Oriole: Adaptively Secure Partially Non-Interactive Threshold Signatures from Lattices

Kaijie Jiang[1], Hoeteck Wee[2], and Chenzhi Zhu[2]

[1] Institute for Advanced Study, Tsinghua University, Beijing, China
[2] NTT Research, California, US

Abstract. We present the first lattice-based, partially non-interactive threshold signature scheme that tolerates the adaptive corruption of up to $T-1$ signers, where T is the signing threshold. Our construction relies on the MSIS and MLWE assumptions, and has two rounds, of which only the second is message-dependent. We substantially improve upon prior adaptively secure lattice-based schemes (CRYPTO '24 and EUROCRYPT '26), which require at least two message-dependent rounds. In addition, compared to prior lattice-based partially non-interactive schemes (CRYPTO '24, S&P '25, CRYPTO '25), we achieve better communication complexity.

1 Introduction

In a T-out-of-N threshold signature [Des88, DF90], a group of N parties hold shares of a signing key so that any subset of at least T out of N parties can produce a valid signature, while any adversary corrupting fewer than T of these parties cannot forge new signatures. Threshold signatures play a crucial role in securing decentralized systems, distributed key management, and fault-tolerant cryptographic protocols. They are widely deployed in practice, particularly for securing digital wallets and other high-value assets. Ongoing IETF [CKGW22] and NIST [Natnt] standardization efforts further underscore their growing practical importance. Thanks to a substantial and fruitful body of recent work, we now have efficient threshold signature schemes from pairings [Bol03, BL22, DR24], RSA [DDFY94, GRJK07, Sho00], pairing-free groups [KG20, BCK+22, Mak22, TZ23, BLT+24, KRT24, BDLR25b, Che25a, CKK+25, BDLR25a, GCRS25, Che25b, BCL+26], and from lattices [BGGK17, BGG+18, ASY22, GKS24, DKM+24, EKT24, KRT24, CATZ24, ZT25, BKL+25, dPENP25, PKN+25, BCdP+25, BdCE+25, JTWZ26].

As deployment settings become increasingly adversarial and asynchronous, two properties emerge as central: minimal online interaction and strong adaptive security against adversaries that may choose which parties to corrupt as the protocol evolves. In particular, partially non-interactive signing—where a single round of message-independent preprocessing enables signature generation in a single online round—substantially improves scalability, latency, and robustness. However, among existing constructions, only [BL22, DR24, CKK+25, BCL+26, Che26] simultaneously achieve partial non-interactivity and adaptive security, and they all rely on classical group-based assumptions. Consequently, no known scheme provides these guarantees under post-quantum assumptions, leaving a fundamental gap in the lattice setting. Addressing this gap is particularly timely, as NIST has explicitly noted the importance of both post-quantum and adaptive security in potential threshold signature standards.

This work. In this work, we close this gap with Oriole, the first lattice-based threshold signature scheme that simultaneously achieves partially non-interactive signing and adaptive security. The security of our scheme is based on the Module Learning With Errors (MLWE) and Module Short Integer Solution (MSIS) assumptions in the random oracle model. This is a significant improvement over the only existing adaptively secure lattice-based threshold signature schemes, KRT [KRT24] and Tweed [JTWZ26], which require at least two message-dependent rounds. In addition, our scheme achieves strong unforgeability against adaptive corruption, which guarantees that the adversary cannot obtain one more signature from any signing session. It is unclear whether KRT or Tweed achieve strong unforgeability. In particular, Tweed is developed from a line of DDH-based constructions [BLT+24, Che25a], which do not naturally support strong unforgeability. The security amplification from unforgeability to strong unforgeability given by Azari et al. [ABHR25] relies on threshold chameleon hash functions, whose lattice instantiation is not yet known. Even for static security, no other lattice-based threshold signature scheme is shown to be strongly unforgeable except the recent work by Borin et al. [BCdP+26]. In Figure 1, we position our scheme in the broader context of the most closely related schemes.

Scheme	Rounds	Corruptions
[BGGK17, BGG+18, ASY22]	1	static
[EY24]	1	part. ad.
[GKS24, GTJ+26]	0 + 2	static
[DKM+24]	0 + 3	static
[EKT24, CATZ24, ZT25] [BKL+25, BCdP+26]	1 + 1	static
[KRT24]	2 + 3	adaptive
[FNR25, ENP24]	2 + 1	static
[JTWZ26]	0 + 2	adaptive
This work	1 + 1	adaptive

Scheme	Rounds	Assumption
Dazzle [Che25a]	0 + 2	DDH
HBTS-ms [Che25b]	0 + 2	DL
HBTS2-ms [Che26]	1 + 1	DDH
FROST [CKK+25]	1 + 1	AOMDL + LDVR + AGM
ms-FROST [BCL+26]	1 + 1	AOMDL + AGM
Tweed [JTWZ26]	0 + 2	MLWE + MSIS
This work	1 + 1	MLWE + MSIS

Fig. 1. Summary of T-out-of-N threshold signatures. The notation $X + Y$ denotes the number of offline (i.e., message independent) and online (i.e., message dependent) rounds, respectively. On the left, we list all existing constructions from lattices. On the right, we list threshold signatures proved to be adaptively secure from pairing-free groups and lattices, with at most two rounds. We only consider schemes that are stateless and without erasures, and for which optimal corruptions (up to $T - 1$ parties) are tolerated. Also, all schemes additionally rely on the random oracle model.

Regarding concrete efficiency, for 128-bit security, our parameter estimation indicates that Oriole produces signatures of approximately 160KB, which is similar to that reported for Tweed [JTWZ26]. While our signatures are larger than those of prior lattice-based partially non-interactive schemes achieving only static security [EKT24, ZT25, BKL+25, BCdP+26], which obtain sizes in the range of 10–30KB, our scheme achieves significantly better communication complexity, requiring roughly about 210KB per signer, compared to 280KB in [EKT24] (or 760KB based on analysis under the standard lattice assumptions [ZT25]) and 610KB in [BKL+25]. Still, we emphasize that our primary contribution lies in introducing new techniques for achieving adaptive security along with partial non-interactivity, and leave the improvement of parameters to future work.

Prior work & our approach. Existing lattice-based partially non-interactive schemes [EKT24, CATZ24, ZT25, BKL+25, BCdP+26] are all based on the Fiat-Shamir framework and can be viewed as lattice analogs of FROST, the state-of-the-art partially non-interactive threshold signatures in the pairing-free groups [KG20]. It appears to be hard to prove adaptive security for those FROST-like schemes. Many of them [EKT24, CATZ24, ZT25, BCdP+26] rely on rewinding-based security proofs, which are ill-suited for adaptive security, as the adversary can corrupt different sets of signers before and after rewinding potentially learning the signing key. FROST and also the mask version of FROST [BCL+26] (referred to as ms-FROST) avoid rewinding by developing security analysis [CKK+25, BCL+26] in the algebraic group model (AGM) [FKL18], which has no counterpart in the lattice setting. Bacho et al. [BCL+26] further show that it is not possible to prove adaptive security of ms-FROST (and FROST) without the AGM. No adaptively secure partially non-interactive scheme without relying on the AGM is known except the concurrent work by Chen [Che26] (discussed below), which is based on a different framework.

The only FROST-like scheme that has a non-rewinding security analysis is Ringtail [BKL+25], which leverages lattice trapdoors to avoid rewinding. However, we will explain in the technical overview (Section 2) that there are still obstacles with proving adaptive security for FROST-like schemes even with non-rewinding security proofs. The existing adaptively secure lattice-based schemes KRT [KRT24] and Tweed [JTWZ26] are adapted from adaptively secure schemes in the pairing-free groups with more rounds [KRT24, Che25a].

Given these challenges, we revisit the design space of lattice-based signatures. In contrast to Fiat–Shamir-style protocols, lattice-based hash-and-sign signatures admit non-rewinding proofs [GPV08, MP12, CGM19, YJW23, EEN+24], providing a more promising foundation for adaptive security. However, current threshold hash-and-sign schemes [FNR25, ENP24] achieve only static security and require three rounds, with a single message-dependent round.

Our approach builds on this alternative foundation. We first show that the hash-and-sign framework can be adapted to obtain a partially non-interactive threshold scheme with static security and without rewinding. We then develop new simulation techniques to prove adaptive security.

Concurrent work. In an independent and concurrent work [Che26], Chen presents a partially non-interactive threshold signature scheme based on DDH in pairing-free groups that tolerates adaptive corruption of up to $T-1$

signers. The scheme builds upon the 2-round multi-signature scheme, HBMS [BD21], and its threshold signature version, HBTS [Che25b], and introduces new techniques that make the scheme partially non-interactive.

In another concurrent work [BLT25], Boneh et al. give a 2-round threshold GPV signatures.[3] However, they only show static security of the scheme. Also, they assume that a unique session ID is shared among all signers for each signing session. Without such an assumption, it requires either one additional round to generate such a session ID, making their scheme 3-round in the usual sense, or each signer to maintain a state that records all previously used session IDs.

2 Technical Overview

We fix a ring $\mathcal{R}$ (either $\mathbb{Z}$ or $\mathbb{Z}[X]/(X^d+1)$) and let $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$.

Hash-and-sign from SIS. The starting point of our construction is the (salted) GPV signature [GPV08]:

- The public key consists of a matrix $\mathbf{A} \in \mathcal{R}_q^{n\times m}$, along with a hash function $H_1 : \{0,1\}^* \to \mathcal{R}_q^n$, and the secret key is a trapdoor $\mathbf{T}$ of $\mathbf{A}$.
- To sign a message μ, the signer first samples $\mathsf{salt} \leftarrow \{0,1\}^\lambda$, and then uses the trapdoor to sample a gaussian $\mathbf{z} \in \mathcal{R}_q^m$ such that $\mathbf{Az} = H_1(\mu, \mathsf{salt})$. It outputs $(\mathbf{z}, \mathsf{salt})$ as the signature.
- The verifier checks that $\|\mathbf{z}\|$ is small and that $\mathbf{Az} = H_1(\mu, \mathsf{salt})$.

