

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

Title: Stoats: Short Two-round vOle-bAsed Threshold Signatures

Subtitle: Two-Party Signing from Pseudorandom Correlated Functions for Vector OLE

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

Team name: The Stoats

Team members: Lennart Braun, Geoffroy Couteau, Kelsey Melissaris, Mahshid Riahinia, Peter Rindal, Elahe Sadeghi

Abstract: This preview writeup presents our plan to submit Stoats, a family of two-round two-party threshold signature crypto-systems built from pseudorandom correlation functions (PCFs). The principal crypto-systems computes ECDSA (Elliptic Curve Digital Signature Algorithm) or Schnorr signatures in two rounds with about a hundred bytes of communication and a runtime in the range of one millisecond. The crypto-systems from the Stoats family satisfy malicious security in the universal composability framework and include variants with stateless and deterministic signing, i.e., nonces are derived pseudorandomly without keeping state across signing sessions. Signing is enabled by a one-time interactive setup that distributes keys for a PCF generating vector oblivious linear evaluation (VOLE) correlations; the PCF is based on the well-studied sparse learning-parity-with-noise (LPN) assumption in the random oracle model. The underlying VOLE-PCF and its maliciously secure distributed key generation are also specified as a standalone gadget and submitted as a crypto-system.

Proposed crypto-systems:

- (I) Stoats-rECDSA: Two-Round Randomized Two-Party ECDSA Signing (Categories N1.2 and N4.1)
- (II) Stoats-dECDSA: Two-Round Deterministic Two-Party ECDSA Signing (Categories N1.2 and N4.1)
- (III) Stoats-Schnorr: Two-Round Deterministic Two-Party Schnorr Signing (Categories S1 and S4)
- (IV) VOLE-PCF: Pseudorandom Correlation Function for Vector-OLE from Sparse LPN (Categories S7)

Keywords: Threshold Cryptography; NIST Threshold Call; ECDSA Signatures; Schnorr Signatures; Deterministic Signing; Pseudorandom Correlation Functions; Sparse LPN

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: Lennart Braun^{i1,a1}, Geoffroy Couteau^{i2,a1}, Kelsey Melissaris^{i3,a1}, Mahshid Riahinia^{i4,a2}, Peter Rindal^{i5,a3}, Elahe Sadeghi^{i6,a4}

Open Researcher and Contributor Identifiers (ORCID):

i1 (0000-0001-9164-305X); i2 (0000-0002-6645-0106); i3 (0000-0001-6840-9863); i4 (0009-0006-2508-7925); i5 (0000-0003-2402-7411); i6 (0000-0002-5224-2613)

Affiliations:

^{a1} Universite Paris Cite, CNRS, IRIF, F-75013, Paris, France

^{a2} DIENS, École normale supérieure, CNRS, Paris, France

^{a3} Category Labs, New York City, United States of America

^{a4} University of Texas at Austin, Austin, United States of America

Main contacts:

- **Team mailing list:** team-stoats@listes.irif.fr
- **Primary technical contact person:** Geoffroy Couteau, couteau@irif.fr
- **Secondary contact person:** 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

Planned Package Submission. The planned package submission will specify a family of UC-secure two-round two-party distributed signature crypto-systems for the ECDSA and Schnorr signature schemes. The schemes are based on the two-party signing protocol recently introduced by a subset of the team's members in [BCMRS26], and instantiate the underlying gadget (a VOLE-PCF) using the approach introduced by a subset of the team's members in [BCMRS25]. This family of crypto-systems fits within categories N1.2 (ECDSA signing) and N4.1 (ECC KeyGen), as well as S1 (Schnorr signing) and S4 (Schnorr key generation).

Secondary Crypto-System. The planned package submission will also include the specification of an additional crypto-system, the core component of our schemes: a pseudorandom correlation function for the vector-OLE correlation (VOLE-PCF) from the sparse LPN assumption (category S7). Since this crypto-system is beneficial in other settings (such as two-party BBS/BBS+ signatures), it makes sense to include it in the Threshold Call.

2. Specification

2.1. Design Principle

We obtain Stoats by following the paradigm introduced in [BCMRS26], which builds 2-party ECDSA and Schnorr signing protocols using VOLE-PCFs. We instantiate this framework by plugging in the sparse-LPN-based VOLE-PCF of [BCMRS25] and designing a thoroughly optimized, fully maliciously secure distributed setup. Security of our distributed setup relies on a variant of the sparse LPN assumption, which we show to be equivalent to the standard sparse LPN assumption. Additionally, we introduce then apply generic techniques to achieve stateless signing while preserving efficiency. We complement our results with concrete performance evaluations and an open-source implementation.

2.2. Model

Security. For distributed signing, we consider a pair of signers, P_A and P_B . We analyze both trusted and distributed setups. We consider both static and adaptive malicious PPT adversaries.

