

Teddy: Threshold EdDSA

NIST MPTC Preview Talk · Category N1 (EdDSA Sign)

Team Octopus: Man Ho Au[1], Robert H. Deng[3], Liming Gao[2], Bowen Jiang[3], Xianhui Lu[2], Tian Qiu[4], Guofeng Tang[3], Ruida Wang[2], **Haiyang Xue**[3], Guomin Yang[3], Chengru Zhang[5]

13 Oct 2026

[1]The Hong Kong Polytechnic University [2]IIE, CAS [3]Singapore Management University
[4]Nanyang Technological University [5]Ethereum Foundation

Presented at NIST TCPT3

Teddy at a Glance

t-out-of-n

any t of n parties sign
up to $t - 1$ corrupted

2 rounds

point-to-point channels
no broadcast

deterministic

nonces fixed by
signer set + message

Ed25519

verification
unmodified

unforgeable

strong notion, static
corruptions, DL + ROM

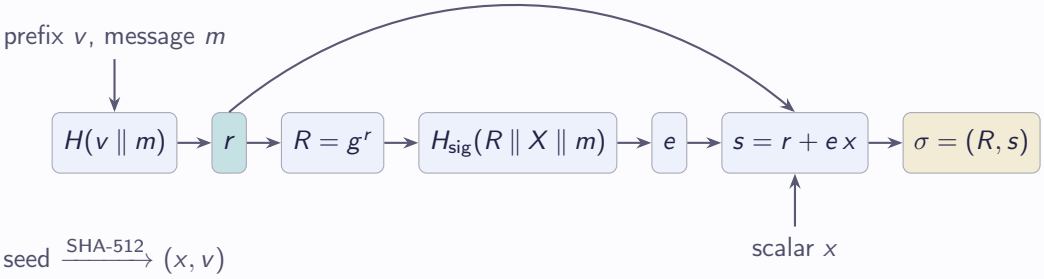
nonce proofs

Poseidon · SHA512 · SHA3
AES: **under investigation**

NIST Threshold Call · Category N1 · EdDSA Sign · Rust reference implementation

Why Threshold Deterministic EdDSA Is Hard

EdDSA: The Nonce Is Derived, Not Sampled



FROST (RFC 9591)

random nonce · stored between rounds

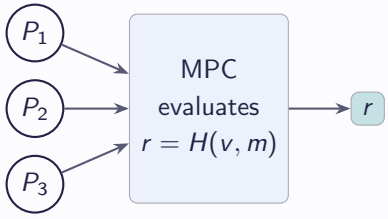
EdDSA

derived nonce · no state

Goal: deterministic + stateless + t-out-of-n + dishonest majority + 2 rounds

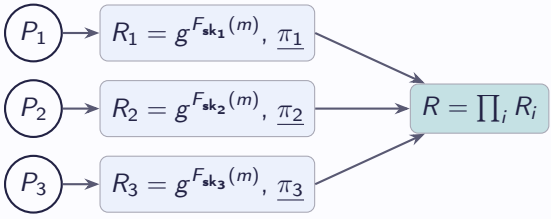
Two Routes to Distributed Determinism

Route 1: joint nonce (MPC)



honest majority, or
3 rounds with all n

Route 2: local nonce + proof



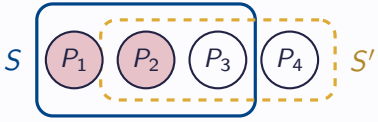
2 rounds, but
only n-out-of-n

Route 1: [BST21, KOR23] Route 2: [NRSW20, GKMN21, FYZ+25, BHLS25]

Teddy takes Route 2. Why is it stuck at n-out-of-n?