The scheme satisfies (strong) unforgeability under (module) SIS. The security proof relies on the fact that we can simulate signatures on μ without a trapdoor for $\mathbf{A}$ by sampling a gaussian $\mathbf{z} \in \mathcal{R}_q^m$ along with a random salt and programming $H_1(\mu, \mathsf{salt}) = \mathbf{Az}$. Following [MP12, EEN+24], given $\mathbf{A}$ with trapdoor $\mathbf{T}$ such that $\mathbf{AT} = \mathbf{G}$ (where $\mathbf{G}$ is the gadget matrix), we compute the pre-image $\mathbf{z}$ of $H_1(\mu, \mathsf{salt})$ as follows:

$$\mathbf{z} = \mathbf{r} + \mathbf{T} \cdot \mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{Ar}) \ . \tag{1}$$

where $\mathbf{r}$ is random gaussian, $\mathbf{G}^{-1}(\cdot)$ is deterministic, and we use noise flooding with $\|\mathbf{r}\| \gg \|\mathbf{T}\|$ to ensure that $\mathbf{z}$ does not leak $\mathbf{T}$.

Our threshold signature. In the threshold scheme, signing is distributed across a subset SS of T out of N signers. We follow the same high-level approach as prior lattice-based schemes:

- Key generation assigns each party a Shamir T-out-of-N share $\mathbf{T}_i$ of the signing key $\mathbf{T}$ (so that any subset of $T-1$ shares are uniformly random matrices over $\mathcal{R}_q$). Following the practice of prior work (e.g., [EKT24, KRT24, JTWZ26]), we do not specify a concrete distributed key generation (DKG) protocol instead assume an ideal DKG.
- To sign a message μ with a set of involved signers SS ($|\mathrm{SS}| \geq T$), each party $i \in \mathrm{SS}$ samples a gaussian $\mathbf{r}_i$ and derives a partial signature $\mathbf{z}_i$ such that $\mathbf{r} = \sum_{i\in\mathrm{SS}} \mathbf{r}_i, \mathbf{z} = \sum_{i\in\mathrm{SS}} \mathbf{z}_i$ satisfy (1).

Concretely, signing consists of two rounds $\mathsf{Sign}_1, \mathsf{Sign}_2$, where the first round is message-independent:

$\underline{\mathsf{Sign}_1(i, \mathbf{T}_i)}$: output $\mathbf{w}_i = \mathbf{Ar}_i$.

$\underline{\mathsf{Sign}_2(i, \mathbf{T}_i, \mathbf{r}_i, \mu, \mathrm{SS}, \{(j, \mathbf{w}_j)\}_{j\in\mathrm{SS}})}$:

$$\begin{aligned} \mathbf{w} &= \textstyle\sum_{j\in\mathrm{SS}} \mathbf{w}_j \ . \\ \mathsf{salt} &= H_2(\mu, \mathrm{SS}, \{(j, \mathbf{w}_j)\}_{j\in\mathrm{SS}}) \ . \\ \text{output } \mathbf{z}_i &= \mathbf{r}_i + L_{\mathrm{SS},i} \cdot \mathbf{T}_i \cdot \mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w}) \ . \end{aligned}$$

Here, the communication is forwarded by an untrusted coordinator (can be one of the signers), who receives the first-round output $\mathbf{w}_i$, sends the second-round query $(\mu, \mathrm{SS}, \{(j, \mathbf{w}_j\}_{j\in\mathrm{SS}})$, and receives the second-round output $\mathbf{z}_i$ from each signer $i \in \mathrm{SS}$. The final signature is given by $(\sum_{i\in\mathrm{SS}} \mathbf{z}_i, \mathsf{salt})$. Correctness follows readily from $\mathbf{T} = \sum_{i\in\mathrm{SS}} L_{\mathrm{SS},i} \cdot \mathbf{T}_i$, so that $\mathbf{r} = \sum_{i\in\mathrm{SS}} \mathbf{r}_i, \mathbf{z} = \sum_{i\in\mathrm{SS}} \mathbf{z}_i$ indeed satisfy (1). We make one more modification to the scheme so far for the proof of security, where-in we incorporate additional masking following [EKT24]:

[3] In Section 7 of the 2026-05-23 revision; such a scheme was not in the original 2025-07-07 posting.

– During key generation, party i additionally receives a masking key mk_i that allows the party given SS and a salt $\mathsf{salt} \in \{0,1\}^*$, derive a mask $\mathsf{mask}_{i,\mathrm{SS},\mathsf{salt}}$, which is a (pseudo)random vector over $\mathcal{R}_q$ subject to the constraint $\sum_{i\in\mathrm{SS}} \mathsf{mask}_{i,\mathrm{SS},\mathsf{salt}} = \mathbf{0}$.
– During signing, party i computes $\mathbf{w}_i, \mathbf{z}_i$ as before, but outputs $\mathbf{z}'_i = \mathbf{z}_i + \mathsf{mask}_{i,\mathrm{SS},\mathsf{salt}}$ instead of $\mathbf{z}_i$.

The final signature is given by $(\sum_{i\in\mathrm{SS}} \mathbf{z}'_i, \mathsf{salt})$. Correctness follows as before since $\sum_{i\in\mathrm{SS}} \mathbf{z}'_i = \sum_{i\in\mathrm{SS}} \mathbf{z}_i$.

Comparison with prior hash-and-sign schemes. A crucial distinction from prior lattice-based threshold hash-and-sign schemes [ENP24, FNR25] is that we derive the salt from $\{(j, \mathbf{w}_j)\}_{j\in\mathrm{SS}}$ and not $\mathbf{w} = \sum_{j\in\mathrm{SS}} \mathbf{w}_j$. This ensures that an adversary cannot substantially bias the distribution of the salt. In particular, an adversary can bias $\mathbf{w}$ in our protocol (e.g., set $\mathbf{w} = \mathbf{0}$) by choosing its message in the first round as a function of the messages from the honest parties. If the salt is derived from $(\mu, \mathbf{w})$ as is the case in prior schemes, this would allow the adversary to bias the salt and thus $H_1(\mu, \mathsf{salt})$. Such an attack strategy is avoided in [ENP24] by using distributed key generation to sample $\mathbf{w}$, and in [FNR25] by using commit-then-open for $\mathbf{w}_i$; both of these work-arounds incur at least an extra round of communication. Another recent work [ALLW25] describes a non-interactive threshold variant of GPV signatures using completely different techniques, but the final signature size grows with the threshold (in particular, the dimensions of the matrix $\mathbf{A}$ underlying the GPV signature grows linearly with T), whereas our final signature size is independent of the threshold.

Comparison with prior partially non-interactive schemes. Previous partially non-interactive schemes use a FROST-like structure [EKT24, CATZ24, ZT25, BKL+25, BCdP+26]. The signing key sk is a low-norm vector in $\mathcal{R}_q^m$ and the corresponding public key is $\mathsf{pk} = \mathbf{A}\mathsf{sk}$. The signature is of the form $(\mathbf{z}, \mathbf{w})$ and is valid for a message μ if and only if $\mathbf{z}$ is a non-zero low-norm vector satisfying $\mathbf{Az} = \mathbf{w} + \mathsf{pk} \cdot H_1(\mathbf{w}, \mu)$.[4] The signing key sk is distributed among N signers via a T-out-of-N secret sharing scheme. During the first round of signing, each signer samples multiple low-norm vectors $\{\mathbf{r}_i^{(k)}\}_{k\in[\ell]}$ and outputs $\{\mathbf{w}_i^{(k)} = \mathbf{A}\mathbf{r}_i^{(k)}\}_{k\in[\ell]}$. During the second round of signing, each signer computes $\mathbf{w} = \sum_{j\in\mathrm{SS},k\in[\ell]} \mathbf{w}_j^{(k)} \cdot b_k$, where $(b_1, \ldots, b_\ell) = H_2(\mu, \mathrm{SS}, \{(j, \{\mathbf{w}_j^{(k)}\}_{k\in[\ell]})\}_{j\in\mathrm{SS}})$, and returns a partial signature $\mathbf{z}_i = \sum_{k\in[\ell]} \mathbf{r}_i^{(k)} \cdot b_k + L_{\mathrm{SS},i} \cdot \mathsf{sk}_i \cdot H_1(\mathbf{w}, \mu)$, where sk_i denotes the signing key share of signer i and $L_{\mathrm{SS},i}$ denotes the reconstruction coefficient. A mask might also be added to the partial signature depending on the underlying secret sharing scheme.

We can see that a major distinction between the FROST-like schemes and our construction is that we do *not* need to generate ℓ pairs of $(\mathbf{r}_i, \mathbf{w}_i)$ in the first round, where $\ell = O(\kappa)$ and is usually set to about 50, and thus achieves a much smaller communication complexity. Generating multiple $(\mathbf{r}_i, \mathbf{w}_i)$ is necessary in their security analysis to simulate signing sessions, while (discussed below) we can simulate a signing session by programming the random oracle H_1.

Overview of static security. As a warm-up, we proceed to provide a proof overview for static security, where the bulk of the work lies in simulating the view of the adversary given only $\mathbf{A}$. A static adversary first specifies a set $\mathrm{CS} \subseteq [N]$ of corrupted parties, where $|\mathrm{CS}| < T$, and learns $\{\mathbf{T}_i\}_{i\in\mathrm{CS}}$, which we can simulate using uniformly random matrices over $\mathcal{R}_q$. Then, the adversary playing as the role of a malicious coordinator can make arbitrary signing queries to any honest signers, and we need to simulate the outputs from the honest parties:

– For the first-round output $\mathbf{w}_i$, we proceed exactly as in Sign_1 and computes $\mathbf{w}_i = \mathbf{A}\mathbf{r}_i$. (Sampling a random $\mathbf{w}_i$ also works for static security.)
– For a second-round query $(\mu, \mathrm{SS}, \{(j, \mathbf{w}_j)\}_{j\in\mathrm{SS}})$ the collection of partial signatures $\{\mathbf{z}'_i\}_{i\in\mathrm{SS}\setminus\mathrm{CS}}$ only leak their sum thanks to the masks. In particular, if the adversary only queries a subset of signers $\mathrm{SS}' \subseteq \mathrm{SS} \setminus \mathrm{CS}$, then $\{\mathbf{z}'_i\}_{i\in\mathrm{SS}'}$ looks uniformly random due to the masks, and the queries can be answered by sampling $\mathbf{z}'_i$ uniformly. As such, it suffices to simulate the partial sum $\sum_{i\in\mathrm{SS}\setminus\mathrm{CS}} \mathbf{z}_i$, which we can rewrite as $\sum_{i\in\mathrm{SS}\setminus\mathrm{CS}} \mathbf{r}_i + \mathbf{T} \cdot \mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w}) - \sum_{i\in\mathrm{SS}\cap\mathrm{CS}} L_{\mathrm{SS},i} \cdot \mathbf{T}_i \cdot \mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w})$. To simulate the preceding expression, it suffices[5] to simulate

$$\hat{\mathbf{z}} = \sum_{i\in\mathrm{SS}\setminus\mathrm{CS}} \mathbf{r}_i + \mathbf{T} \cdot \mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w}) \,. \tag{2}$$

[4] Some [EKT24, BKL+25, BCdP+26] do further optimization by the bit truncation technique and output Raccoon signatures [dPKPR24]. For simplicity, we stick to the basic form here to explain the main ideas.
[5] We can compute the preceding expression given $\hat{\mathbf{z}}$ in (2) along with $\{\mathbf{T}_i\}_{i\in\mathrm{CS}}$.