Our distributed signatures are proven secure in the Universal Composability (UC) framework which guarantees security under arbitrary composition within an adversarial environment. We prove a strong variant of simulation-based security, i.e., UC-realization of a signing functionality, which further guarantees that the output signatures are distributed indistinguishably from their single-signer counterparts. Our signing functionalities model *stateless* signing, *security with abort*, and *deterministic* signing when relevant. We model asymmetric signing, where only P_A learns the signature, and note that our protocols straightforwardly generalize to the symmetric setting.

For the two-party VOLE-PCF primitive, the key generation algorithm can be run in interaction with an adversary that corrupts one of the parties (P_A and P_B), and our PCF is secure in the presence of any PPT adversary with limited influence over key generation.

Communication Channels. In all protocols, all parties, including P_A and P_B , and, in the trusted setup variant, the trusted dealer, communicate over an asynchronous authenticated channel with adversarial delay. For security against an adaptive adversary this pairwise channel should be non-committing, which is achievable from, e.g., the one-time-pad in the random oracle model.

Security Assumptions. All of our protocols are proven secure in the random oracle model.

Distributed Setup: The security of the distributed setup further relies on: (1) the sparse LPN with static leakage assumption, which we prove equivalent to the standard sparse LPN assumption ([Ale03]) for certain parameters, and (2) a maliciously-secure oblivious transfer protocol, instantiated by generating 128 bits of base OTs [MRR21] and expanding with SoftSpokenOT [Roy22].

Further Assumptions for Each Protocol:

1. VOLE-PCF (2.3.1): sparse LPN ([Ale03]).
2. Stoats-Schnorr (2.3.2), Deterministic 2-Party Schnorr: VOLE-PCF.
3. Stoats-rECDSA (2.3.3), Non-Deterministic 2-Party ECDSA: (1) VOLE-PCF, (2) knowledge of exponent (KEA, [Dam92]), and (3) existential unforgeability (EUF-CMA, [GMR88]) of ECDSA.
4. Stoats-dECDSA (2.3.4), Deterministic 2-Party ECDSA: (1) VOLE-PCF, (2) KEA, and (3) EUF-CMA of ECDSA.

2.3. Techniques

Notation. Fix an additive group G_S of order q_S with generator G_S , and a function $H_S: G_S^2 \times \{0, 1\}^* \rightarrow \mathbb{F}_{q_S}$. Fix an additive group G_E of order q_E with generator G_E , and a function $H_E: \{0, 1\}^* \rightarrow \mathbb{F}_{q_E}$. Let $P.x$ be projection to the x -coordinate. Parties P_A and P_B are assumed access to random oracle $H_{\text{com}}: G_S^2 \rightarrow \mathbb{F}_{q_S}$ for Schnorr, or $H_{\text{com}}: G_E^4 \rightarrow \mathbb{F}_{q_E}^2$ for ECDSA.

2.3.1. Pseudorandom Correlation Function for VOLE

Our signing protocols makes fundamental use of *pseudorandom correlation functions* (PCFs, [BCGIKS20]) for the vector-OLE (VOLE) correlation over a large prime-order field $\mathbb{F}_p$ (VOLE-PCFs). A VOLE-PCF over $\mathbb{F}_p$ enables parties P_0 and P_1 to on-the-fly generate VOLE-correlated tuples for random $\Delta \in \mathbb{F}_p$ (held by P_1) from short correlated keys: With each evaluation, P_0 obtains $u, v \in \mathbb{F}_p$ and P_1 obtains $w \in \mathbb{F}_p$ such that $w = \Delta \cdot u + v$. A VOLE-PCF has the following syntax:

- $\text{PCF.KeyGen}(\Delta \in \mathbb{F}_p) \rightarrow (k_0^{\text{PCF}}, k_1^{\text{PCF}})$, where k_1^{PCF} contains Δ .
- $\text{PCF.Eval}(\sigma \in \{0, 1\}, k_\sigma^{\text{PCF}}, x \in \mathcal{X}) \rightarrow y_\sigma$ for $y_0 = (u_x, v_x)$ and $y_1 = w_x$.

A VOLE-PCF is *correct* if evaluations $\text{Eval}(0, k_0^{\text{PCF}}, x)$ and $\text{Eval}(1, k_1^{\text{PCF}}, x)$ are correlated according to Δ , *pseudorandom* if no adversary given Δ can distinguish adaptively-chosen evaluations from uniform correlated samples, *secure against* P_0 if k_0^{PCF} leaks no information about Δ , and *secure against* P_1 if no adversary given k_1^{PCF} can distinguish $\text{Eval}(0, k_0^{\text{PCF}}, x)$ from $(u_x, \text{Eval}(1, k_1^{\text{PCF}}, x) - \Delta \cdot u_x)$ for adaptively chosen x and uniform u_x .