Message-Only Nonces Break in t-out-of-n

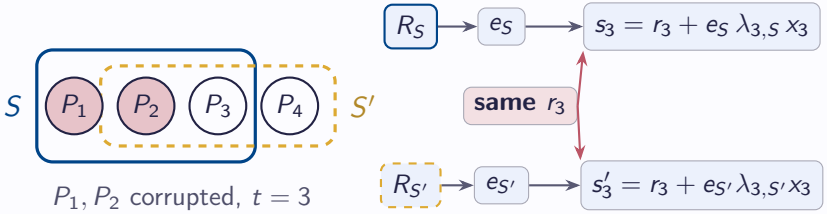
each party: $r_i = F_{\mathsf{sk}_i}(m)$, $s_i = r_i + e\,\lambda_{i,S}\,x_i$ [BHLS25] with Shamir shares



P_1, P_2 corrupted, $t = 3$
honest P_3 in both sets

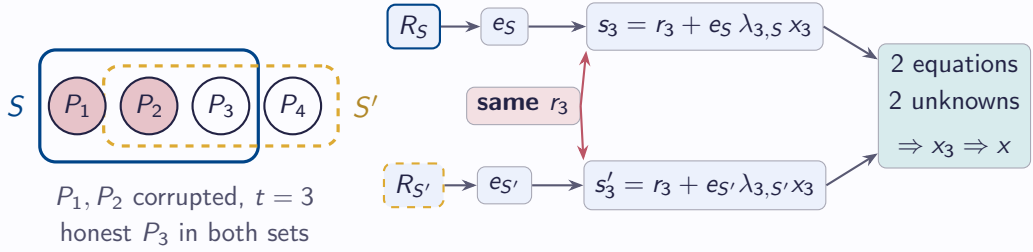
Message-Only Nonces Break in t-out-of-n

each party: $r_i = F_{\mathsf{sk}_i}(m)$, $s_i = r_i + e\,\lambda_{i,S}\,x_i$ [BHLS25] with Shamir shares



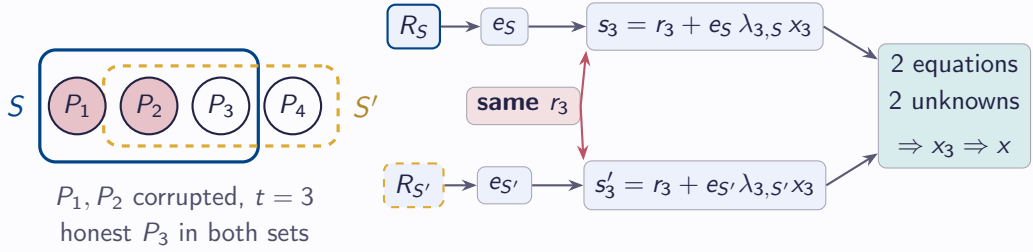
Message-Only Nonces Break in t-out-of-n

each party: $r_i = F_{\mathsf{sk}_i}(m)$, $s_i = r_i + e\,\lambda_{i,S}\,x_i$ [BHLS25] with Shamir shares



Message-Only Nonces Break in t-out-of-n

each party: $r_i = F_{\mathsf{sk}_i}(m)$, $s_i = r_i + e\,\lambda_{i,S}\,x_i$ [BHLS25] with Shamir shares



No deviation needed

Every nonce is honestly computed, so proofs of correct derivation do not help.

The Obvious Repairs Fail

$r_i = F_{\mathsf{sk}_i}(m, R)$
circular:
R needs all r_j ✗

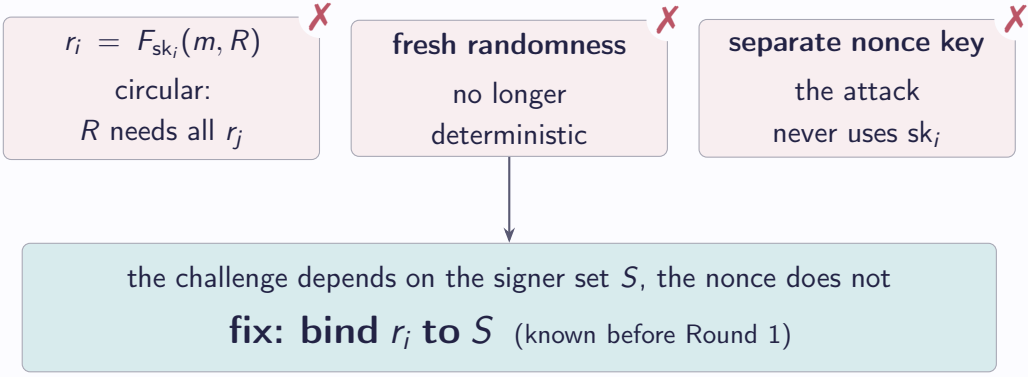
fresh randomness
no longer
deterministic ✗

separate nonce key
the attack
never uses sk_i ✗

the challenge depends on the signer set S, the nonce does not

fix: bind r_i to S (known before Round 1)