Now, thanks to the entropy in $\{\mathbf{r}_i\}_{i\in \mathrm{SS}\setminus \mathrm{CS}}$, $\hat{\mathbf{z}}$ does not leak much information about the trapdoor $\mathbf{T}$. In particular, we can simply simulate $\hat{\mathbf{z}}$ by sampling a random gaussian vector $\tilde{\mathbf{z}}$ (of the appropriate norm) and then programming $H_1(\mu, \mathsf{salt})$ satisfying

$$\mathbf{A}\cdot\hat{\mathbf{z}} = H_1(\mu,\mathsf{salt}) - \mathbf{w} + \sum_{i\in \mathrm{SS}\setminus \mathrm{CS}} \mathbf{w}_i \ , \tag{3}$$

We can show that the distribution of the simulated $\hat{\mathbf{z}}$ has small Rényi divergence from the real distribution (2).

We stress that the above simulation strategy crucially uses the fact that we derive salt from $\{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}}$ and not $\mathbf{w}$. This ensures that every $(\mu, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$ yields a different salt (via collision-resistance of H_2), which allows us to program a different $H_1(\mu,\mathsf{salt})$ each time.

For unforgeability, we can show that, by querying $(\mu^*, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$, the adversary can only obtain valid signatures $(\mathbf{z},\mathsf{salt})$ for μ^* with $\mathsf{salt} = H_2(\mu^*, \mathrm{SS}, , \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$. This guarantee is slightly stronger than normal unforgeability as we additionally require $\mathsf{salt} = H_2(\mu^*, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$. This additional requirement is used for proving strong unforgeability as discussed below. Suppose that the adversary outputs a valid signature $(\mathbf{z}^*, \mathsf{salt}^*)$ for a message μ^*, but it did not *complete* any signing session $(\mu, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$ satisfying $\mu^* = \mu$ and $\mathsf{salt}^* = H_2(\mu^*, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$. Here, completing a session means receiving all partial signatures $\{\mathbf{z}'_i\}_{i\in \mathrm{SS}\setminus \mathrm{CS}}$. We can extract a MSIS solution from the forgery as in the vanilla GPV signatures: we guess which query to H_1 corresponds to the forgery, and we program its output to be $\mathbf{A}\mathbf{z}_0$ for a random gaussian $\mathbf{z}_0$. Suppose we guess it correctly. After receiving the forgery from the adversary, it holds that $\mathbf{A}\mathbf{z}^* = H_1(\mu^*, \mathsf{salt}^*) = \mathbf{A}\mathbf{z}_0$, and then we can output $\mathbf{z}_0 - \mathbf{z}^*$, which with overwhelming probability constitutes a valid MSIS solution w.r.t. $\mathbf{A}$. Note that although $H_1(\mu^*, \mathsf{salt}^*)$ is programmed differently, since the adversary did not make the second-round queries $(\mu^*, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$ to all honest signers in SS, as discussed above, those queries can be simulated by sampling $\mathbf{z}'_i$ uniformly.

Analogous to strong unforgeability of digital signature schemes, strong unforgeability of threshold signatures additionally guarantees that the adversary cannot obtain more than one signature from any signing session. Thanks to the above unforgeability guarantee, strong unforgeability of our scheme follows naturally from strong unforgeability of the underlying GPV signatures. Suppose the adversary can output two distinct valid signatures by engaging in only one signing session $(\mu, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$. From unforgeability, the signatures $(\mathbf{z},\mathsf{salt})$ and $(\mathbf{z}',\mathsf{salt}')$ output by the adversary must satisfy $\mathsf{salt} = \mathsf{salt}' = H_2(\mu, \mathrm{SS}, \{(j,\mathbf{w}_j)\}_{j\in \mathrm{SS}})$. However, this gives a valid MSIS solution w.r.t. $\mathbf{A}$ as $\mathbf{A}\mathbf{z} = H_1(\mu,\mathsf{salt}) = \mathbf{A}\mathbf{z}'$ and $\mathbf{z}\neq\mathbf{z}'$.

Challenges in proving adaptive security. Moving from static security to adaptive security, the only difference is that the adversary can corrupt an honest signer i and, in particular, learn $\mathbf{r}_i$ after receiving the second-round responses. More precisely, the simulator needs to first simulate $\{\mathbf{w}_i\}_{i\in \mathrm{SS}}$, then simulate $\hat{\mathbf{z}}$, and finally simulate $\mathbf{r}_i$ for any newly corrupted signer i, while, in the static case, the simulator only needs to simulate $\{\mathbf{w}_i\}_{i\in \mathrm{SS}}$ and $\hat{\mathbf{z}}$. Note that even though the adversary can only corrupt $T-1$ signers, which means that at least one of $\{\mathbf{r}_i\}_{i\in \mathrm{SS}}$ is not leaked, the simulator does not know which $\mathbf{r}_i$ will be leaked ahead of time. Therefore, it is actually necessary for the simulator to be able to simulate *each* $\mathbf{r}_i$ generated by honest signers.

It is not hard to see that our simulation strategy described above fails for the adaptive security. Consider an adversary that first asks for all the partial signatures corresponding to (μ, SS) where $\mathrm{SS} = \{1,\dots,T\}$, and then corrupts parties $1, 2, \dots, T-1$ to learn $\mathbf{T}_1, \dots, \mathbf{T}_{T-1}, \mathbf{r}_1, \dots, \mathbf{r}_{T-1}$. From the partial signatures, it can compute the corresponding $\hat{\mathbf{z}}$. With the current simulation strategy, the norm of $\hat{\mathbf{z}} - (\mathbf{r}_1 + \cdots + \mathbf{r}_{T-1})$ will be much larger than it should be (roughly $\sqrt{T}\cdot\|\mathbf{r}_1\|$ instead of $\|\mathbf{r}_1\|$).

Similar challenges arise in all existing FROST-like schemes [EKT24, CATZ24, ZT25, BKL+25, BCdP+26], where the simulator needs to first simulate the first-round messages, then the second-round responses, and finally the randomness used to generate the first-round messages. In particular, in the FROST-like schemes, an aggregated second-round response is of the form $\mathbf{z} = \sum_{i\in \mathrm{SS}, k\in[\ell]} \mathbf{r}_i^{(k)}\cdot b_k + \mathsf{sk}\cdot H(\mathsf{pk}, \sum_{i\in \mathrm{SS}}\mathbf{w}_i, \mu)$. Intuitively, the ability to simulate $\mathbf{z}$ and all $\{\mathbf{r}_i^{(k)}\}_{k\in[\ell]}$ values implies the knowledge of the secret key sk, and thus it is hard to construct a simulator that can extract useful information from the adversary.

Our solution. To address this issue, our key observation is that, due to the use of the hash-and-sign signatures, it is easy to find short $\hat{\mathbf{z}}$ satisfying Equation (3) without any trapdoor by programming the random oracle even if all $\{\mathbf{r}_i\}_{i\in \mathrm{SS}\setminus \mathrm{CS}}$ are given. Concretely, we sample $\mathbf{r}_i$ as before and modify our simulation strategy for $\hat{\mathbf{z}}$ as follows:

– sample a random gaussian vector $\tilde{\mathbf{y}}$ (with $\|\tilde{\mathbf{y}}\| \approx \|\mathbf{T}\|$), i.e. we use $\tilde{\mathbf{y}}$ as a proxy for $\mathbf{T}\cdot\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})$;
– simulate $\hat{\mathbf{z}} = \tilde{\mathbf{y}} + \sum_{i\in \mathrm{SS}\setminus \mathrm{CS}} \mathbf{r}_i$;
– program $H_1(\mu,\mathsf{salt}) = \mathbf{A}\tilde{\mathbf{y}} + \mathbf{w}$ (satisfying (3) as before).[6]

Here, CS denotes the set of corrupted signers when the adversary queries $H_2(\mu, \mathrm{SS}, , \{(i,\mathbf{w}_i)\}_{i\in\mathrm{SS}})$.

Next, we claim that for any $i^* \in \mathrm{SS}\setminus\mathrm{CS}$, we have:

$$\begin{aligned}&\Big(\{(i,\mathbf{w}_i)\}_{i\in\mathrm{SS}}, \quad \sum_{i\in\mathrm{SS}\setminus\mathrm{CS}} \mathbf{r}_i + \mathbf{T}\cdot\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w}), \quad H_1(\mu,\mathsf{salt}), \quad \{\mathbf{r}_i\}_{i\in\mathrm{SS}\setminus\mathrm{CS}, i\neq i^*}\Big)\\ \approx_s &\Big(\{(i,\mathbf{w}_i)\}_{i\in\mathrm{SS}}, \quad \sum_{i\in\mathrm{SS}\setminus\mathrm{CS}} \mathbf{r}_i + \tilde{\mathbf{y}}, \quad \mathbf{A}\tilde{\mathbf{y}}+\mathbf{w}, \quad \{\mathbf{r}_i\}_{i\in\mathrm{SS}\setminus\mathrm{CS}, i\neq i^*}\Big).\end{aligned}$$

To prove this claim, we first replace $H_1(\mu,\mathsf{salt})$ on the LHS with $\mathbf{A}\tilde{\mathbf{y}}+\mathbf{w}$. Then, we use $(\mathbf{A}\mathbf{r}_{i^*}, \mathbf{r}_{i^*} + \mathbf{T}\cdot\mathbf{G}^{-1}(\mathbf{A}\tilde{\mathbf{y}})) \approx_s (\mathbf{A}\mathbf{r}_{i^*}, \mathbf{r}_{i^*}+\tilde{\mathbf{y}})$ given $\mathbf{A}, \tilde{\mathbf{y}}$. This follows from noise flooding over $\Lambda_q^\perp(\mathbf{A})$ given the standard deviation of $\mathbf{r}_{i^*} \gg \max\{\|\mathbf{T}\cdot\mathbf{G}^{-1}(\mathbf{A}\tilde{\mathbf{y}})\|, \|\tilde{\mathbf{y}}\|\}$. To prove the adaptive security of our scheme, we need to prove a more complex statement where the index i^* could be adaptively chosen. A simple solution is to guess a random $i^* \in \mathrm{SS}$ and combine this with a hybrid argument across Q_s signing queries, with a total security loss of $T\cdot Q_s$. Still, this analysis results in an exponential-size modulus.

