children. Merkle vector commitments can be instantiated with any collision-resistant hash function, and thus offer post-quantum security.

²A preliminary optimized and configurable implementation of the sumcheck protocol is available at <https://github.com/compsec-epfl/efficient-sumcheck>.

Despite their apparent simplicity, Merkle vector commitments can be highly configurable. For example, leaves may be encoded in various ways, and each level of the tree may specify a separate arity and hash function.

Our submission will include an implementation of Merkle vector commitments, with support for flexible configurations.

1.3. Arguments

An IOP and a vector commitment scheme can be compiled into a succinct (interactive) argument via the BCS transformation [BCS16]; the resulting argument is a proof system that is sound against computationally bounded adversaries. If the IOP is knowledge sound, the resulting argument is also knowledge sound. If the IOP is zero-knowledge and the vector commitment scheme is hiding, then the resulting argument is also zero-knowledge.

A (public-coin) interactive argument can be compiled into a non-interactive argument using the Fiat–Shamir transform by deriving verifier challenges from the protocol transcript. Interactive arguments in ark-pq are public-coin. If the starting interactive argument is succinct, zero-knowledge, and knowledge sound, the resulting argument is a succinct non-interactive argument of knowledge (zkSNARK).

ark-pq implements IOPs and vector commitment schemes, and provides the BCS and Fiat–Shamir transformations to build zkSNARKs from these building blocks.

1.4. The arkworks ecosystem

arkworks [ark] is an ecosystem of Rust libraries for zkSNARKs and related tools (fields and groups, serialization, circuits, Fiat–Shamir transform, etc.). Rust crates in the arkworks ecosystem are downloaded hundreds of thousands of times daily, and are used both in academia and industry.

A key integration point is the Fiat–Shamir transform, which makes a public-coin interactive protocol non-interactive by deriving verifier challenges from the protocol transcript. arkworks implements the CFRG Internet-Draft on the Fiat–Shamir transformation [Orr25]. Where applicable to the IOP setting, our transcript layer will follow this interface, making protocol instantiations compatible with the standard.

Our submission is intended as a post-quantum extension of the arkworks ecosystem, and will be implemented as a set of Rust crates that are compatible with, but not strongly tied to, the existing arkworks ecosystem. This will allow users to easily integrate our post-quantum zkSNARKs into existing applications that already use the arkworks ecosystem, as well as to bridge the gap to other protocols and applications relying on a variety of cryptographic primitives.

1.5. Applications

The zkSNARKs in ark-pq can prove arbitrary NP statements, and thus can be used to prove an extensive range of complex well-formedness conditions. We envision ark-pq to be used in a variety of applications, including but not limited to threshold cryptography.

In particular, ark-pq can be used to prove the “Example Proofs of Interest” outlined in the threshold call [NIST-IR8214C, Appendix B.2.1] (correct key generation for a signature scheme, knowledge of a (secret) signature, proof of public key encryption, proof of public key encryption). Thanks to its generality, ark-pq can also be used to prove batches of such statements, combinations of such statements, and more complex statements (e.g., proving additional predicates about the secret witness).

2. Specification

2.1. Organization

The specification document will be organized as follows:

Description and security analysis: Describes the proposed interfaces (including interfaces for interactive oracle proofs, vector commitment schemes, interactive arguments, and non-interactive arguments), generic transformations to build arguments (including the BCS and Fiat–Shamir transformations), and security analyses (including soundness and knowledge soundness).

Instantiation: Describes concrete instantiations of interactive oracle proofs and Merkle vector commitments. This includes Merkle vector commitments built from NIST-standardized hash functions (e.g., SHA-2), and may include other hash functions (e.g., Blake3, Poseidon).

Implementation and evaluation: Implementation of the proposed proof systems and evaluations.

2.2. System model

Our zkSNARKs do not require a trusted setup. The proofs are non-interactive and do not require networking. Our primitive is non-thresholded (involving only a single prover and a single verifier), but can be used as a building block for threshold protocols.

2.3. Security

Our zkSNARKs are zero-knowledge and knowledge sound in the random oracle model, and are post-quantum secure. We target the 128-bit security level.

3. Open-Source Implementation

3.1. Code structure

Our core code will be implemented in Rust, and will include submodules for interactive oracle proofs, Merkle vector commitments, and generic transformations.

3.2. Code progress and availability

The code is under active development and is closing in on a first public release. A public Git-compatible repository will be made available.