The Obvious Repairs Fail

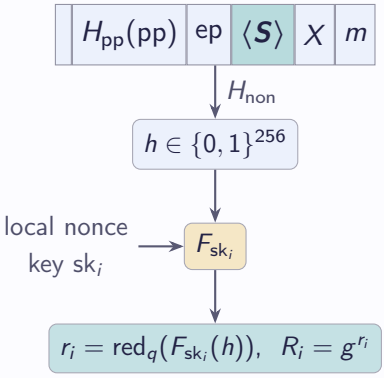


Ed25519-verifiable

but **not the FIPS 186-5/RFC 8032 output**: different signer sets give different signatures.

The Teddy Scheme

Key Idea: Bind the Nonce to the Signer Set



e.g. $n = 5$, $S = \{P_1, P_2, P_4\}$

$\langle S \rangle =$ | 1 | 1 | 0 | 1 | 0 |

set-and-message deterministic

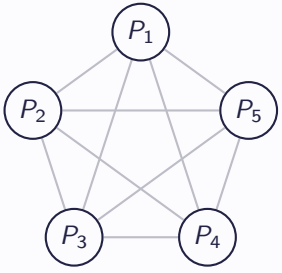
$(S_1, m) \longrightarrow \sigma_1$

$(S_2, m) \longrightarrow \sigma_2$

one signature per (S, m),
each a standard Ed25519 signature

Idea sketched in [BHLS25, §4.4]; also used in [GLRS26].

Setup and Communication Model



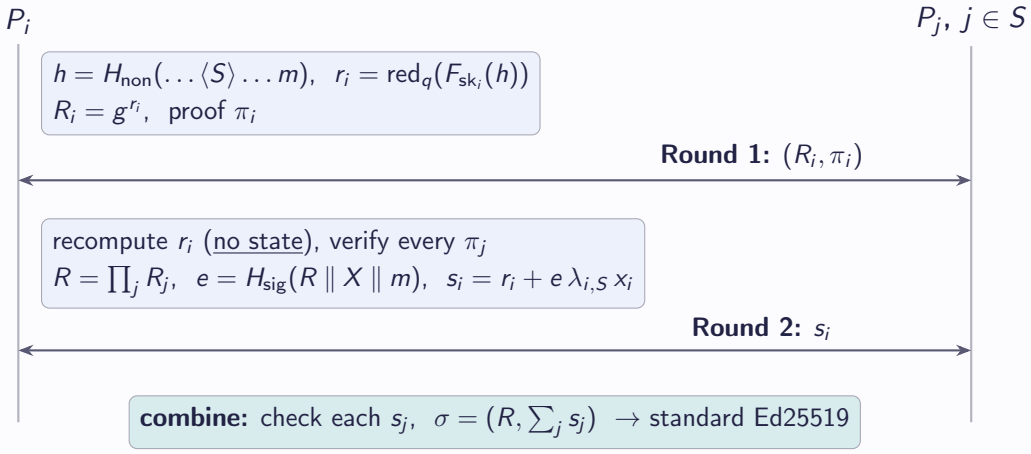
authenticated point-to-point
no broadcast channel

each P_i holds: Shamir share x_i, nonce key sk_i
public: X, $X_i = g^{x_i}$, $\mathsf{pk}_i = g^{\mathsf{sk}_i}$ + PoK

① **once per epoch:** agree on pp
every party signs $H_{\mathsf{pp}}(\mathsf{pp})$

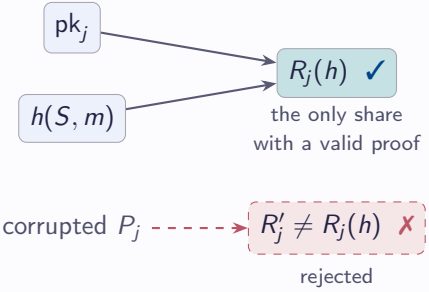
② **before each session:** agree on
the signer set S and the message m