In order to achieve smaller modulus size and better concrete parameters, we use Rényi divergence to analyze the difference between the real and the simulated views, similar to [CATZ24]. Concretely, to show the Rényi divergence between the real and the simulated views is small, it is enough to show that for any value received by the adversary, conditioned on the transcripts the adversary has received so far, the distributions of the value in the real and simulated worlds has small Rényi divergence. In particular, we need to establish this property for any $\hat{\mathbf{z}}$ and $\mathbf{r}_i$ received by the adversary. By extending the tool-kit for discrete Gaussians developed in [GMPW20, CATZ24], we can compute the distributions of $\hat{\mathbf{z}}$ (or $\mathbf{r}_i$) in the real and simulated worlds conditioned on the same prior transcripts, and show that they follow discrete Guassian distributions over the same lattice, of which we can easily bound the Rényi divergence. We refer to Sections 3.3 and 4.3 for more details.

Finally, the extraction of a MSIS solution from the forgery follows exactly as the static case.

Optimizations. We use the truncated trapdoor techniques following [CGM19, YJW23, EEN+24] to obtain smaller signature sizes. We sample $\mathbf{A}_0 \leftarrow \mathcal{R}_q^{n\times k}$ along with low-norm matrices $\mathbf{R} \leftarrow \mathcal{R}_q^{k\times\ell}, \mathbf{E} \leftarrow \mathcal{R}_q^{n\times\ell}$ and define

$$\begin{aligned}\mathbf{A} &= [\mathbf{I}_n \mid \mathbf{A}_0 \mid \mathbf{A}_0\mathbf{R}+\mathbf{E}+\mathbf{G}_1] \in \mathcal{R}_q^{n\times m},\\ \mathbf{G}_1 &= \lceil q/p^t\rceil \cdot \Big[\mathbf{I}_n \mid p\mathbf{I}_n \mid \cdots \mid p^{t-1}\mathbf{I}_n\Big] \in \mathcal{R}_q^{n\times tn},\end{aligned}$$

where $m = k+n+\ell, \ell = tn$.[7] We can define $\mathbf{G} := [\mathbf{I}_n \mid \mathbf{G}_1]$ so that

$$\mathbf{A}\cdot \overbrace{\begin{bmatrix}\mathbf{I}_n & -\mathbf{E}\\ \mathbf{0} & -\mathbf{R}\\ \mathbf{0} & \mathbf{I}_\ell\end{bmatrix}}^{\mathbf{T}} = \mathbf{G}.$$

We can then bound $\max_{\mathbf{u}\in\mathcal{R}_q^n}\|\mathbf{T}\cdot\mathbf{G}^{-1}(\mathbf{u})\|$ by roughly $\frac{q}{p^t} + p\cdot\max\{\|\mathbf{R}\|, \|\mathbf{E}\|\}$ and choose an optimal p to minimize it. We defer further details to Sections 4.5 and 4.6.

3 Preliminaries

3.1 Notation

For any positive integers $k < n$, $[n]$ denotes $\{1,\ldots,n\}$, and $[k..n]$ denotes $\{k,\ldots,n\}$. We use κ to denote the security parameter. For a sequence of variables $x_1,\ldots,x_\ell$, we use $x_{[i]}$ to denote $(x_1,\ldots,x_i)$ and $x_{[i..j]}$ to denote $(x_i,\ldots,x_j)$.

[6] We mention here an additional subtlety that arises for adaptive corruptions, namely the set CS when we simulate the second-round message may be different from that when we program H_1. Programming $H_1(\mu,\mathsf{salt}) = \mathbf{A}\tilde{\mathbf{y}}+\mathbf{w}$ has the advantage that it is oblivious to CS, unlike the earlier strategy in (3).

[7] Previous lattice-based threshold hash-and-sign use $n = t = \ell = 1$, following [EEN+24].

For a finite set S, $|S|$ denotes the size of S, and $x \leftarrow S$ denotes sampling an element uniformly from S and assigning it to x. For a distribution $\mathcal{D}$, denote the support by $\mathsf{Supp}(\mathcal{D})$, and $x \leftarrow \mathcal{D}$ denotes sampling x according to $\mathcal{D}$. For any $y \in \mathsf{Supp}(\mathcal{D})$, denote $\mathcal{D}(y) := \Pr_{x \leftarrow \mathcal{D}}[x = y]$, and for $S \subseteq \mathsf{Supp}(\mathcal{D})$, denote $\mathcal{D}(S) := \Pr_{x \leftarrow \mathcal{D}}[x \in S]$. For any set T and any function $F : \mathsf{Supp}(\mathcal{D}) \to T$, denote $F(\mathcal{D})$ as the distribution of $F(x)$ for x sampled from $\mathcal{D}$. We denote by $\mathcal{B}_n$ the closed Euclidean ball of radius 1 around the origin, that is $\mathcal{B}_n := \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| \leq 1\}$.

For a distribution $\mathcal{D}$, we denote $\mathcal{D}^k$ as a matrix $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_k)$ with k columns, and each column is drawn i.i.d. from distribution $\mathcal{D}$, where $\mathsf{Supp}(\mathcal{D})$ is a column vector space.

For a matrix $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_n)$, we denote its Gram-Schmidt orthogonalization by $\widetilde{\mathbf{A}} = (\widetilde{\mathbf{a}}_1, \ldots, \widetilde{\mathbf{a}}_n)$, and we use $\|\mathbf{A}\|$ to denote the maximum norm among its columns, namely, $\max_{i \in [n]} \|\mathbf{a}_i\|$.

3.2 Polynomial Rings

Let q be an odd prime and d be a power of 2. We denote the ring $\mathcal{R} := \mathbb{Z}[X]/(X^d+1)$, contained in the cyclotomic field $K := \mathbb{Q}[X]/(X^d+1)$, and let $\mathcal{R}_q := \mathcal{R}/q\mathcal{R} \cong \mathbb{Z}_q[X]/(X^d+1)$. Denote $K_{\mathbb{R}} := \mathbb{R} \otimes K \cong \mathbb{R}[X]/(X^d+1)$. For an element $v \in K_{\mathbb{R}}$, where $v = \sum_{i=0}^{d-1} v_i X^i$, we denote its conjugate as $v^* = \sum_{i=0}^{d-1} -v_i X^{d-i}$. We use ϕ to denote the coefficient embedding that embeds $K_{\mathbb{R}}$ in $\mathbb{R}^d$, and ϕ maps v to vector $(v_0, \ldots, v_{d-1}) \in \mathbb{R}^d$. When applying ϕ to a vector $\mathbf{v} \in K_{\mathbb{R}}^m$, ϕ maps $\mathbf{v}$ to a vector in $\mathbb{R}^{md}$ by applying ϕ to each entry of $\mathbf{v}$. The map ϕ is a bijection, and we denote its inverse by ϕ^{-1}. An ℓ_p-norm of $\mathbf{v} \in K_{\mathbb{R}}^m$ is given by $\|\mathbf{v}\|_p := \|\phi(\mathbf{v})\|_p = \left(\sum_{i=1}^m \sum_{j=0}^{d-1} |v_{i,j}|^p\right)^{\frac{1}{p}}$, where $v_{i,j}$ denotes the coefficient of X^j of the i-th entry of $\mathbf{v}$. Additionally, the ℓ_∞-norm of $\mathbf{v}$ is defined as $\|\mathbf{v}\|_\infty := \max_{i \in [m], j \in [0..d-1]} |v_{i,j}|$. For the ℓ_2-norm, we omit the subscript and denote $\|\mathbf{v}\|$ as the ℓ_2-norm of $\mathbf{v}$. Denote the conjugate transpose of $\mathbf{v} \in K_{\mathbb{R}}^m$ as $\mathbf{v}^\dagger := (\mathbf{v}^*)^\top$. We define the inner product of two vectors $\mathbf{v}, \mathbf{v}' \in K_{\mathbb{R}}^m$ as $\langle \mathbf{v}, \mathbf{v}' \rangle := \phi(\mathbf{v})^\top \phi(\mathbf{v}') = \langle \phi(\mathbf{v}), \phi(\mathbf{v}') \rangle$. We have $\|\mathbf{v}\|^2 = \langle \mathbf{v}, \mathbf{v} \rangle$. We say $\mathbf{v}$ is a unit vector if $\|\mathbf{v}\| = 1$.

Also, we define a map ϕ_M that maps each element in $K_{\mathbb{R}}$ to a matrix in $\mathbb{R}^{d \times d}$ as follows. Let $M_X := \begin{pmatrix} \mathbf{0} & -1 \\ \mathbf{I}_{d-1} & \mathbf{0} \end{pmatrix} \in \mathbb{R}^{d \times d}$, where $\mathbf{I}_{d-1}$ is the identity matrix in $\mathbb{R}^{d-1}$. For each $v \in K_{\mathbb{R}}$, $\phi_M(v) := \sum_{i=0}^{d-1} v_i M_X^i$, which can be viewed as the matrix representation of v. In particular, for ϕ and ϕ_M, the following properties hold: for any $v, v' \in K_{\mathbb{R}}$, $\phi_M(v^*) = \phi_M(v)^\top$, $\phi_M(vv') = \phi_M(v)\phi_M(v')$ and $\phi_M(v)\phi(v') = \phi(vv')$. We extend the above definitions to $\mathcal{R}_q$ by representing each $v \in \mathcal{R}_q$ as $v = \sum_{i=0}^{d-1} v_i X^i$, where $v_i \in \{-(q-1)/2, \ldots, (q-1)/2\}$.

