

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

**Title:** Distributed Schnorr Signatures

**Subtitle:** Classic Schnorr

**Version:** 1.0 (2026-01-15)<sup>1</sup>

**Team name:** Fireblocks–3MI Labs

**Team members:** Michael Adjedj, Tomer Ashur, Amit Singh Bhati, Geoffroy Couteau, Cyprien Delpech de Saint Guilhem, Michael Gutkin, Nikos Makriyannis

**Abstract:** This preview write-up presents the plan to submit the classic Schnorr construction for many-party EdDSA signing.

**Proposed crypto-systems:** (I) Classic Schnorr EdDSA Signing (Categories N1.1 and N4).

**Keywords:** Threshold Cryptography; NIST Threshold Call; EC Key Generation; EdDSA Signature

---

<sup>1</sup>Version 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:** Michael Adjedj<sup>i1,a1</sup>, Tomer Ashur<sup>i2,a2</sup>, Amit Singh Bhati<sup>i3,a2</sup>, Geoffroy Couteau<sup>i4,a3\*</sup>, Cyprien Delpech de Saint Guilhem<sup>i5,a2</sup>, Michael Gutkin<sup>i6,a1</sup>, Nikos Makriyannis<sup>i7,a1</sup>

### Open Researcher and Contributor Identifiers (ORCID):

i1 (0009-0001-2738-9728); i2 (0000-0001-6091-4857); i3 (0000-0003-0843-4885); i4 (0000-0002-6645-0106); i5 (0000-0002-0147-2566); i6 (0009-0002-5533-7420); i7 (0000-0002-9818-456X)

### Affiliations:

<sup>a1</sup> Fireblocks, New York City, NY, United States of America

<sup>a2</sup> 3MI Labs, Leuven, Belgium

<sup>a3</sup> Université Paris Diderot, Paris, France

### Associateship clarifications:

\* Consultant at Fireblocks Ltd.

### Main contacts:

- **Team mailing list:** [NIST-MPTC@fireblocks.com](mailto:NIST-MPTC@fireblocks.com)
- **Primary technical contact person:** Nikos Makriyannis, [nikos@fireblocks.com](mailto:nikos@fireblocks.com)
- **Secondary contact person 1:** Michael Gutkin, [gutkin@fireblocks.com](mailto:gutkin@fireblocks.com)
- **Secondary contact person 2:** Cyprien Delpech de Saint Guilhem, [cyprien@3milabs.tech](mailto:cyprien@3milabs.tech)

**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

**Principal Crypto-Systems.** The planned package submission will specify a protocol in the family of threshold schemes for Schnorr signatures [MOR01; SS01; NKDM03; Mak22]. This protocol fits within categories N1.1 and N4 (subcategory QV-ECC-KeyGen). This protocol is essentially the folklore protocol for threshold Schnorr signatures, dubbed as *Classic Schnorr* in [Mak22].

# 2. Specification

**Organization.** The main parts of the planned submission package will specify the *Classic Schnorr* protocol. The protocol proceeds in two phases: a key generation phase and a signing phase. The signing phase produces signatures that are interchangeable from signatures produced with a non-threshold implementation with regard to the NIST-standardized EdDSA verification algorithm [FIPS-186-5].

- **Key Generation.** All parties sample  $t$  random shares and set  $n - t$  more shares to be the identity element. Using a commit-open protocol, they exchange discrete-logarithm commitments to these shares. After exchanging commitments, each party privately sends to every other party the secret information they need to verify the commitments, and compute and broadcast public key shares. Each party can then output the shared public key and their respective share of the private key.
- **Signing.**
  1. Each party begins by sampling a random share of the private signature nonce and computes the corresponding EC point. They then broadcast a commitment to this point.
  2. After receiving all the commitments, each party reveals their random EC point to every other party.
  3. After receiving the EC point opening from every other party, each party can sum the random EC points to obtain the public signature nonce and compute their share of the signature string.

**Output:** After broadcasting these signature shares, each party can reconstruct the signature and verify it against the joint public key. If it passes verification, each party outputs it.

This signature phase can be augmented with some operations or messages which enable the *identification* of malicious parties in case the reconstructed signature fails to verify.

**System model and security.** The protocols will be specified for a theoretically unbounded number  $n$  of parties, and the key generation protocol will accept a threshold value  $t$  ranging from 1 to  $n$  inclusive. The experimental evaluation will test performance for values of  $n$  ranging from 2 to several dozens.

The protocols use an *authenticated and synchronous broadcast channel*, modelled with a UC functionality. They also use a private point-to-point channel between each pair of parties.

The analysis of the protocol is conducted in the *strict global random oracle model*, and no other setup assumption (trusted or ideal) is used.

The protocols provide security with abort against  $t$  active and adaptive corruptions; proven in the UC model as indistinguishability from an *ideal threshold signature functionality* [CGGMP20, Figure 15]. The security theorem of the protocol only assumes the unforgeability of non-threshold Schnorr signatures.

An adversary against the protocol can influence the signatures it outputs by withholding their final broadcast, examining the value of the produced signature after receiving all other parties' shares, and deciding to abort or release their broadcast based on the observed signature. However, no adversary can influence the *sampling* of the signature randomness since that step is protected by a commit-open mechanism.