Signing in Two Rounds



Why Two Rounds Suffice, Without Broadcast

each nonce share is pinned by (S, m)



honest or corrupted, no party has a nonce to choose

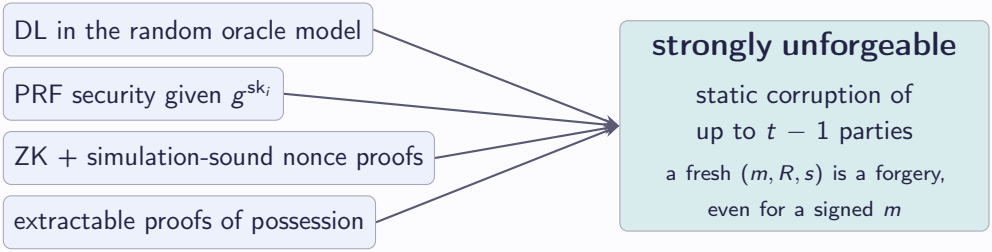
no nonce-commitment round
$\Rightarrow$ no rough-key attack

equivocation in Round 1 $\Rightarrow$ abort

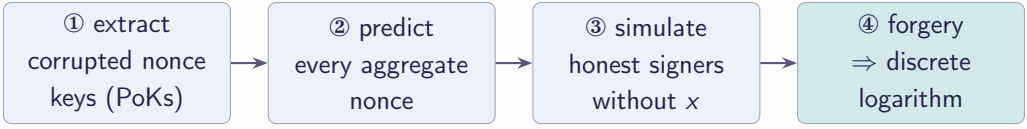
disagreement on S $\Rightarrow$ abort
liveness issue, not a forgery

retry with a new S $\Rightarrow$ fresh nonces

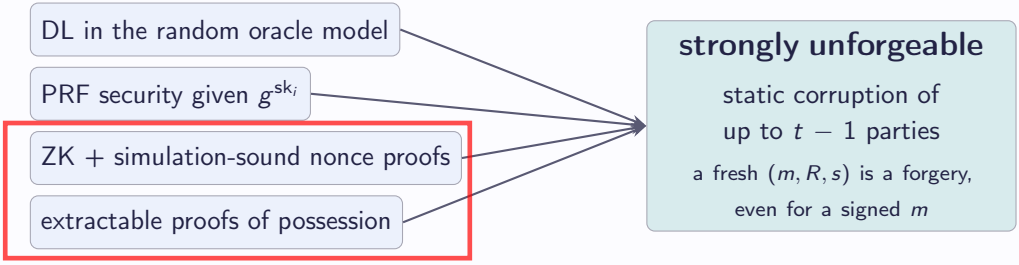
Security, Informally



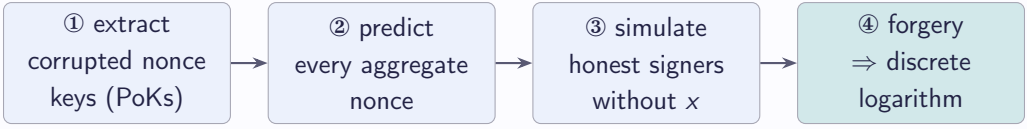
proof idea



Security, Informally



proof idea



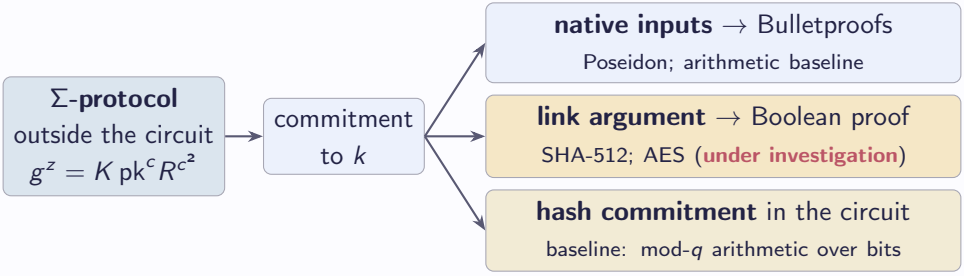
Proving Correct Nonce Derivation

Mixed Statements: Keep Group Operations Outside

$\mathsf{pk} = g^{\mathsf{sk}} \ \wedge \ R = g^r$
group operations (edwards25519)