For a matrix $\mathbf{M} \in K_{\mathbb{R}}^{m \times m}$, we denote its conjugate transpose as $\mathbf{M}^\dagger = (\mathbf{M}^*)^\top$, and we say $\mathbf{M}$ is *hermitian* if $\mathbf{M} = \mathbf{M}^\dagger$. We say $\mathbf{M}$ is *positive definite* if and only if $\mathbf{M}$ is hermitian and for all $\mathbf{x} \in K_{\mathbb{R}}^m \setminus \{\mathbf{0}\}$, $\langle \mathbf{x}, \mathbf{M}\mathbf{x} \rangle > 0$, or equivalently, $\phi_M(\mathbf{M})$ is positive definite. Also, denote $\sigma_{\min}(\mathbf{M}) := \inf_{\mathbf{x} \in K_{\mathbb{R}}^m, \|\mathbf{x}\|=1} \|\mathbf{M}\mathbf{x}\|$ as the smallest singular value of $\mathbf{M}$ and $\sigma_{\max}(\mathbf{M}) := \sup_{\mathbf{x} \in K_{\mathbb{R}}^m, \|\mathbf{x}\|=1} \|\mathbf{M}\mathbf{x}\|$ as the largest singular value of $\mathbf{M}$.

3.3 Lattices and Discrete Gaussian Distributions

In this subsection, we give definitions for lattices and discrete Gaussian distributions over $\mathbb{R}$ and $K_{\mathbb{R}}$. An m-dimensional lattice Λ over $\mathbb{Z}$ (resp. $\mathcal{R}$) is a discrete additive subgroup of $\mathbb{Z}^m$ (resp. $\mathcal{R}^m$). Equivalently, $\Lambda = \Lambda(\{\mathbf{b}_1, \ldots, \mathbf{b}_k\}) := \{\sum_{i \in [k]} x_i \mathbf{b}_i : x_i \in \mathbb{Z}\}$ for a set of linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k \in \mathbb{Z}^m$ (resp. $\mathcal{R}^m$), which is referred to as a basis of Λ. The size k is the *rank* of the lattice Λ. We say Λ is a *full rank* lattice if $k = m$ (resp. $k = md$ for Λ over $\mathcal{R}$). For any $\mathbf{a} \in \mathbb{Z}^m$ (resp. $\mathcal{R}^m$), $\Lambda + \mathbf{a}$ is a *coset* of Λ. The *dual lattice* of Λ is denoted as $\Lambda^* = \{\mathbf{x} \in \mathsf{Span}(\Lambda) : \forall\ \mathbf{y} \in \Lambda, \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}\}$. A Λ-subspace is the linear span of some subset of Λ, i.e., a subspace S such that $S = \mathsf{Span}(S \cap \Lambda)$.

The *minimum distance* of a n-dimensional lattice Λ, denoted $\lambda_1(\Lambda)$, is the minimum distance between any two distinct lattice points. Equivalently, it is the Euclidean norm of a shortest nonzero lattice vector:

$$\lambda_1(\Lambda) \ := \ \min\{\|\mathbf{x} - \mathbf{y}\| : \mathbf{x} \neq \mathbf{y} \in \Lambda\} \ = \ \min\{\|\mathbf{x}\| : \mathbf{x} \in \Lambda \setminus \{\mathbf{0}\}\}.$$

More generally, the i-th successive minimum $\lambda_i(\Lambda)$ is the smallest radius r such that the ball $r \cdot \mathcal{B}_n$ contains i linearly independent lattice vectors:

$$\lambda_i(\Lambda) := \min\{r : \dim(\mathsf{span}(\Lambda \cap r \cdot \mathcal{B}_n)) \geq i\}.$$

For a matrix $\mathbf{A} \in \mathcal{R}_q^{k\times m}$, we define the $\mathcal{R}$-lattice $\Lambda_q^\perp(\mathbf{A}) \subseteq \mathcal{R}^m$ as

$$\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathcal{R}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \mod q\} .$$

We know $\Lambda_q^\perp(\mathbf{A})$ has full-rank since $q\mathcal{R}^m \subseteq \Lambda_q^\perp(\mathbf{A})$.

For a positive definite matrix $\Sigma \in \mathbb{R}^{m\times m}$ (resp. a positive definite matrix $\Sigma \in K_{\mathbb{R}}^{m\times m}$) and a vector $\mathbf{c} \in \mathbb{R}^m$ (resp. $K_{\mathbb{R}}^m$), we define the function $\rho_{\Sigma,\mathbf{c}}$ over $\mathbb{R}^m$ (resp. $K_{\mathbb{R}}^m$) as

$$\rho_{\Sigma,\mathbf{c}}(\mathbf{x}) := \exp\left(-\pi\langle \mathbf{x}-\mathbf{c}, \Sigma^{-1}(\mathbf{x}-\mathbf{c})\rangle\right) .$$

Then, we denote $\mathcal{D}_{\Lambda+\mathbf{a},\Sigma,\mathbf{c}}$ as the discrete Gaussian distribution over a lattice coset $\Lambda+\mathbf{a} \subseteq \mathbb{Z}^m$ (resp. $\mathcal{R}^m$) with covariance matrix Σ, centered at $\mathbf{c} \in \mathbb{R}^m$, where for $\mathbf{x} \in \Lambda+\mathbf{a}$, we define

$$\mathcal{D}_{\Lambda+\mathbf{a},\Sigma,\mathbf{c}}(\mathbf{x}) := \Pr[\mathbf{x} \leftarrow \mathcal{D}_{\Lambda+\mathbf{a},\Sigma,\mathbf{c}}] = \frac{\rho_{\Sigma,\mathbf{c}}(\mathbf{x})}{\rho_{\Sigma,\mathbf{c}}(\Lambda+\mathbf{a})} ,$$

where $\rho_{\Sigma,\mathbf{c}}(\Lambda+\mathbf{a}) = \sum_{\mathbf{x}\in\Lambda+\mathbf{a}} \rho_{\Sigma,\mathbf{c}}(\mathbf{x})$.

The smoothing parameter of a lattice Λ with respect to $\varepsilon > 0$, denoted by $\eta_\varepsilon(\Lambda)$, is the smallest $s > 0$ such that $\rho_{1/s}(\Lambda^*) \le 1+\varepsilon$.

We adopt the notation $P \overset{\varepsilon}{\approx} Q$ from [GMPW20]: for any two distributions P, Q with the same support and $\varepsilon > 0$, we say that $P \overset{\varepsilon}{\approx} Q$ if and only if $\max_{x\in\mathrm{Supp}(P)} |\log P(x) - \log Q(x)| \le \log(1+\varepsilon)$, or equivalently, $\max\left(R_\infty(P\|Q), R_\infty(Q\|P)\right) \le 1+\varepsilon$. Note that if $P \overset{\varepsilon}{\approx} Q$, then the statistical distance between P and Q is bounded by $\varepsilon/2$, i.e., $\frac{1}{2}\sum_{x\in\mathrm{Supp}(P)} |P(x) - Q(x)| \le \frac{\varepsilon}{2}$.

We collect several useful lemmas on discrete Gaussian distributions below that will be used throughout the paper.

Lemma 1 ([AGHS13, Lemma 3]). *For any $\varepsilon \in (0,1)$, a lattice $\Lambda \subseteq \mathcal{R}^m, \mathbf{c} \in K_{\mathbb{R}}^m$, and $\sigma \ge \eta_\varepsilon(\Lambda)$, then* $\Pr[\|\mathbf{x}-\mathbf{c}\| \ge \sigma\sqrt{md} : \mathbf{x} \leftarrow \mathcal{D}_{\Lambda,\sigma,\mathbf{c}}] \le \frac{1+\varepsilon}{1-\varepsilon}\cdot 2^{-md}$.

Lemma 2 ([Pei06, Lemma 2.11]). *For any n-dimensional full rank lattice Λ, center $\mathbf{c} \in \mathbb{R}^n$, positive $\varepsilon > 0$, and $s \ge 2\eta_\varepsilon(\Lambda)$, and for every $\mathbf{x} \in \Lambda$, we have*

$$D_{\Lambda,s,\mathbf{c}}(\mathbf{x}) \le \frac{1+\varepsilon}{1-\varepsilon}\cdot 2^{-n} .$$

In particular, for $\varepsilon < \frac{1}{3}$, the min-entropy of $D_{\Lambda,s,\mathbf{c}}$ is at least $n-1$.

Lemma 3 ([MP12, Lemma 2.8]). *Let $\Lambda \subseteq \mathbb{R}^n$ be a full-rank lattice and $s > 0$. For $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda,s}$ and every $\mathbf{y} \in \mathbb{R}^n$, we have*

$$\mathbb{E}\left[e^{\langle \mathbf{y},\mathbf{x}\rangle}\right] \le \exp\left(\frac{s^2\|\mathbf{y}\|^2}{4\pi}\right) .$$

Lemma 4 ([HKZ12, Theorem 2.1]). *Let $\mathbf{x} \in \mathbb{R}^n$ be a random vector satisfying*

$$\mathbb{E}\left[e^{\langle \mathbf{y},\mathbf{x}\rangle}\right] \le \exp\left(\frac{s^2\|\mathbf{y}\|^2}{2}\right) \qquad \text{for all } \mathbf{y} \in \mathbb{R}^n,$$

where $s > 0$. Then, for every $t > 0$,

$$\Pr\left[\|\mathbf{x}\|^2 > s^2\left(n + 2\sqrt{nt} + 2t\right)\right] \le e^{-t}.$$

Lemma 5 ([GPV08, Theorem 3.1]). *For any full-rank lattice $\Lambda(\mathbf{B})$ in $\mathbb{R}^m$ and $\varepsilon > 0$,*

$$\eta_\varepsilon(\Lambda) \le \sqrt{\frac{\log(2m(1+1/\varepsilon))}{\pi}}\|\tilde{\mathbf{B}}\| .$$

Lemma 6 ([GPV08, Corollary 2.8]). *Let Λ, Λ' be n-dimensional lattices, with $\Lambda' \subseteq \Lambda$. Then for any $\varepsilon \in (0, \frac{1}{2})$, any $s \ge \eta_\varepsilon(\Lambda')$, and any $\mathbf{c} \in \mathbb{R}^n$, the distribution of $\mathcal{D}_{\Lambda,s,\mathbf{c}} \bmod \Lambda'$ is within statistical distance at most 2ε of uniform over $\Lambda \bmod \Lambda'$.*