### 3. Open-Source Implementation

The core code is written in C and C++17 and is currently divided into two main modules: (i) a “cosigner” module which contains the functions required to run the Schnorr protocol as a signing party, and (ii) a cryptography module with functions implementing the secondary crypto-systems, additional gadgets, and required cryptographic and arithmetic operations. The submission package will include scripts to build the code on the target reference platform and run benchmarks of the various components for varying threshold configurations and parameter choices.

The code does not bundle any dependencies. It requires the following *external* dependencies:

1. libssl from OpenSSL.
2. libuuid (for testing).
3. libsecp256k1 (for testing).

There already is a public repository [Fir25] with an implementation of the CMP protocol which will be restructured to match the organisation of the submission. This code currently does not implement the network model, but the submission will implement a suitable reliable broadcast over peer-to-peer authenticated TLS channels.

In addition to testing correctness in the all-honest scenario, the current test suite also tests the behaviour of the code in case of maliciously constructed messages or of delivery delays caused by an unreliable network within the boundaries of the reliable broadcast assumption.

### 4. Experimental Performance Evaluation

The current code package's performance is reported in [Table 1](#).

Configuration	Phase	128-bit security (ms)
2-out-of-2	KeyGen	430.3
	Sign	1.8
3-out-of-3	KeyGen	758.7
	Sign	2.0

Table 1: Computation time per party of the core computation for the distributed Schnorr protocol over the `secp256k1` elliptic curve, not including computation required for the networking model. Measurements obtained on Intel® Core™ i7-1365U CPU.

The submission will benchmark larger values of the number  $n$  of parties and different configurations for the threshold  $t$ . It will also benchmark the protocol for signatures computed over the Edwards255 curve.

**Platform.** The baseline platform is expected to provide sufficient resources for realistic benchmarking, and we do not anticipate particular challenges.

## 5. Licensing, Patent Claims, and Funding

The *core code* is currently open-sourced under the GPL-3.0 licence [Fre07]. For the *external dependencies*: `libuuid` is available under the LGPL-2.0 licence [Fre91], `libssl` is available under the Apache-2.0 licence [Apa07], and `libsecp256k1` is available under the MIT licence [MIT].

The team is not aware of any patents with claims covering the content of the planned submission.

## References

- [Apa07] Apache Software Foundation. *The Apache License, Version 2.0*. <https://www.apache.org/licenses/LICENSE-2.0>. 2007.
- [CGGMP20] Ran Canetti, Rosario Gennaro, Steven Goldfeder, Nikolaos Makriyannis, and Udi Peled. “UC Non-Interactive, Proactive, Threshold ECDSA with Identifiable Aborts”. In: *ACM CCS 2020*. Ed. by Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna. ACM Press, November 2020, pp. 1769–1787. DOI: [10.1145/3372297.3423367](https://doi.org/10.1145/3372297.3423367). Also at [ia.cr/2021/060](https://ia.cr/2021/060) IACR ePrint version dates October 21, 2024.
- [Fir25] Fireblocks Ltd. *Fireblocks-MPC*. <https://github.com/fireblocks/mpc-lib/>. 2025.
- [Fre07] Free Software Foundation. *The GNU General Public License version 3.0*. <https://www.gnu.org/licenses/gpl-3.0.en.html>. 2007.
- [Fre91] Free Software Foundation. *The GNU Library General Public License version 2.0*. <https://opensource.org/licenses/lgpl-2-0>. 1991.
- [Mak22] Nikolaos Makriyannis. *On the Classic Protocol for MPC Schnorr Signatures*. Cryptology ePrint Archive, Report 2022/1332. 2022. URL: <https://eprint.iacr.org/2022/1332>.
- [MIT] MIT. *The MIT License*. <https://opensource.org/licenses/mit>.
- [MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. “Accountable-Subgroup Multisignatures: Extended Abstract”. In: *ACM CCS 2001*. Ed. by Michael K. Reiter and Pierangela Samarati. ACM Press, November 2001, pp. 245–254. DOI: [10.1145/501983.502017](https://doi.org/10.1145/501983.502017).
- [NKDM03] Antonio Nicolosi, Maxwell N. Krohn, Yevgeniy Dodis, and David Mazières. “Proactive Two-Party Signatures for User Authentication”. In: *NDSS 2003*. The Internet Society, February 2003.
- [SS01] Douglas R. Stinson and Reto Strohli. “Provably Secure Distributed Schnorr Signatures and a  $(t, n)$  Threshold Scheme for Implicit Certificates”. In: *ACISP 01*. Ed. by Vijay Varadharajan and Yi Mu. Vol. 2119. LNCS. Springer, Berlin, Heidelberg, July 2001, pp. 417–434. DOI: [10.1007/3-540-47719-5\\_33](https://doi.org/10.1007/3-540-47719-5_33).
- [FIPS-186-5] National Institute of Standards and Technology. *Digital Signature Standard (DSS)*. (U.S. Department of Commerce) Federal Information Processing Standards (FIPS) 186-5. 2023. DOI: [10.6028/NIST.FIPS.186-5](https://doi.org/10.6028/NIST.FIPS.186-5).
- [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).