We instantiate the VOLE-PCF with the (stateful) construction of [BCMRS25] based on the sparse LPN assumption. The PCF keys include a short seed VOLE correlation with uniform $\mathbf{u}^{(0)} \in \mathbb{F}_p^n$. During evaluation, this seed gets expanded L times by a factor s by applying LPN with an expansion factor s ($\mathbf{u}^{(\ell-1)} \mapsto \mathbf{u}^{(\ell)} := \mathbf{A}^{(\ell)} \cdot \mathbf{u}^{(\ell-1)} + \mathbf{e}^{(\ell)}$) into a VOLE correlation with pseudorandom $\mathbf{u}^{(L)}$ of length $m := n \cdot s^L$. The matrices $\mathbf{A}^{(\ell)}$ are sparse, of dimensions $(s^\ell \cdot n) \times (s^{\ell-1} \cdot n)$, and compressed using a random oracle. The vectors $\mathbf{e}^{(\ell)}$ are sparse, regular error vectors, and compressed via distributed point functions (DPFs). This allows the efficient computation of single entries $u_x^{(L)}$ for $x \in [m]$ yielding a PCF with domain $[m]$.

We use the standard, GGM-based construction for known-index DPFs [BCGIKRS19], and include the keys in $(k_0^{\text{PCF}}, k_1^{\text{PCF}})$. We will specify both the key generation by a trusted dealer and with a distributed key generation protocol. For the latter, we rely on a variant of the single-point VOLE protocol from Wolverine [WYKW21] to generate the DPF keys.

In our signing protocols, we have to ensure that an index x is never reused for two different messages. Since the PCF domain $[m]$ is too small to prevent collisions when hashing into it, the parties need to maintain a state to iterate over the domain. Under the non-standard *filtered* sparse LPN assumption, the VOLE-PCF of [BCMRS25] also supports larger domains $\mathcal{X}$, when the rows of $\mathbf{A}^{(L)}$ are derived by hashing $x \in \mathcal{X}$. To obtain stateless signing protocols without new assumptions, we introduce a novel transformation that generically turns a PCF with domain $\mathcal{X} = [m]$ into a PCF with domain $\mathcal{X}' = \{0, 1\}^*$, while preserving security for up to $m' \lesssim m$ evaluation queries.

2.3.2. Stoats-Schnorr: Deterministic Two-Party Schnorr Signing

Setup. Keys $(k_0^A, k_1^A) \leftarrow \text{PCF.KeyGen}(\Delta_A)$ and $(k_0^B, k_1^B) \leftarrow \text{PCF.KeyGen}(\Delta_B)$ for VOLE-PCFs over $\mathbb{F}_{q_S}$, and $(\text{sk}_A, \text{sk}_B)$ such that $\text{pk} = (\text{sk}_A + \text{sk}_B) \cdot G_S$ are generated.

P_A receives $(\text{pk}, \text{sk}_A, \Delta_A, k_1^A, k_0^B)$. P_B receives $(\text{pk}, \text{sk}_B, \Delta_B, k_0^A, k_1^B)$.

Signing. For a message $m \in \{0, 1\}^*$:

Round 1: P_A runs $(k_A, v_A) \leftarrow \text{PCF.Eval}(0, k_0^B, m)$ and sends $R_A \leftarrow k_A \cdot G_S$ to P_B .

Round 2: P_B runs $w_A \leftarrow \text{PCF.Eval}(1, k_1^B, m)$ and $(k_B, v_B) \leftarrow \text{PCF.Eval}(0, k_0^A, m)$, computes $R_B := k_B \cdot G_S$, $V'_A := w_A \cdot G_S - \Delta_B \cdot R_A$, $V_B := v_B \cdot G_S$, challenge $e \leftarrow H_S(\text{pk}, R_A + R_B, m)$, and message $s_B := (k_B + e \cdot \text{sk}_B) \oplus H_{\text{com}}(V'_A, V_B)$, then sends (R_B, s_B) to P_A .

Output: P_A runs $w_B \leftarrow \text{PCF.Eval}(1, k_0^B, m)$, defines $V_A := v_A \cdot G_S$, $V'_B := w_B \cdot G_S - \Delta_A \cdot R_B$, recovers $\sigma_B := s_B \oplus H_{\text{com}}(V_A, V'_B)$, and aborts if $\sigma_B \neq R_B + e \cdot \text{pk}_B$. Otherwise, P_A computes $\sigma_A := k_A + e \cdot \text{sk}_A$ and outputs $\sigma := \sigma_A + \sigma_B$.

2.3.3. Stoats-rECDSA: Randomized Two-Party ECDSA Signing

Setup. Keys $(k_0^A, k_1^A) \leftarrow \text{PCF.KeyGen}(\text{sk}_A)$ for a VOLE-PCF over $\mathbb{F}_{q_E}$, and $(\text{sk}_A, \text{sk}_B)$ such that $\text{pk} = (\text{sk}_A + \text{sk}_B) \cdot G_E$ are generated.