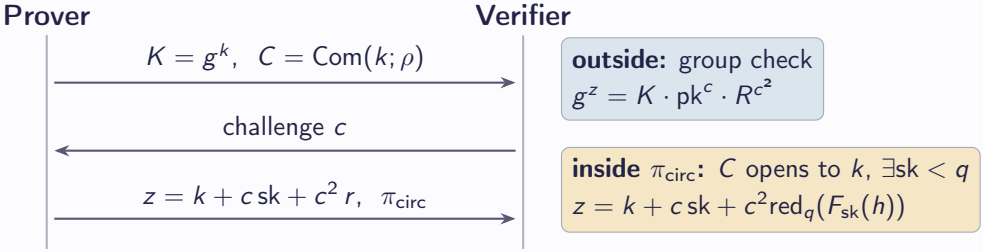
$r = \mathsf{red}_q(F_{\mathsf{sk}}(h))$
PRF circuit

one circuit for both
$\Rightarrow$ foreign arithmetic

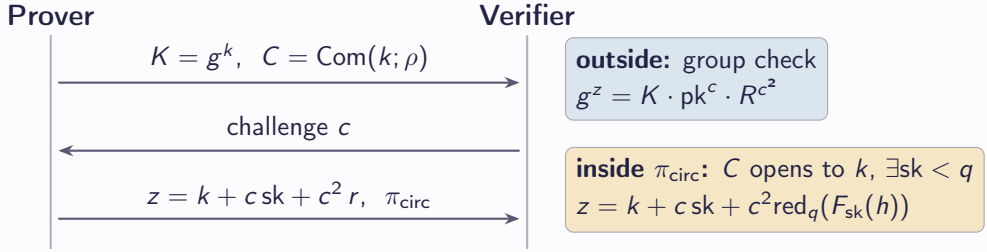


commit-and-prove, following [KMN23, OKMZ25]

One Σ-Protocol for Both Exponents

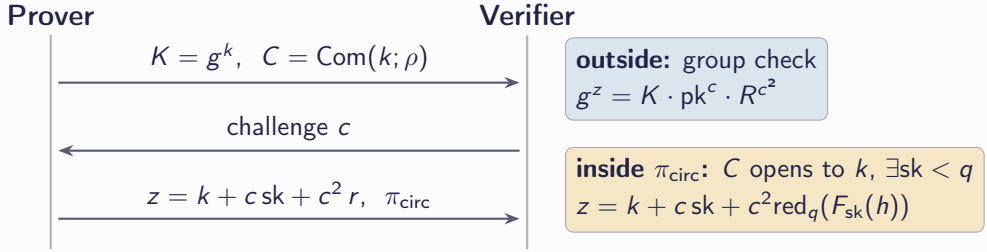


One Σ-Protocol for Both Exponents



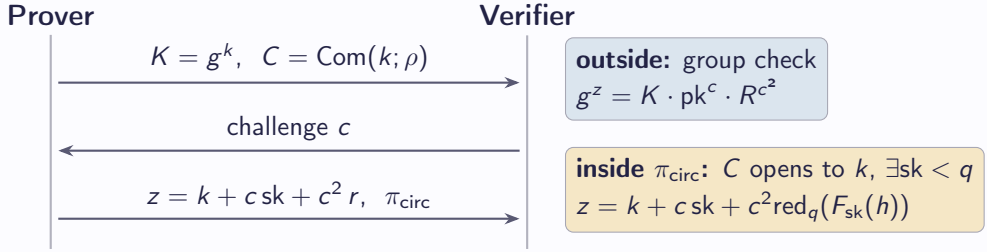
- Teddy[Poseidon]: Com and/or F_{sk} are Poseidon

One Σ-Protocol for Both Exponents



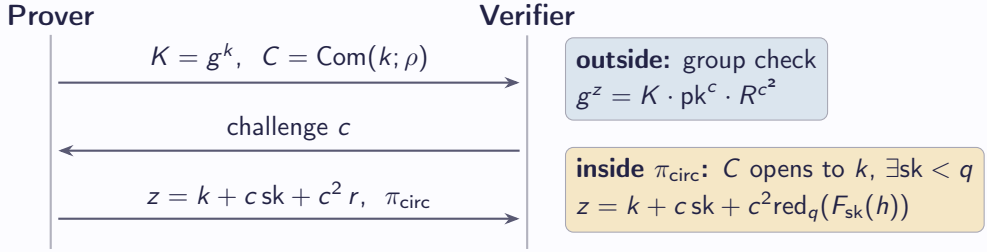
- Teddy[Poseidon]: Com and/or F_{sk} are Poseidon
- Teddy[SHA512]: Com and/or F_{sk} are SHA512