Lemma 7 (Adapted from [GMPW20, Lemma 2.6]). *For any $\varepsilon \in (0,1)$ defining $\varepsilon' = 2\varepsilon/(1-\varepsilon)$, $\sigma > 0$, lattice coset $\Lambda + \mathbf{a} \subseteq K_{\mathbb{R}}^m$, and matrix $\mathbf{T} \in K^{k\times m}$ such that $\phi_{\mathbf{M}}(\mathbf{T})$ has full row rank, $\ker(\mathbf{T})$ is a Λ-rational subspace and $\eta_\varepsilon(\Lambda \cap \ker(\mathbf{T})) \le \sigma$, we have*

$$\mathbf{T} \cdot \mathcal{D}_{\Lambda+\mathbf{a},\sigma} \overset{\varepsilon'}{\approx} \mathcal{D}_{\mathbf{T}\Lambda+\mathbf{T}\mathbf{a},\sigma^2\mathbf{T}\mathbf{T}^\dagger} \ .$$

Lemma 8 (Adapted from [CATZ24, Lemma 11]). *For any constant $\varepsilon \in (0,1)$, $\sigma_r > 0$, $1 \le k \le t-1$, full rank $\mathcal{R}$-lattice $\Lambda \subseteq \mathcal{R}^m$ with $2\eta_\varepsilon(\Lambda) \le \sigma_r$, arbitrary elements $\mathbf{s}_1, \ldots, \mathbf{s}_t \in \mathcal{R}^m$. Let $\mathbf{r}_1, \ldots, \mathbf{r}_t$ be independent samples with $\mathbf{r}_i \leftarrow \mathcal{D}_{\Lambda+\mathbf{s}_i,\sigma_r}$ and $\mathbf{P} = \begin{pmatrix} \mathbf{I}_k & \mathbf{0} \\ 1,1,\ldots,1 \end{pmatrix} \in \mathcal{R}^{(k+1)\times t}$ and $\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{x} \end{pmatrix} := \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_{k+1} \end{pmatrix} = (\mathbf{P} \otimes \mathbf{I}_m) \begin{pmatrix} \mathbf{r}_1 \\ \vdots \\ \mathbf{r}_t \end{pmatrix}$. Denote the joint distribution of $\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{x} \end{pmatrix}$ as $\mathcal{D}$. Then,*

$$\mathcal{D} \overset{\varepsilon''_{t,k}}{\approx} \mathcal{D}_{(\mathbf{P}\otimes\mathbf{I}_m)\Lambda^t+\mathbf{c},\Sigma\otimes\mathbf{I}_m},$$

where $\varepsilon''_{t,k} = \frac{2((1+\varepsilon)^{t-k}-1)}{2-(1+\varepsilon)^{t-k}}$, Λ^t is the t-th direct sum of Λ, which is a tm-dimensional lattice over $\mathcal{R}$, $\Sigma = \sigma_r^2\mathbf{P}\mathbf{P}^\dagger \in K_{\mathbb{R}}^{(k+1)\times(k+1)}$ is positive definite, and $\mathbf{c} = (\mathbf{P}\otimes\mathbf{I}_m) \cdot \begin{pmatrix} \mathbf{s}_1 \\ \vdots \\ \mathbf{s}_t \end{pmatrix} \in \mathcal{R}^{(k+1)m}$.

Moreover, denote $\mathcal{D}_1$ as the marginal distribution of $\mathbf{x}$ and $\mathcal{D}_{2|\mathbf{x}_0}$ as the distribution of $\mathbf{y}_1$ conditioning on $\mathbf{x} = \mathbf{x}_0 := (\mathbf{x}_0[1], \ldots, \mathbf{x}_0[k])^\top$, for any $\mathbf{x}_0 \in \Lambda^k + \mathbf{a}_1$, and we have

$$\mathcal{D}_1 \overset{\varepsilon''_{t,k}}{\approx} \mathcal{D}_{\Lambda^k+\mathbf{a}_1,\sigma_r^2\Sigma_{22}}, \quad \mathcal{D}_{2|\mathbf{x}_0} \overset{\varepsilon''_{t,k}}{\approx} \mathcal{D}_{\Lambda+\mathbf{s}_1,V_{11}^{-1}\cdot\mathbf{I}_m,-V_{11}^{-1}\cdot(\mathbf{V}_{21}\otimes\mathbf{I}_m)\cdot\mathbf{x}_0} \ ,$$

where $\Sigma = \begin{pmatrix} \sigma_r^2 & \sigma_r^2\mathbf{e}_k^\top \\ \sigma_r^2\mathbf{e}_k & \sigma_r^2\Sigma_{22} \end{pmatrix}$, $\Sigma^{-1} := \begin{pmatrix} V_{11} & \mathbf{V}_{21} \\ \mathbf{V}_{12} & \mathbf{V}_{22} \end{pmatrix}$, $\mathbf{a}_1 = \begin{pmatrix} \mathbf{s}_2 \\ \vdots \\ \mathbf{s}_k \\ \sum_{i\in[t]}\mathbf{s}_i \end{pmatrix} \in \mathcal{R}^{km}$, where $\Sigma_{22} = \mathbf{P}_1\mathbf{P}_1^\dagger$, $\mathbf{P}_1 = \begin{pmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ 1,1,\ldots,1 \end{pmatrix} \in \mathcal{R}^{k\times t}$, $V_{11} = \frac{1}{\sigma_r^2}\cdot(1+\frac{1}{t-k})$, $\mathbf{V}_{21} = \frac{1}{\sigma_r^2}\cdot(\frac{1}{t-k},\ldots,\frac{1}{t-k},-\frac{1}{t-k})$.

The proof of Lemma 8 is deferred to Appendix A.

Lemma 9 ([MP13, Lemma 2.6]). *For any lattice $\Lambda_1, \Lambda_2 \subseteq \mathcal{R}^m$, let $\tilde{\Lambda}_2$ be the projection of Λ_2 orthogonal to $\mathsf{Span}(\Lambda_1)$, and we have*

$$\eta_\varepsilon(\Lambda_1 + \Lambda_2) \le \max\{\eta_{\varepsilon_1}(\Lambda_1), \eta_{\varepsilon_2}(\tilde{\Lambda}_2)\},$$

where $\varepsilon = (1+\varepsilon_1)(1+\varepsilon_2) - 1$.

Lemma 10 ([CATZ24, Lemma 13]). *For any real value $\varepsilon > 0$, a $\mathcal{R}$- lattice $\Lambda \subseteq \mathcal{R}^m$ and a vector $\mathbf{v} \in K^n$, we have*

$$\eta_\varepsilon(\mathbf{v} \otimes \Lambda) \le \sqrt{\sigma_{\max}(\mathbf{v}^\dagger\mathbf{v})} \cdot \eta_\varepsilon(\Lambda) \ .$$

Moreover, we have $\sqrt{\sigma_{\max}(\mathbf{v}^\dagger\mathbf{v})} \le \|\mathbf{v}\|_1$.

3.4 Lattice Assumptions

We recall the module short integer solution (MSIS) problem, its Hermite normal form variants, and the module learning with errors (MLWE) problem, as defined in Fig. 2. We also introduce a variant of MSIS, denoted the HNF-MSIS′ problem. Recall that MSIS can be reduced to its Hermite normal form variants [Pei15]. In addition, we show that the HNF-MSIS′ problem can be reduced to the HNF-MSIS problem together with the MLWE problem.

Game $\mathsf{MSIS}^{\mathcal{A}}_{q,d,n,m,\beta_{\mathsf{sis}}}$:	Game $\mathsf{MLWE}^{\mathcal{A}}_{q,d,n,m,\beta_{\mathsf{lwe}}}$:
$\mathbf{A} \leftarrow \mathcal{R}_q^{n\times m}$ $\mathbf{x} \leftarrow \mathcal{A}(\mathbf{A})$ Return $(0 < \|\mathbf{x}\| \le \beta_{\mathsf{sis}} \wedge \mathbf{A}\mathbf{x} = \mathbf{0})$	$\mathbf{A}' \leftarrow \mathcal{R}_q^{n\times(m-n)}, \mathbf{A} := [\mathbf{I}_n \mid \mathbf{A}']$ $\mathbf{s} \leftarrow \mathscr{B}^m_{\beta_{\mathsf{lwe}}}; \mathbf{t}_0 \leftarrow \mathbf{A}\mathbf{s}; \mathbf{t}_1 \leftarrow \mathcal{R}_q^n$ $b \leftarrow \{0,1\}$ $\bar{b} \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{t}_b)$ Return $(\bar{b} = b)$
Game $\mathsf{HNF\text{-}MSIS}^{\mathcal{A}}_{q,d,n,m,\beta_{\mathsf{sis}}}$:	Game $\mathsf{HNF\text{-}MSIS}'^{\mathcal{A}}_{q,d,n,k,\ell,\beta_{\mathsf{sis}},\beta_{\mathsf{lwe}}}$:
$\mathbf{A}' \leftarrow \mathcal{R}_q^{n\times(m-n)}, \mathbf{A} := [\mathbf{I}_n \mid \mathbf{A}']$ $\mathbf{x} \leftarrow \mathcal{A}(\mathbf{A})$ Return $(0 < \|\mathbf{x}\| \le \beta_{\mathsf{sis}} \wedge \mathbf{A}\mathbf{x} = \mathbf{0})$	$\mathbf{A}_0 \leftarrow \mathcal{R}_q^{n\times k}, \mathbf{R} \leftarrow \mathscr{B}^{k\times\ell}_{\beta_{\mathsf{lwe}}}, \mathbf{E} \leftarrow \mathscr{B}^{n\times\ell}_{\beta_{\mathsf{lwe}}}$ $\mathbf{A} := [\mathbf{I}_n \mid \mathbf{A}_0 \mid \mathbf{A}_0\mathbf{R} + \mathbf{E}] \in \mathcal{R}_q^{n\times m}$, $m = k + n + \ell$ $\mathbf{x} \leftarrow \mathcal{A}(\mathbf{A})$ Return $(0 < \|\mathbf{x}\| \le \beta_{\mathsf{sis}} \wedge \mathbf{A}\mathbf{x} = \mathbf{0})$

Fig. 2. The (HNF-)MSIS, MLWE, HNF-MSIS′ problems, where $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$, $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$ and $\mathscr{B}_\beta := \{x \in \mathcal{R} : \|x\|_\infty \le \beta\}$.

