# A Key-Recovery Attack on TALUS-MPC in TALUS v4

Sunghyeon Jo
QED Audit and Georgia Institute of Technology
a@qedaudit.io

August 19, 2026

**Abstract**

We give a two-transcript key-recovery attack on TALUS-MPC in TALUS v4. TALUS was presented in the second round of the NIST Threshold Call Preview Talks [4]. In Algorithm 3 of TALUS v4, the committee's nonce polynomials are evaluated at all $N$ points so that any $T$ parties can sign. An adversary controlling the coordinator and $T-1$ parties can reuse the same pooled nonce with two quorums whose only common members are corrupt. Each participating honest party uses its signing share once, yet the coordinator obtains

$$\boldsymbol{z}_0 = \boldsymbol{y} + c_0\boldsymbol{s}_1, \qquad \boldsymbol{z}_1 = \boldsymbol{y} + c_1\boldsymbol{s}_1.$$

The nonce cancels on subtraction. Moreover, no invertibility assumption in $R_q$ is needed: the response equations lift to $\mathbb{Z}[X]/(X^{256}+1)$, and every nonzero $c_0 - c_1$ is invertible in the cyclotomic field $\mathbb{Q}[X]/(X^{256}+1)$. Thus two distinct challenges recover $\boldsymbol{s}_1$ exactly; since the TALUS v4 public key includes the full $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2$, the adversary also recovers $\boldsymbol{s}_2$ and forges signatures.

## 1 Introduction

TALUS is a threshold construction for ML-DSA that produces signatures accepted by an unmodified FIPS 204 verifier [1, 5]. It specifies two deployments: one-round TALUS-TEE, assisted by a trusted execution environment, and two-round TALUS-MPC, based on honest-majority MPC. This note analyzes TALUS-MPC as specified in TALUS v4, dated July 23, 2026 [1]. Its offline protocol generates a Shamir sharing of one nonce and distributes an evaluation to every party. The resulting pooled nonce can then be used by any size-$T$ signing quorum.

This allows the same pooled nonce to be used by two signing quorums. Let an adversary control the aggregator (coordinator) and statically corrupt $T-1$ parties, as permitted by the threat model of TALUS-MPC. The coordinator runs two signing sessions using the same pooled nonce and two quorums that contain the same corrupt parties and different honest parties. The corrupt parties reuse their shares; each honest party responds in only one session. Both quorums nevertheless reconstruct the same nonce, yielding two linear equations whose difference reveals the signing key. The attack is sequential and requires two online response transcripts, whether or not either transcript passes the final rejection checks. This contradicts the EUF-CMA claim for two-round TALUS-MPC in Theorem 8.3 of TALUS v4.

Niot previously gave key-recovery attacks on an earlier version of TALUS by inverting public values of the form $\boldsymbol{A}\boldsymbol{v}$ and by exploiting the omitted $\boldsymbol{s}_2$ rejection step [3]. TALUS v4 removes the public $\boldsymbol{A}\boldsymbol{v}$ commitments. The attack here instead targets the all-$N$ nonce sharing specified in Algorithm 3. It uses the same cyclotomic-field observation as [3]: a nonzero polynomial of degree below 256 is invertible modulo $X^{256}+1$ over $\mathbb{Q}$.

## 2 Preliminaries

Let

$$q = 8380417, \qquad R_q = \mathbb{Z}_q[X]/(X^{256}+1),$$

and let bold lower-case letters denote vectors over $R_q$. The ML-DSA secret contains

$$\boldsymbol{s}_1 \in R_q^{\ell}, \qquad \boldsymbol{s}_2 \in R_q^{k}, \qquad \|\boldsymbol{s}_1\|_\infty, \|\boldsymbol{s}_2\|_\infty \le \eta.$$

The TALUS v4 public key includes the full value

$$\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2,$$

including its low part $\boldsymbol{t}_0$ [1]. For a challenge $c \in R_q$ produced by SampleInBall, exactly $\tau$ coefficients lie in $\{-1, 1\}$ and the others are zero. Consequently,

$$\|c\boldsymbol{s}_1\|_\infty \le \tau\eta = \beta.$$

We use a separate indeterminate $U$ for Shamir sharing [6]. Let

$$\boldsymbol{F}(U) \in R_q^{\ell}[U], \qquad \deg \boldsymbol{F} < T, \qquad \boldsymbol{F}(0) = \boldsymbol{s}_1,$$

so party $i$ holds the key share $\boldsymbol{s}_{1,i} = \boldsymbol{F}(i)$.