One Σ-Protocol for Both Exponents



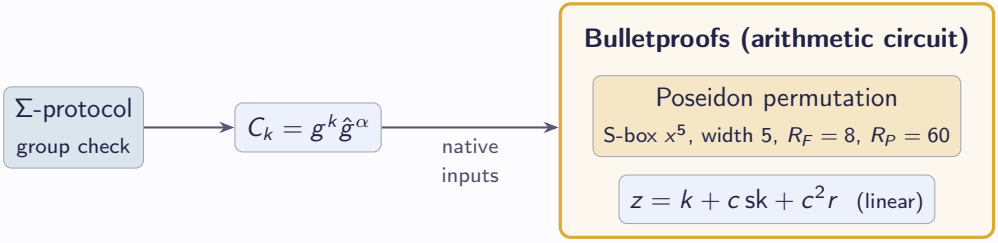
- Teddy[Poseidon]: Com and/or F_{sk} are Poseidon
- Teddy[SHA512]: Com and/or F_{sk} are SHA512
- Teddy[SHA3]: Com and/or F_{sk} are SHAKE256

One Σ-Protocol for Both Exponents



- Teddy[Poseidon]: Com and/or F_{sk} are Poseidon
- Teddy[SHA512]: Com and/or F_{sk} are SHA512
- Teddy[SHA3]: Com and/or F_{sk} are SHAKE256
- ... Teddy[AES]: Com and/or F_{sk} are AES

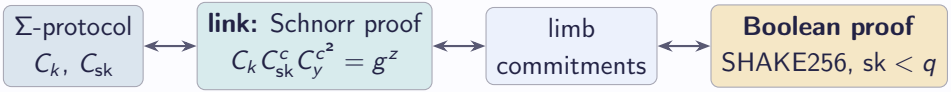
Teddy[Poseidon]: Native Commit-and-Prove



300	1120 B	DL
multiplication gates	nonce proof	in the ROM [DG23]

estimates; PRF security models keyed Poseidon as a random oracle (heuristic)

Teddy[SHA3]: SHAKE256 and the Link Argument



arithmetic baseline

all of SHAKE256 in Bulletproofs

$\leq$ **154,112** gates · **1696 B**

link argument

SHAKE256 stays Boolean

38,400 AND + backend proof

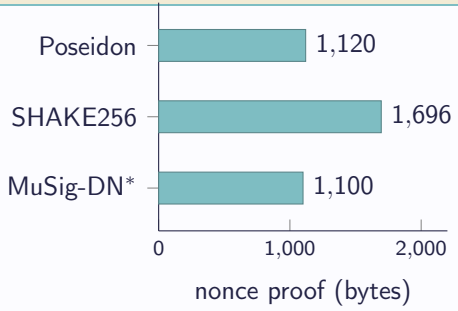
ZK and simulation extractability reduce to those of the Boolean backend

AES: Under Investigation

standardized (FIPS 197) and Boolean-friendly: it would reuse the link argument



Cost Overview (Estimates)



Bulletproofs instances; *reported ≈1.1 KB

[NRSW20]

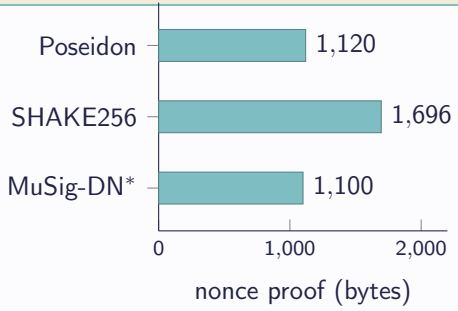
instance	circuit size
Poseidon, BP	300 mult.
SHAKE256, BP	$\leq$ 154,112 mult.
SHAKE256, link	38,400 AND + backend
AES, link	**under investigation**

BP: Bulletproofs

per session: 1 proof and $t-1$ verifications per signer

timings: benchmarks in the package

Cost Overview (Estimates)