The advantage of $\mathcal{A}$ for the (HNF-)MSIS problem is defined as

$$\mathrm{Adv}^{(\mathsf{hnf\text{-}})\mathsf{msis}}_{q,d,n,m,\beta_{\mathsf{sis}}}(\mathcal{A}) := \Pr\left[(\mathsf{HNF\text{-}})\mathsf{MSIS}^{\mathcal{A}}_{q,d,n,m,\beta_{\mathsf{sis}}} = 1\right].$$

The advantage of $\mathcal{A}$ for the MLWE problem is defined as

$$\mathrm{Adv}^{\mathsf{mlwe}}_{q,d,n,m,\beta_{\mathsf{lwe}}}(\mathcal{A}) := |\Pr\left[\mathsf{MLWE}^{\mathcal{A}}_{q,d,n,m,\beta_{\mathsf{lwe}}} = 1\right] - \frac{1}{2}|.$$

The advantage of $\mathcal{A}$ for the HNF-MSIS′ problem is defined as

$$\mathrm{Adv}^{\mathsf{hnf\text{-}msis'}}_{q,d,n,k,\ell,\beta_{\mathsf{sis}},\beta_{\mathsf{lwe}}}(\mathcal{A}) := \Pr\left[\mathsf{HNF\text{-}MSIS}'^{\mathcal{A}}_{q,d,n,k,\ell,\beta_{\mathsf{sis}},\beta_{\mathsf{lwe}}} = 1\right].$$

Lemma 11 (Hardness of HNF-MSIS′). *For any adversary $\mathcal{A}$ against the* HNF-MSIS′ *game, there exist an* MSIS *adversary $\mathcal{B}$ and an* MLWE *adversary $\mathcal{C}$ running in time roughly that of $\mathcal{A}$, such that*

$$\mathsf{Adv}^{\mathsf{hnf\text{-}msis'}}_{q,d,n,k,\ell,\beta_{\mathsf{sis}},\beta_{\mathsf{lwe}}}(\mathcal{A}) \le \mathsf{Adv}^{\mathsf{hnf\text{-}msis}}_{q,d,n,m,\beta_{\mathsf{sis}}}(\mathcal{B}) + 2\ell \cdot \mathsf{Adv}^{\mathsf{mlwe}}_{q,d,n,n+k,\beta_{\mathsf{lwe}}}(\mathcal{C}).$$

Proof. Given an HNF-MSIS′ instance of the form $\mathbf{A} = [\mathbf{I}_n \mid \mathbf{A}_0 \mid \mathbf{A}_0\mathbf{R} + \mathbf{E}]$, we can replace the term $\mathbf{A}_0\mathbf{R} + \mathbf{E}$ with a uniformly random matrix $\mathbf{A}' \in \mathcal{R}_q^{n\times\ell}$. The resulting matrix $\mathbf{A} = [\mathbf{I}_n \mid \mathbf{A}_0 \mid \mathbf{A}']$ is then a valid instance of HNF-MSIS. Consequently, we obtain

$$\left|\mathsf{Adv}^{\mathsf{hnf\text{-}msis'}}_{q,d,n,k,\ell,\beta_{\mathsf{sis}},\beta_{\mathsf{lwe}}}(\mathcal{A}) - \mathsf{Adv}^{\mathsf{hnf\text{-}msis}}_{q,d,n,m,\beta_{\mathsf{sis}}}(\mathcal{B})\right| \le 2\ell \cdot \mathsf{Adv}^{\mathsf{mlwe}}_{q,d,n,n+k,\beta_{\mathsf{lwe}}}(\mathcal{C}).$$

This yields the desired result, where $m = n + k + \ell$.

Similarly, for any fixed matrix $\mathbf{M} \in \mathcal{R}_q^{n\times\ell}$, if an HNF-MSIS′ instance $\mathbf{A}$ has form $[\mathbf{I}_n \mid \mathbf{A}_0 \mid \mathbf{A}_0\mathbf{R} + \mathbf{E} + \mathbf{M}]$ the same reduction also applies. □

3.5 Rényi divergence

We define the notion of Rényi divergence between two distributions P, Q which we will use in the analysis of the scheme. For a discrete distribution P, we denote the support of P as $\mathrm{Supp}(P) := \{x : P(x) > 0\}$.

Definition 1 (Rényi Divergence). *Let P, Q be two discrete probability distributions such that* $\mathrm{Supp}(P) \subseteq \mathrm{Supp}(Q)$ *and $\alpha \in [1, +\infty]$. We define the Rényi Divergence of order α, for $\alpha \in (1, \infty)$, as*

$$R_\alpha(P\|Q) := \left(\sum_{x\in\mathrm{Supp}(P)} \frac{P(x)^\alpha}{Q(x)^{\alpha-1}}\right)^{\frac{1}{\alpha-1}}.$$

For $\alpha = 1$ and $\alpha = \infty$, we define

$$R_1(P\|Q) := \exp\left(\sum_{x\in\mathrm{Supp}(P)} P(x)\log\frac{P(x)}{Q(x)}\right),\ R_\infty(P\|Q) := \max_{x\in\mathrm{Supp}(P)}\frac{P(x)}{Q(x)}.$$

Below, we collect several useful lemmas on Rényi divergence that will be used in the analysis of the scheme.

Lemma 12 ([ASY22, Lemma 2.27]). *Let $\alpha \in [1,\infty]$ and P, Q be discrete probability distributions with $\mathrm{Supp}(P) \subseteq \mathrm{Supp}(Q)$. Then, the following properties hold:*

- ***Log Positivity:*** *$R_\alpha(P\|Q) \geq R_\alpha(P\|P) = 1$.*
- ***Data Processing Inequality:*** *$R_\alpha(P^f\|Q^f) \leq R_\alpha(P\|Q)$ for any function f, where P^f (and Q^f) denotes the distribution which samples $x \leftarrow P$ ($x \leftarrow Q$) and outputs $f(x)$.*
- ***Probability Preservation:*** *Let $E \subseteq \mathrm{Supp}(Q)$ be an arbitrary event. Then, for $\alpha \in (1,\infty)$,*

$$\Pr_{x\leftarrow P}[E]^{\alpha/(\alpha-1)}/R_\alpha(P\|Q) \leq \Pr_{x\leftarrow Q}[E].$$

- ***Weak Triangle Inequality:*** *Let P_1, P_2, P_3 be three probability distributions where $\mathrm{Supp}(P_1) \subseteq \mathrm{Supp}(P_2) \subseteq \mathrm{Supp}(P_3)$. Then, we have*

$$R_\alpha(P_1\|P_3) \leq \begin{cases} R_\alpha(P_1\|P_2)\cdot R_\infty(P_2\|P_3) \\ R_\infty(P_1\|P_2)^{\frac{\alpha}{\alpha-1}}\cdot R_\alpha(P_2\|P_3) \quad \text{if } \alpha\in(1,\infty) \end{cases}.$$

Lemma 13 ([Ros20, Proposition 2]). *Let P and Q denote two distributions of a sequence of random variables $(X_1,\ldots,X_n)$. For $1 \leq i \leq n$, denote $P_{i|x_{[i-1]}}$ (resp. $Q_{i|x_{[i-1]}}$) as the conditional distribution of X_i given $(X_1,\ldots,X_{i-1}) = (x_1,\ldots,x_{i-1})$. Then, for any $\alpha > 1$,*

$$R_\alpha(P\|Q) \leq \prod_{i\in[n]}\max_{x_{[i-1]}} R_\alpha(P_{i|x_{[i-1]}}\|Q_{i|x_{[i-1]}}).$$

Lemma 14 ([TT15, Lemma 5]). *For any m-dimensional lattice $\Lambda \subseteq \mathcal{R}^m$, $\sigma \in \mathbb{R}$ and $\mathbf{c}, \mathbf{c}' \in \Lambda$, then*

$$R_\alpha(\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}\|\mathcal{D}_{\Lambda,\sigma,\mathbf{c}'}) \leq \exp\left(\frac{\alpha\pi\|\mathbf{c}-\mathbf{c}'\|^2}{\sigma^2}\right).$$

If $\mathbf{c}, \mathbf{c}' \in K_{\mathbb{R}}^m$, $\sigma \geq \eta_\varepsilon(\Lambda)$, then

$$R_\alpha(\mathcal{D}_{\Lambda,\sigma,\mathbf{c}}\|\mathcal{D}_{\Lambda,\sigma,\mathbf{c}'}) \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{\frac{\alpha}{\alpha-1}}\exp\left(\frac{\alpha\pi\|\mathbf{c}-\mathbf{c}'\|^2}{\sigma^2}\right).$$

4 Our construction

We first provide the formal syntax and security definitions of partially non-interactive threshold signatures adapted from [BTZ22], then describe our construction in Section 4.2, analyze its security in Section 4.3, and finally discuss the parameter selection and concrete efficiency in Section 4.6.

4.1 Syntax and security definitions

Syntax. T-out-of-N partially non-interactive threshold signature is a tuple of algorithms $\mathsf{TS} = (\mathsf{Setup}, \mathsf{Sign}_1, \mathsf{Sign}_2, \mathsf{Combine}, \mathsf{Ver})$ that act as follows.

- $\mathsf{Setup}(1^\kappa) \to (\mathsf{pk}, (\mathsf{st}_i)_{i\in[N]})$: The setup algorithm takes the security parameter as input and outputs a public key pk and an initial state st_i for each signer i. Similar to prior work, we assume an ideal key generation process and do not model distributed key generation.
- $\mathsf{Sign}_1(i, \mathsf{st}_i) \to (pm_i, \mathsf{st}_i)$: The first-round signing algorithm takes a signer index $i \in [N]$ and the corresponding state of signer i as input and outputs an offline nonce pm_i and the updated state.

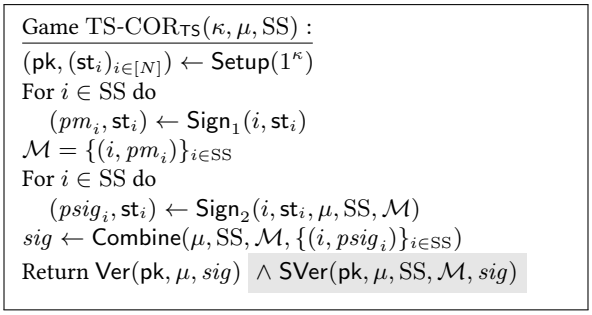

Fig. 3. The TS-COR game for a threshold signature scheme TS for N signers and threshold T. The highlighted part only applies to TS schemes with the extended verification algorithm SVer, which is used to define strong unforgeability of TS.