3.3. Implementation of the networking model

Not applicable, as our zkSNARKs are non-interactive and do not require networking.

3.4. Testing

We will provide a comprehensive test suite to provide assurance of the correctness and security of our implementation. This will include unit tests for individual components, integration tests for the overall system, and performance benchmarks.

4. Experimental Performance Evaluation

4.1. Performance

We do not yet have performance results to report, but we will provide a comprehensive evaluation of our implementation, including measurements of proving time, proof size, and verifier time for different configurations of our zkSNARKs.

4.2. Platform

We do not anticipate any significant challenges when using the baseline platform. Our implementation may have platform-specific optimizations for architectures with vector instruction sets.

5. Licensing, Patent Claims, and Funding

The core code will be made available under a dual MIT and Apache 2.0 license.

To the best of our knowledge, there are no known patents that may cover the contents of the submission.

6. Comparison with Other Submissions

We compare our proposal to the other S6 proposals *Schmivitz* [BOSRHMRP26], *SmallWood* [FR26], and *LaZer* [LNOSS26]. All proposals introduce cryptographic primitives which are post-quantum secure and do not require a trusted setup.

LaZer targets statements about lattice-based cryptography, while our proposal targets arbitrary NP statements. Further, *LaZer* relies on lattice assumptions, while our proposal relies solely on hash-based primitives.

Schmivitz, *SmallWood*, and our proposal target arbitrary NP statements, albeit at different scales. *Schmivitz* relies on VOLE-in-the-head techniques, which are typically very efficient for small statements. *SmallWood* targets moderate-size statements. Our proposal targets large statements, allowing for complex statements about thresholded cryptographic primitives.

Further, our proposal explicitly targets modularity and flexibility, allowing users to choose among different configurations to achieve different efficiency trade-offs (proving time, proof size, verifier time).

References

[ark] arkworks contributors. *arkworks zkSNARK ecosystem*. URL: <https://github.com/arkworks-rs/>.

[BCFFMMZ25] Anubhav Baweja, Alessandro Chiesa, Elisabetta Fedele, Giacomo Fenzi, Pratyush Mishra, Tushar Mopuri, and Andrew Zitek-Estrada. “Time-Space Trade-Offs for Sumcheck”. 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. 37–70. DOI: [10.1007/978-3-032-12290-2_2](https://doi.org/10.1007/978-3-032-12290-2_2). Also at ia.cr/2025/1473.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. “Interactive Oracle Proofs”. In: *Proceedings of the 14th Theory of Cryptography Conference*. TCC '16-B. 2016, pp. 31–60.

[BOSRHMRP26] Carsten Baum, Emmanuela Orsini, Peter Scholl, Benoit Razet, Marcella Hastings, Shibam Mukherjee, Christian Rechberger, and James Parker. *Schmivitz: VOLEitH based ZK gadgets for threshold cryptography*. Preview writeup submitted to the NIST Multi-Party Threshold Cryptography Call. Version 1.0 (2026-01-20). 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Schmivitz-PW01.pdf>.

[FR26] Thibault Feneuil and Matthieu Rivain. *SmallWood: Hash-Based Zero-Knowledge Arguments for Relatively Small Instances*. Preview writeup submitted to the NIST Multi-Party Threshold Cryptography Call. Version 1.0 (2026-01-20). 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/SmallWood-PW01.pdf>.

[LFKN90] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. “Algebraic Methods for Interactive Proof Systems”. In: *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*. IEEE Computer Society, 1990, pp. 2–10. DOI: [10.1109/FSCS.1990.89518](https://doi.org/10.1109/FSCS.1990.89518). Journal version (1992) at [DOI:10.1145/146585.146605](https://doi.org/10.1145/146585.146605).

[LNOSS26] Vadim Lyubashevsky, Ngoc Khanh Nguyen, Michał Osadnik, Gregor Seiler, and Patrick Steuer. *LaZer: Lattice-Based Zero-Knowledge Arguments for Lattice Relations*. Preview writeup submitted to the NIST Multi-Party Threshold Cryptography Call. Preview Writeup 0.1 (2026-06-24). 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/LaZer-PW02.pdf>.

[Orr25] Michele Orrù. *Fiat-Shamir Transformation*. Internet-Draft, Crypto Forum Research Group. 2025. URL: <https://datatracker.ietf.org/doc/draft-irtf-cfrg-fiat-shamir/>.

[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).