P_A receives $(\text{pk}, \text{sk}_A, k_1^A)$. P_B receives $(\text{pk}, \text{sk}_B, k_0^A)$.

Signing. For a message $m \in \{0, 1\}^*$:

Round 1: P_A samples $k_A, v_A \leftarrow \mathbb{F}_{q_E}$ and sends $R_A \leftarrow k_A \cdot G_E$ to P_B .

Round 2: P_B runs $(k_B, v_B) \leftarrow \text{PCF.Eval}(0, k_0^A, (\text{pk}, R_A, m))$, sets $R_B := k_B \cdot G_E$, $r := (k_B \cdot R_A).x$, runs $(u_C, v_C) \leftarrow \text{PCF.Eval}(0, k_0^A, (\text{pk}, R_A, m))$, sets $u := k_B^{-1} - u_C$, $v := k_B^{-1} \cdot (H_E(m) + r \cdot \text{sk}_B) - r \cdot v_C$, $V_B := v_B \cdot G_E$, KEA proofs $\pi_R := k_B \cdot \text{pk}_A$ and $\pi_V := v_B \cdot \text{pk}_A$, and message $s_B := (u, v) \oplus H_{\text{com}}(V_B, \pi_R, \pi_V)$, then sends (R_B, s_B) to P_A .

Output: P_A runs $w_B \leftarrow \text{PCF.Eval}(1, k_1^A, (\text{pk}, R_A, m))$ and $w_C \leftarrow \text{PCF.Eval}(1, k_1^A, (\text{pk}, R_A, m))$, sets $V'_B := w_B \cdot G_E - \Delta_A \cdot R_B$, $\pi'_R := \text{sk}_A \cdot R_B$, $\pi'_V := \text{sk}_A \cdot V'_B$, and recovers $(u', v') := s_B \oplus H_{\text{com}}(V'_B, \pi'_R, \pi'_V)$, sets $r := (k_A \cdot R_B).x$, $\sigma := k_A^{-1} \cdot (v' + r \cdot (w_C + \text{sk}_A \cdot u'))$, aborts if $r \neq \sigma^{-1} \cdot (H_E(m) \cdot G_E + r \cdot \text{pk}).x$ and otherwise outputs σ .

2.3.4. Stoats-dECDSA: Deterministic Two-Party ECDSA Signing

Setup. Keys $(k_0^A, k_1^A) \leftarrow \text{PCF.KeyGen}(\text{sk}_A)$ and $(k_0^B, k_1^B) \leftarrow \text{PCF.KeyGen}(\Delta_B)$ for VOLE-PCFs over $\mathbb{F}_{q_E}$, and $(\text{sk}_A, \text{sk}_B)$ such that $\text{pk} = (\text{sk}_A + \text{sk}_B) \cdot G_E$ are generated.

P_A receives $(\text{pk}, \text{sk}_A, k_1^A, k_0^A)$. P_B receives $(\text{pk}, \text{sk}_B, \Delta_B, k_0^B, k_1^B)$.

Signing. For a message $m \in \{0, 1\}^*$:

Round 1: P_A runs $(k_A, v_A) \leftarrow \text{PCF.Eval}(0, k_0^B, m)$ and sends $R_A \leftarrow k_A \cdot G_E$ to P_B .

Round 2: P_B runs $w_A \leftarrow \text{PCF.Eval}(1, k_1^B, m)$, $(k_B, v_B) \leftarrow \text{PCF.Eval}(0, k_0^A, (0, m))$, and $(u_C, v_C) \leftarrow \text{PCF.Eval}(0, k_0^A, (1, m))$. Then, P_B defines $R_B := k_B \cdot G_E$, $V_B := v_B \cdot G_E$, $V'_A := w_A \cdot G_E - \Delta_B \cdot R_A$, $r := (k_B \cdot R_A).x$, $v := k_B^{-1} \cdot (H_E(m) + r \cdot \text{sk}_B) - r \cdot v_C$, $u := k_B^{-1} - u_C$, KEA proofs $\pi_R := k_B \cdot \text{pk}_A$, $\pi_V := v_A \cdot \text{pk}_A$, and message $s_B := (u, v) \oplus H_{\text{com}}(V'_A, V_B, \pi_R, \pi_V)$, then sends (R_B, s_B) to P_A .

Output: P_A runs $w_B \leftarrow \text{PCF.Eval}(1, k_1^A, (0, m))$ and $w_C \leftarrow \text{PCF.Eval}(1, k_1^A, (1, m))$, defines $V_A := v_A \cdot G_E$, $V'_B := w_B \cdot G_E - \Delta_A \cdot R_B$, $\pi'_R := \text{sk}_A \cdot R_B$, $\pi'_V := \text{sk}_A \cdot V'_B$, and recovers $(u', v') := s_B \oplus H_{\text{com}}(V_A, V'_B, \pi'_R, \pi'_V)$. Then, P_A computes $r := (k_A \cdot R_B).x$ and $\sigma := k_A^{-1} \cdot (v' + r \cdot (w_C + \text{sk}_A \cdot u'))$, aborts if $r \neq \sigma^{-1} \cdot (H_E(m) \cdot G_E + r \cdot \text{pk}).x$, and otherwise outputs σ .