Bulletproofs instances; *reported ≈1.1 KB

[NRSW20]

instance	circuit size
Poseidon, BP	300 mult.
SHAKE256, BP	$\leq$ 154,112 mult.
SHAKE256, link	38,400 AND + backend
AES, link	**under investigation**

BP: Bulletproofs

per session: 1 proof and $t-1$ verifications per signer

timings: benchmarks in the package

- Under investigation: SHA-512, Groth16 (setup, but 32ms+2ms and 512B proof size)
- Under investigation: Poseidon, Groth16 (setup, but 6ms+1ms and 256B proof size)

Takeaways

problem

message-only nonces leak the key in t-out-of-n, even without deviation

fix

bind the nonce to the signer set: 2 rounds, stateless, Ed25519/Ed448 verification

proofs

link argument
Poseidon: 1120 B
SHA-512:
SHAKE256
AES:
under investigation

ePrint available soon...

Contact: `octopus_mptc@googlegroups.com`



Thank you!

Questions?

Preliminary results

Table 4: Costs of nonce proofs in deterministic multi-party Schnorr signing, measured on our machine.

Scheme	Nonce derivation	Prove	Proof	Verify
MuSig-DN [54]	Purify, Bulletproofs	562 ms	1124 B	24.7 ms
GLRS26 [39], $d = 16$	equivocable polynomial	1.5 ms	640 B	0.56 ms
GLRS26 [39], $d = 16384$	equivocable polynomial	813 ms	1280 B	169 ms
GKMN21 [38]	ZK from garbled circuits	1.4 ms	980 KB	1.3 ms
Teddy (ours)	SHA-512, Groth16 with Locq	32 ms	512 B	1.85 ms
Teddy (ours)	Poseidon, Groth16	5.5 ms	256 B	0.83 ms

References

[BBD+23] Baum et al. Publicly verifiable zero-knowledge and post-quantum signatures from VOLE-in-the-head. CRYPTO 2023.

[BHH+19] Backes, Hanzlik, Herzberg, Kate, Pryvalov. Efficient non-interactive zero-knowledge proofs in cross-domains without trusted setup. PKC 2019.

[BHLS25] Boneh, Haitner, Lindell, Segev. Exponent-VRFs and their applications. EUROCRYPT 2025.

[BST21] Bonte, Smart, Tanguy. Thresholdizing HashEdDSA: MPC to the rescue. IJIS 2021.

[DG23] Dao, Grubbs. Spartan and Bulletproofs are simulation-extractable (for free!). EUROCRYPT 2023.

[FYZ+25] Feng, Yang, Zhang, Wang, Yu, Xie. Stateless deterministic multi-party EdDSA signatures with low communication. PKC 2025.

[GKMN21] Garillot, Kondi, Mohassel, Nikolaenko. Threshold Schnorr with stateless deterministic signing from standard assumptions. CRYPTO 2021.

[GLRS26] Gerhart, Li Calsi, Russo, Schröder. Fully-adaptive two-round threshold Schnorr signatures from DDH. EUROCRYPT 2026.

[KG25] Komlo, Goldberg. Arctic: lightweight and stateless threshold Schnorr signatures. PKC 2025.

[KMN23] Kadianakis, Maller, Novakovic. Sigmabus: binding sigmas in circuits for fast curve operations. ePrint 2023/1406.

[NRSW20] Nick, Ruffing, Seurin, Wuille. MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces. ACM CCS 2020.

[OKMZ25] Orrù, Kadianakis, Maller, Zaverucha. Beyond the circuit: how to minimize foreign arithmetic in ZKP circuits. IACR CiC 2025.

[GKR+21] Grassi, Khovratovich, Rechberger, Roy, Schofnegger. Poseidon: a new hash function for zero-knowledge proof systems. USENIX Security 2021.

[BBB+18] Bünz, Bootle, Boneh, Poelstra, Wuille, Maxwell. Bulletproofs: short proofs for confidential transactions and more. IEEE S&P 2018.

[RFC 8032] Josefsson, Liusvaara. Edwards-curve digital signature algorithm (EdDSA). 2017.

[RFC 9591] Connolly, Komlo, Goldberg, Wood. The FROST protocol for two-round Schnorr signatures. 2024.

[FIPS 186-5] NIST. Digital Signature Standard (DSS). 2023.

[NIST IR 8214C] Brandão, Peralta. NIST first call for multi-party threshold schemes. 2026.