In Algorithm 3 of TALUS v4, each committee member $h \in \mathcal{C}$ samples a degree-$(T-1)$ polynomial $\boldsymbol{g}_h(U)$ and sends $\boldsymbol{g}_h(i)$ to every $i \in [N]$. Define

$$\boldsymbol{G}(U) = \sum_{h \in \mathcal{C}} \boldsymbol{g}_h(U), \qquad \boldsymbol{y} = \boldsymbol{G}(0), \qquad \widehat{\boldsymbol{y}}_i = \boldsymbol{G}(i).$$

Thus every party holds an evaluation of the same nonce-sharing polynomial, and every size-$T$ set reconstructs the same nonce $\boldsymbol{y}$. The same pooled nonce also fixes

$$\boldsymbol{w}_1 = \mathrm{HighBits}(\boldsymbol{A}\boldsymbol{y})$$

and the commitment $\mathsf{com} = H(\boldsymbol{w}_1 \,\|\, r)$.

For a message representative $\mu$ and a signing quorum $S \subseteq [N]$ of size $T$, Algorithm 2 computes

$$c = \mathrm{SampleInBall}\big(H(\mu \,\|\, \boldsymbol{w}_1)\big).$$

Each signer $i \in S$ sends

$$\boldsymbol{z}_i = \widehat{\boldsymbol{y}}_i + c\boldsymbol{s}_{1,i} = \boldsymbol{G}(i) + c\boldsymbol{F}(i).$$

Writing

$$\lambda_i^S = \prod_{j \in S \setminus \{i\}} \frac{j}{j-i}$$

for interpolation at zero, the coordinator obtains

$$\boldsymbol{z} = \sum_{i \in S} \lambda_i^S \boldsymbol{z}_i = \boldsymbol{y} + c\boldsymbol{s}_1. \tag{1}$$

The coordinator reconstructs $\boldsymbol{z}$ before the norm, hint-weight, and self-verification checks. The parties communicate through an untrusted aggregator (coordinator); parties outside $S$ do not take part in that signing session [1].

## 3 Cross-quorum nonce reuse

Fix $T \geq 2$, and let $\mathcal{B} \subseteq [N]$ be a corrupt set with $|\mathcal{B}| = T - 1$. Since TALUS-MPC requires $N \geq 2T - 1$, there are distinct honest parties $i_0, i_1 \notin \mathcal{B}$. Set

$$S_0 = \mathcal{B} \cup \{i_0\}, \qquad S_1 = \mathcal{B} \cup \{i_1\}.$$

Both are valid signing quorums, and their intersection contains only corrupt parties.

The coordinator selects one BCC-passing pooled nonce and runs Algorithm 2 with quorum $S_0$ on a message $\mu_0$. After receiving the response shares, it reconstructs

$$\boldsymbol{z}_0 = \boldsymbol{y} + c_0 \boldsymbol{s}_1, \qquad c_0 = \mathrm{SampleInBall}(H(\mu_0 \,\|\, \boldsymbol{w}_1)).$$

It may then abort the session. The coordinator next runs Algorithm 2 with quorum $S_1$ on a message $\mu_1$ using the same pooled nonce and the same opening $(\boldsymbol{w}_1, r)$, obtaining

$$\boldsymbol{z}_1 = \boldsymbol{y} + c_1 \boldsymbol{s}_1.$$

The corrupt parties reuse their evaluations of $\boldsymbol{G}$. Party $i_0$ uses its evaluation only in the first session, and party $i_1$ only in the second. Neither honest party sees the pooled nonce twice.

Subtracting gives

$$\boldsymbol{z}_0 - \boldsymbol{z}_1 = (c_0 - c_1)\boldsymbol{s}_1. \tag{2}$$

The following lemma places this identity in the exact integer cyclotomic ring.

**Lemma 1** (Exact lifting)**.** *Let*

$$R = \mathbb{Z}[X]/(X^{256} + 1).$$

*For each response reconstructed in the attack, the centered representative satisfies*

$$\boldsymbol{z}_b = \boldsymbol{y} + c_b \boldsymbol{s}_1$$

*in $R^\ell$, not only in $R_q^\ell$. Consequently, Equation* (2) *holds in $R^\ell$.*

*Proof.* Algorithm 3 restricts the aggregate nonce to $\|\boldsymbol{y}\|_\infty \leq \gamma_1$. For every ML-DSA challenge, $\|c_b \boldsymbol{s}_1\|_\infty \leq \beta$. Thus

$$\|\boldsymbol{y} + c_b \boldsymbol{s}_1\|_\infty \leq \gamma_1 + \beta < q/2$$

for all three FIPS 204 parameter sets. No coefficient wraps modulo $q$, so the centered representative in $R_q^\ell$ is the corresponding element of $R^\ell$. □

**Theorem 2** (Two-transcript key recovery)**.** *If $c_0 \neq c_1$, the adversary recovers $\boldsymbol{s}_1$ exactly in polynomial time from the two response transcripts. It then recovers $\boldsymbol{s}_2$ and can forge ML-DSA signatures under the corresponding public key.*