2.4. Organization

The submission begins by introducing the context, background, and preliminary definitions of the objects used in our project, including signing protocols, PCFs, and security assumptions. Next, we review the paradigm of [BCMRS26], which builds two-party signing protocols from VOLE-PCFs (our underlying gadget). We then review the recursive approach of [BCMRS25] which builds a VOLE-PCF recursively in the random oracle model, assuming the hardness of sparse LPN and the existence of distributed multi-point functions. This is followed by a discussion on our design choices for instantiating the VOLE-PCF.

Next, we present our family of cryptosystems: threshold ECDSA and Schnorr. We describe these protocols in two parts: setup and signing. In particular, we explain our approach to distributed setup and our distributed key generation for the underlying VOLE-PCFs. We analyze the security of this phase, which in particular includes the analysis of a new variant of the sparse LPN assumption and its reduction to the standard sparse LPN. Finally, we present our generic techniques for achieving stateless signing. Our submission will conclude with suggestions on parameter choices, optimized for fast distributed setup, fast evaluation, and small key sizes. We provide performance evaluation based on the open-source implementation that will accompany our submission.

3. Open-Source Implementation

We intend to release the implementation as open source under the MIT license. The current code base is a functioning prototype that is not yet ready for public release. The implementation is written in C++20 for 64-bit Windows, macOS, and Linux, using `libOTe`² and `cryptoTools`³ for core cryptographic infrastructure and `coproto`⁴/`boost::asio` for two-party networking. Our code implements the DPF and PCF layers, setup, online evaluation, and two-party signing. The prototype is heavily tested and reproducible, with unit and end-to-end tests, malicious consistency tests, and microbenchmarks. Current measurements use a single machine with intra-process communication; final results will include realistic local- and wide-area network settings.

4. Experimental Performance Evaluation

Our preliminary measurements already reflect a complete prototype for our dECDSA protocol realizing malicious two-party P-256 signing. In the present local benchmark environment, full key generation is about 1.5 seconds for approximately 2^{23} signatures. Once generated, a single dECDSA signing session via the local socket API is about 1.10 ms, with only $240 \mu\text{s}$ of that being the PCF computation. The benchmarked configuration uses a PCF domain of size 2^{25} , which supports approximately 2^{23} signatures. The number of supported signing sessions scales linearly with the setup time. The setup can be rerun under the same long-term signing key to refresh this material. We will investigate the tradeoff between larger PCF domain sizes and more frequent key refreshes in the full submission.

The protocol description above presents the VOLE-PCF abstractly: a short seed VOLE is expanded through sparse-LPN levels, and signing evaluates the resulting PCF at message-derived inputs. Our prototype realizes the protocol via various optimizations. During setup, the first phase is Base OT generation, which produces base OTs used to generate the “Base VOLE” correlations used on the first level of recursion. This initial level is then expanded several times to obtain the “cached level,” which is cached for later use in the signing phase. The setup then performs “full level expansion,” where all the remaining levels are generated and checked via a random linear combination. The fully expanded levels are computed in a streaming manner and discarded

²<https://github.com/osu-crypto/libOTe>

³<https://github.com/ladnir/cryptotools>

⁴<https://github.com/ladnir/coproto>

Phase	Component	Preliminary time
Setup	Base OT generation	15.85 ms
	Base VOLE generation	23.19 ms
	Cached level expansion	114.20 ms
	Full level expansion	1319.67 ms
	Key authentication	1.06 ms
Signing	PCF evaluations	240.37 μ s
	Full signing	1.10 ms

after the checks. Finally, key authentication is a short setup-time ECDSA check that binds the generated PCF material to the public-key share used by the signing protocol.

The signing rows measure the online phase after setup has completed. The “PCF evaluations” row is the aggregate PCF evaluation work for one complete dECDSA signing session in the optimized two-PCF mode. The “full signing” row includes these PCF evaluations together with the ordinary P-256 scalar and group operations, message handling, and local socket/coroutine overhead.

The measurements were taken on a laptop-class 64-bit x86 platform with 32 GB of RAM, using single-machine intra-process communication through the `coproto/boost::asio` stack. After setup, key authentication exchanges three affine P-256 points in total (192 bytes before protocol framing), and each signing instance exchanges one affine point from P_A , followed by one affine point and two scalars from P_B (192 bytes before protocol framing). For setup, the present implementation reports an estimate of tree-correction traffic rather than final network-byte measurements: in the benchmarked 2^{25} PCF-domain configuration, this is about 256 KiB per PCF for the setup-only levels below the cached level and about 2.9 MiB per PCF for the online levels above it, plus smaller base-OT, base-VOLE, and Fiat–Shamir check messages. The final submission will fully specify this optimized setup layout, its parameter choices, and its concrete communication accounting, and will include measurements in realistic LAN and WAN settings.

5. Licensing, Patent Claims, and Funding

We intend to release the core implementation under the MIT license, which is Open Source Initiative approved and therefore compatible with the licensing requirement in Section 7.4 of the Threshold Call [NIST-IR8214C]. Our prototype depends on widely used open-source libraries, and is expected to remain within standard permissive or otherwise acceptable open-source licenses.

Authors 1, 2, and 3 acknowledge the support of the French Agence Nationale de la Recherche (ANR) under the France 2030 ANR Project ANR-22-PECY-0003 SecureCompute. This work is supported by ERC grant OBELiSC (101115790) and by an Amazon Research Award.

6. Comparison with Other Submissions

ECDSA signing (category N1.2) has preview writeups from TECLA [BIC26a] and THE CLASH [BIC26b] (class groups), Fireblocks's CGGMP [AABCDGM26a] and BAM [AABCDGM26c; ABCGJM24] (Paillier), and Trout++ [DNP26] (class groups); none to date is based on oblivious transfer (e.g., the DKLS line [DKLS24]). Schnorr/EdDSA signing (categories N1.1 and S1) has previews from Fireblocks's Classic Schnorr [AABCDGM26b], FROST [FRO26; KG20], and Gargos [BDLR26]. Stoats is “best in class” on efficiency and security in exchange for a few clearly identified downsides, making comparisons straightforward.

Efficiency benefits. Stoats signs in a millisecond, with optimal two-message latency and around 100 Bytes of communication per session. No previewed threshold-ECDSA submission comes close: only BAM [AABCDGM26c] and Trout++ [DNP26] sign in two rounds, at around $15\times$ more communication and $100\times\text{--}300\times$ more computation ($10^4\times$ for [DNP26]'s constant-time implementation). TECLA [BIC26a], the closest, still requires $6\times$ more communication and $\sim 70\times$ more computation than Stoats-rECDSA and Stoats-dECDSA (and likely more rounds, unspecified in the preview). Stoats-Schnorr matches FROST [FRO26] in efficiency (slightly lower communication, comparable runtime, two rounds); its main edge is security.

Security benefits. Stoats UC-realizes standard distributed ECDSA and Schnorr functionalities. All other planned ECDSA submissions (N1.2) satisfy only a weaker game-based notion, except Trout++ [DNP26] (which realizes a non-standard functionality: the adversary may tweak the nonce) and CGGMP [AABCDGM26a] (an abstract signing functionality). For Schnorr (S1), Classic Schnorr [AABCDGM26b] UC-realizes the standard functionality but uses three rounds.

Furthermore, our protocols assume only standard ECDSA/Schnorr EUF-CMA security, whereas every other round-optimal scheme needs a nonstandard signature assumption: doubly-enhanced ECDSA unforgeability for Trout++ and BAM, algebraic one-more discrete logarithm for FROST. Stoats-dECDSA and Stoats-Schnorr additionally guarantee deterministic stateless signing. Randomness is a well-known source of critical failures — good entropy or well-seeded PRNGs are notoriously hard to obtain, e.g., on cloud infrastructures [GKMN21; KOR23] — so these two schemes resist realistic attacks such as state resets and randomness failures, and mirror standardized signatures like EdDSA, whose nonces are deterministic. To our knowledge, no other previewed submission offers this. Stoats-rECDSA is stateless (resilient to state resets) but not deterministic.

Downside. Stoats is restricted to the two-party setting. This covers many common real-world uses—cryptocurrency custody, embedded wallets, virtualizing hardware security modules—but rules out settings with more signers. In the two-party setting, identifiable aborts hold trivially and adaptive security is achieved by guessing with a 1-bit loss.

Stoats also trades its fast signing for a heavy setup. Our experiments (Section 4) distribute the Stoats-rECDSA setup for $B = 2^{25}$ signing sessions in around 1.5 seconds, and 3 seconds for Stoats-dECDSA and Stoats-Schnorr (with $B = 2^{24}$, due to our stateful-to-stateless conversion); setup time grows linearly with B (e.g., 12s for $B = 2^{28}$ with Stoats-rECDSA). After B sessions the setup must be refreshed (without re-sampling the signing keys), and it produces large keys (around 2.5MB per party). Finally, Stoats relies on the sparse LPN assumption of Alekhovich (2003) [Ale03], which has received less attention than group-based and factoring-related assumptions.