*Proof.* Set $d = c_0 - c_1$ and

$$K = \mathbb{Q}[X]/(X^{256} + 1).$$

Because $X^{256} + 1 = \Phi_{512}(X)$ is irreducible over $\mathbb{Q}$, $K$ is a field. The standard representatives of $c_0$ and $c_1$ have degree below 256, so $c_0 \neq c_1$ implies $d \neq 0$ in $K$. The extended Euclidean algorithm therefore computes $d^{-1}$, and Equation (2) yields

$$\boldsymbol{s}_1 = d^{-1}(\boldsymbol{z}_0 - \boldsymbol{z}_1) \in K^\ell. \tag{3}$$

This is the exact secret vector in $R^\ell$. Since the public key includes the full $\boldsymbol{t}$, the adversary computes

$$\boldsymbol{s}_2 = \boldsymbol{t} - \boldsymbol{A}\boldsymbol{s}_1.$$

The recovered short vectors suffice to run ML-DSA signing with fresh randomness. □

After the first session, the coordinator knows $\boldsymbol{w}_1$ and can evaluate the challenge for candidate second messages before contacting $S_1$. It selects any message producing $c_1 \neq c_0$. Under the random-oracle model, one trial succeeds except with negligible probability.

*Remark* 3 (No unit condition in $R_q$). When $d$ is a unit in $R_q$, Equation (3) can be evaluated slotwise in the NTT domain. If $d$ is a zero divisor modulo $q$, the exact lifted equation remains invertible in $K$ and recovers $\boldsymbol{s}_1$.

# 4 Why per-party nonce state is insufficient

The attack follows whenever a nonce-sharing polynomial is evaluated at all parties and only the active signing quorum records use of its shares.

**Proposition 4** (Reuse across signing quorums)**.** *Consider a $T$-of-$N$ linear-response protocol in which a nonce is represented by a degree-$(T-1)$ polynomial $\boldsymbol{G}(U)$ evaluated at all $N$ parties, and a quorum $S$ responds with evaluations of*

$$\boldsymbol{G}(U) + c\boldsymbol{F}(U).$$

*Assume that each party only records whether it has used its own nonce share. For $N \geq T+1$, an adversary corrupting $T-1$ parties can obtain two aggregate responses with the same free term $\boldsymbol{G}(0)$ while every honest party uses its nonce share at most once.*

*Proof.* Choose a corrupt set $\mathcal{B}$ of size $T-1$ and two honest parties $i_0, i_1$ outside it. Use the nonce with $S_b = \mathcal{B} \cup \{i_b\}$ for $b \in \{0,1\}$. Only corrupt parties occur in both quorums, while interpolation by either quorum returns the same free term $\boldsymbol{G}(0)$. □

TALUS v5 binds each preprocessed nonce to one fixed signing quorum and enforces atomic one-time use. Only that quorum receives the nonce shares, and the nonce material is erased after use. This closes the cross-quorum reuse path analyzed in this note [2].

# 5 Conclusion

TALUS-MPC v4 shared each pooled nonce with every party, making it usable by every size-$T$ quorum. With $T-1$ corrupt parties, two quorums can reuse that nonce without any honest party observing two uses. The resulting response transcripts cancel the nonce and recover the signing key from two distinct challenges. The attack does not require $c_0 - c_1$ to be invertible in $R_q$, because the response equation can be solved in $\mathbb{Q}[X]/(X^{256}+1)$.


# Acknowledgement

The author thanks Leo Kao and Raymond Chang for promptly confirming the attack and discussing the fix.


**AI usage disclosure.** An LLM harness using GPT-5.6 Sol was used in the discovery and development of the attack presented in this paper. OpenAI Codex was used to assist with manuscript preparation, including editing and consistency checks. The author independently verified the attack, including the analysis of the TALUS v4 protocol and the mathematical arguments, and takes full responsibility for the paper's contents and correctness.

# References


[1] Leo Kao and Raymond Chang. TALUS: FIPS-204-Exact Threshold ML-DSA via Boundary Clearance. arXiv:2603.22109v4, 2026. `https://arxiv.org/abs/2603.22109v4`.

[2] Leo Kao and Raymond Chang. TALUS: FIPS-204-Exact Threshold ML-DSA via Boundary Clearance. arXiv:2603.22109v5, 2026. https://arxiv.org/abs/2603.22109v5.

[3] Guilhem Niot. Key-Recovery Attacks on TALUS: A Cryptanalytic Note. Cryptology ePrint Archive, Paper 2026/1386, 2026. https://eprint.iacr.org/2026/1386.

[4] National Institute of Standards and Technology. Preview Talk by Team TALUS, Threshold Call Preview Talks 2. 2026. https://csrc.nist.gov/presentations/2026/tcpt2-2a1.

[5] National Institute of Standards and Technology. Module-Lattice-Based Digital Signature Standard. FIPS PUB 204, 2024. https://doi.org/10.6028/NIST.FIPS.204.

[6] Adi Shamir. How to Share a Secret. Communications of the ACM, 22(11):612–613, 1979.