References

[TestTag] Test Author. *Test Title in a @misc Style Bib Entry*. Original howpublished content. Original note content. Example ad-hoc note using \bibfieldadd. August 2025. doi: [NN.NNNN/NNNN.NNNN](https://doi.org/10.1111/NNNN.NNNN). url: <https://www.example.com/testURL-in-alternative-to-a-hyperlinked-doi>. Original addendum content. Ad-hoc addendum using \bibfieldadd. Ad-hoc “final note” using \bibfinalnote.

[AABCDGM26a] Michael Adjedj, Tomer Ashur, Amit Singh Bhati, Geoffroy Couteau, Cyprien Delpech de Saint Guilhem, Michael Gutkin, and Nikos Makriyannis. *Distributed ECDSA Signatures: Fireblocks’s CGMP Protocol*. Preview writeup, NIST First Call for Multi-Party Threshold Schemes. 2026. url: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Fireblocks-3MI-c-PW01.pdf>.

[AABCDGM26b] Michael Adjedj, Tomer Ashur, Amit Singh Bhati, Geoffroy Couteau, Cyprien Delpech de Saint Guilhem, Michael Gutkin, and Nikos Makriyannis. *Distributed Schnorr Signatures: Fireblocks’s Classic Schnorr Protocol*. Preview writeup, NIST First Call for Multi-Party Threshold Schemes. 2026. url: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Fireblocks-3MI-a-PW01.pdf>.

[AABCDGM26c] Michael Adjedj, Tomer Ashur, Amit Singh Bhati, Geoffroy Couteau, Cyprien Delpech de Saint Guilhem, Michael Gutkin, and Nikos Makriyannis. *Two-Party ECDSA Signatures: Fireblocks’s BAM Protocol*. Preview writeup (version 1.0), NIST First Call for Multi-Party Threshold Schemes. 2026.

[ABCGJM24] Michael Adjedj, Constantin Blokh, Geoffroy Couteau, Arik Galansky, Antoine Joux, and Nikolaos Makriyannis. *Two-Round 2PC ECDSA at the Cost of 1 OLE*. Cryptology ePrint Archive, Report 2024/1950. 2024. url: <https://eprint.iacr.org/2024/1950>.

[Ale03] Michael Alekhovich. “More on Average Case vs Approximation Complexity”. In: *44th Annual Symposium on Foundations of Computer Science*. Cambridge, MA, USA: IEEE Computer Society Press, October 2003, pp. 298–307. doi: [10.1109/SFCS.2003.1238204](https://doi.org/10.1109/SFCS.2003.1238204).

[BCGIKRS19] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Rindal, and Peter Scholl. “Efficient Two-Round OT Extension and Silent Non-Interactive Secure Computation”. In: *ACM CCS 2019: 26th Conference on Computer and Communications Security*. Ed. by Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz. London, UK: ACM Press, November 2019, pp. 291–308. doi: [10.1145/3319535.3354255](https://doi.org/10.1145/3319535.3354255).

[BCGIKS20] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. “Correlated Pseudorandom Functions from Variable-Density LPN”. In: *61st Annual Symposium on Foundations of Computer Science*. Durham, NC, USA: IEEE Computer Society Press, November 2020, pp. 1069–1080. doi: [10.1109/FOCS46700.2020.00103](https://doi.org/10.1109/FOCS46700.2020.00103).

[BCMRS25] Lennart Braun, Geoffroy Couteau, Kelsey Melissaris, Mahshid Riahinia, and Elahe Sadeghi. “Fast Pseudorandom Correlation Functions from Sparse LPN”. In: *Advances in Cryptology – ASI-*

ACRYPT 2025, Part VII. Ed. by Goichiro Hanaoka and Bo-Yin Yang. Vol. 16251. Lecture Notes in Computer Science. Melbourne, VIC, Australia: Springer, Singapore, Singapore, December 2025, pp. 439–473. doi: [10.1007/978-981-95-5122-4_14](https://doi.org/10.1007/978-981-95-5122-4_14).

[BCMRS26] Lennart Braun, Geoffroy Couteau, Kelsey Melissaris, Mahshid Riahinia, and Elahe Sadeghi. “Succinct Two-Round Two-Party Signing from PCFs”. In: *Advances in Cryptology – CRYPTO 2026*. 2026.

[BDLR26] Renas Bacho, Sourav Das, Julian Loss, and Ling Ren. *Gargos: Adaptively Secure Threshold Schnorr Signatures*. Preview writeup, NIST First Call for Multi-Party Threshold Schemes. 2026. url: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/BDLR-Gargos-PW01.pdf>.

[BIC26a] BICYCCLIST Team. *TECLA: Two-Party ECDSA from Class Groups*. Preview writeup, NIST First Call for Multi-Party Threshold Schemes. 2026. url: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/BICYCCLIST-a-PW01.pdf>.

[BIC26b] BICYCCLIST Team. *THE CLASH: Threshold ECDSA from Class Groups*. Preview writeup, NIST First Call for Multi-Party Threshold Schemes. 2026. url: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/BICYCCLIST-b-PW01.pdf>.

[Dam92] Ivan Damgård. “Towards Practical Public Key Systems Secure Against Chosen Ciphertext Attacks”. In: *Advances in Cryptology – CRYPTO’91*. Ed. by Joan Feigenbaum. Vol. 576. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer Berlin Heidelberg, Germany, August 1992, pp. 445–456. doi: [10.1007/3-540-46766-1_36](https://doi.org/10.1007/3-540-46766-1_36).

[DKLs24] Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. “Threshold ECDSA in Three Rounds”. In: *2024 IEEE Symposium on Security and Privacy*. San Francisco, CA, USA: IEEE Computer Society Press, May 2024, pp. 3053–3071. doi: [10.1109/SP54263.2024.00178](https://doi.org/10.1109/SP54263.2024.00178).

[DNP26] Hila Dahari-Garbian, Ariel Nof, and Luke Parker. *Trout++: Two-Round Threshold ECDSA Composable with ROAST*. Preview writeup, NIST First Call for Multi-Party Threshold Schemes. 2026. url: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Trout-PW02.pdf>.

[FRO26] FROST Team. *FROST: Flexible Round-Optimized Schnorr Threshold Signatures*. Preview writeup, NIST First Call for Multi-Party Threshold Schemes. 2026. url: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/FROST-PW01.pdf>.

[GKMN21] François Garillot, Yashvanth Kondi, Payman Mohassel, and Valeria Nikolaenko. “Threshold Schnorr with Stateless Deterministic Signing from Standard Assumptions”. In: *Advances in Cryptology – CRYPTO 2021, Part I*. Ed. by Tal Malkin and Chris Peikert. Vol. 12825. Lecture Notes in Computer Science. Virtual Event: Springer, Cham, Switzerland, August 2021, pp. 127–156. doi: [10.1007/978-3-030-84242-0_6](https://doi.org/10.1007/978-3-030-84242-0_6).

- [GMR88]** Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. “A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks”. In: *SIAM Journal on Computing* 17.2 (1988), pp. 281–308. doi: [10.1137/0217017](https://doi.org/10.1137/0217017). eprint: <https://doi.org/10.1137/0217017>. url: <https://doi.org/10.1137/0217017>.
- [KG20]** Chelsea Komlo and Ian Goldberg. “FROST: Flexible Round-Optimized Schnorr Threshold Signatures”. In: *SAC 2020: 27th Annual International Workshop on Selected Areas in Cryptography*. Ed. by Orr Dunkelman, Michael J. Jacobson Jr., and Colin O’Flynn. Vol. 12804. Lecture Notes in Computer Science. Halifax, NS, Canada (Virtual Event): Springer, Cham, Switzerland, October 2020, pp. 34–65. doi: [10.1007/978-3-030-81652-0_2](https://doi.org/10.1007/978-3-030-81652-0_2).
- [KOR23]** Yashvanth Kondi, Claudio Orlandi, and Lawrence Roy. “Two-Round Stateless Deterministic Two-Party Schnorr Signatures from Pseudorandom Correlation Functions”. In: *Advances in Cryptology – CRYPTO 2023, Part I*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14081. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Cham, Switzerland, August 2023, pp. 646–677. doi: [10.1007/978-3-031-38557-5_21](https://doi.org/10.1007/978-3-031-38557-5_21).
- [MRR21]** Ian McQuoid, Mike Rosulek, and Lawrence Roy. “Batching Base Oblivious Transfers”. In: *Advances in Cryptology – ASIACRYPT 2021, Part III*. Ed. by Mehdi Tibouchi and Huaxiong Wang. Vol. 13092. Lecture Notes in Computer Science. Singapore: Springer, Cham, Switzerland, December 2021, pp. 281–310. doi: [10.1007/978-3-030-92078-4_10](https://doi.org/10.1007/978-3-030-92078-4_10).
- [Roy22]** Lawrence Roy. “SoftSpokenOT: Quieter OT Extension from Small-Field Silent VOLE in the Minicrypt Model”. In: *Advances in Cryptology – CRYPTO 2022, Part I*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13507. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Cham, Switzerland, August 2022, pp. 657–687. doi: [10.1007/978-3-031-15802-5_23](https://doi.org/10.1007/978-3-031-15802-5_23).
- [WYKW21]** Chenkai Weng, Kang Yang, Jonathan Katz, and Xiao Wang. “Wolverine: Fast, Scalable, and Communication-Efficient Zero-Knowledge Proofs for Boolean and Arithmetic Circuits”. In: *2021 IEEE Symposium on Security and Privacy*. San Francisco, CA, USA: IEEE Computer Society Press, May 2021, pp. 1074–1091. doi: [10.1109/SP40001.2021.00056](https://doi.org/10.1109/SP40001.2021.00056).
- [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).