- $\mathsf{Sign}_2(i, \mathsf{st}_i, \mu, \mathrm{SS}, \mathcal{M} = \{(j, pm_j)\}_{j\in\mathrm{SS}}) \to (psig_i, \mathsf{st}_i)$: The second-round signing algorithm takes a signer index i, the corresponding state st_i, a message μ to be signed, a set SS of participating signers, and a set of first-round nonces $\mathcal{M} = \{(j, pm_j)\}_{j\in\mathrm{SS}}$ as input and outputs a partial signature $psig_i$ and the updated state st_i. The algorithm may output $\perp$ if the inputs are invalid.
- $\mathsf{Combine}(\mu, \mathrm{SS}, \mathcal{M}, \{(i, psig_i)\}_{i\in\mathrm{SS}}) \to sig$: The algorithm takes the second-round input $(\mu, \mathrm{SS}, \mathcal{M})$ and the set of partial signatures $\{(i, psig_i)\}_{i\in\mathrm{SS}}$ as input and outputs the final signature.
- $\mathsf{Ver}(\mathsf{pk}, \mu, sig) \to 1/0$: The verification algorithm takes the public key pk, a message μ, and a signature sig as input and outputs 1 (resp. 0) if the signature is valid (resp. invalid).

The honest signing process can be described by the correctness game TS-COR defined in Figure 3. In the first (offline) round, each signer i runs the Sign_1 algorithm to generate a first-round nonce pm_i and sends it to a coordinator. The coordinator can be one of the signers. In the second (online) round, to sign a message μ with the set of involved signers SS, the coordinator sends $(\mu, \mathrm{SS}, \mathcal{M} = \{(j, pm_j)\}_{j\in\mathrm{SS}})$ to each signer $i \in \mathrm{SS}$. Each signer i runs the Sign_2 algorithm and sends the partial signature $psig_i$ to the coordinator. Finally, the coordinator runs Combine to obtain the final signature. We say that the scheme TS is correct with correctness error at most $\varepsilon_{\mathsf{cor}}$ if for any $\mu \in \{0,1\}^*$ and $\mathrm{SS} \subseteq [N]$ with $|\mathrm{SS}| \geq T$, we have $\Pr[\text{TS-COR}_{\mathsf{TS}}(\kappa, \mu, \mathrm{SS}) = 1] \geq 1 - \varepsilon_{\mathsf{cor}}$.

Unforgeability. Unforgeability guarantees that the adversary cannot forge a signature for a message that is *not considered signed* even if the adversary can corrupt the coordinator and fewer than T signers. As pointed out by Bellare et al. [BTZ22], there are different ways to define whether a message is *considered signed*, which leads to different security levels. Here, we adopt the strongest security notion, TS-UF-4 from [BTZ22] and extend it to the adaptive setting, referred to as adp-TS-UF-4. A similar definition is also used in [ZT25].

The adp-TS-UF-4 game is defined in Figure 4. In adp-TS-UF-4, a message μ is *considered signed* if and only if the adversary made a second-round signing query $(\mu, \mathrm{SS}, \mathcal{M})$ for some $\mathrm{SS} \subseteq [N]$ and $\mathcal{M}$ such that all honest signers in SS responded to the query, i.e., each honest signer received the query and the Sign_2 algorithm did not return $\perp$.

The game is defined in the random oracle model, where the hash function used by the algorithms of TS is modeled as a random oracle. The advantage of an adversary $\mathcal{A}$ playing the adp-TS-UF-4 game is defined as
$\mathsf{Adv}^{\mathsf{adp\text{-}ts\text{-}uf\text{-}4}}_{\mathsf{TS}}(\mathcal{A}, \kappa)$
$:= \Pr[\text{adp-TS-UF-4}^{\mathcal{A}}_{\mathsf{TS}}(\kappa) = 1]$.

Strong unforgeability. Further, we consider strong unforgeability, first proposed by Bellare et al. [BTZ22] in the setting of threshold signatures. Informally, this notion guarantees that an adversary cannot produce a signature that is not considered issued, analogous to strong unforgeability for standard signature schemes.

To define strong unforgeability of a scheme TS, we require an *extended verification* algorithm SVer defined in TS that checks whether a signature sig is *associated* with a second round query $(\mu, \mathrm{SS}, \mathcal{M})$, i.e., the signature is expected to be issued given the query. More formally, the algorithm SVer, taking a second round query $(\mu, \mathrm{SS}, \mathcal{M})$ and a signature sig as input, outputs 1 if and only if the signature is *associated* with the query. In addition, we require that for any honest signing execution, SVer outputs 1 on the resulting signature and the corresponding query, which is formally captured by the correctness game (Figure 3).

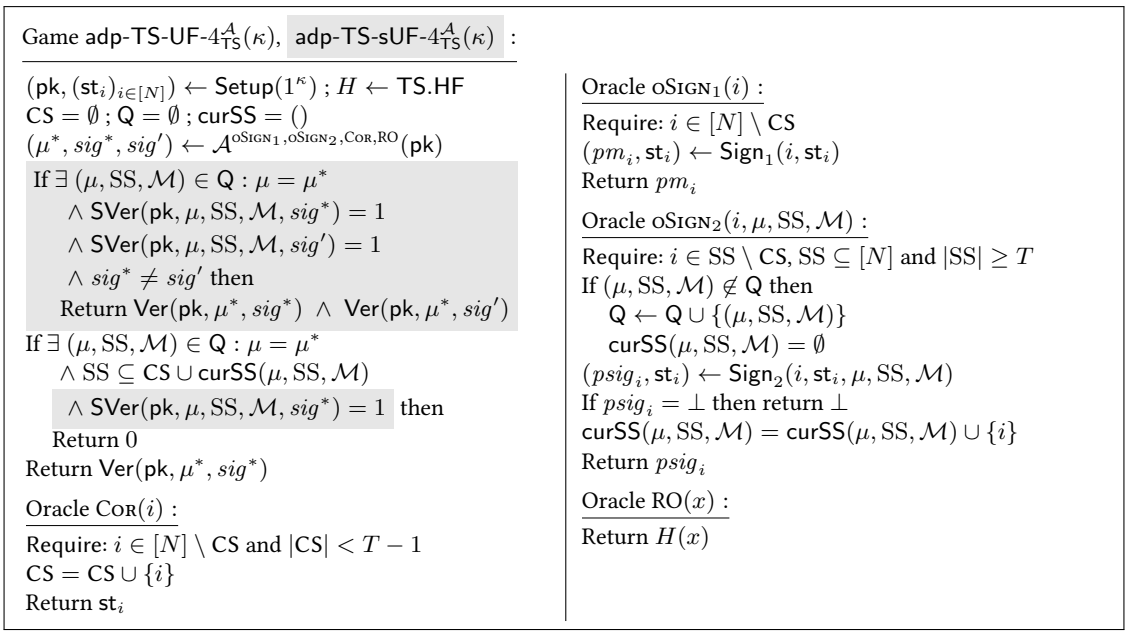

Fig. 4. The adaptive unforgeability games adp-TS-UF-4 and adp-TS-sUF-4 for a threshold signature scheme TS for N signers and threshold T, where adp-TS-UF-4 contains all but the highlight box and adp-TS-sUF-4 contains everything. The games are defined in the random oracle model, where the hash function used by the algorithms of TS is modeled as a random oracle, and TS.HF denotes the space of the hash functions from which the random oracle is drawn.

The strong unforgeability game adp-TS-sUF-4 is defined in Figure 4. In adp-TS-sUF-4, a signature sig is considered issued for a message μ if and only if the adversary made a second-round signing query $(\mu, \mathrm{SS}, \mathcal{M})$ for some $\mathrm{SS} \subseteq [N]$ and $\mathcal{M}$ such that (1) sig is valid and *associated* with the query and (2) all honest signers in SS responded to the query. In addition to guaranteeing that the adversary cannot forge a signature that is not considered issued, adp-TS-sUF-4 guarantees that, even if there is a signature *associated* with some query considered issued to the adversary, the adversary cannot forge two *distinct* valid signatures that are *associated* with that query. This latter property is captured by the first highlight box in Figure 4. The advantage of an adversary $\mathcal{A}$ playing the adp-TS-sUF-4 game is defined as $\mathsf{Adv}^{\text{adp-ts-suf-4}}_{\mathsf{TS}}(\mathcal{A}, \kappa) := \Pr[\text{adp-TS-sUF-4}^{\mathcal{A}}_{\mathsf{TS}}(\kappa) = 1]$.

Remark 1 (Comparison to strong unforgeability from [BTZ22] and one-more unforgeability (OMUF) [NT24]). Note that our strong unforgeability notion (without adaptive corruption) is slightly weaker than the original notion from [BTZ22] in the sense that we only require that it is *computationally* hard to find two distinct valid signatures that are associated with the same query, while, in [BTZ22], it guarantees that there is at most one valid signature that is associated with any query. This relaxation is necessary for our construction, since the underlying randomized hash-and-sign scheme may admit multiple distinct valid short preimages for the same signing query, and hence cannot satisfy such an information-theoretic uniqueness requirement; nevertheless, finding two such signatures is computationally hard under the MSIS assumption.

Despite this relaxation, our security notion still implies one-more unforgeability (OMUF), another notion for capturing strong unforgeability first proposed in [NT24] and also used in [KLL25, ABHR25], which guarantees that for any message μ, the adversary cannot forge one more signatures than the number of signing queries for which a signature of μ is considered issued. To see our security notion implies OMUF, suppose there is an adversary that can break OMUF. Then, the adversary can either forge a signature for a signing query where a signature is not considered issued, or forge two distinct signatures that are associated with the same query, which breaks our strong unforgeability.

4.2 Construction

We present our threshold signature scheme in Figure 5, and refer to Section 2 for a high-level overview. Relative to the scheme described there, we introduce a signer-specific state st_i to facilitate recording previously used $\mathbf{w}_i$.

$\underline{\mathsf{Setup}(1^\kappa):}$
$(\mathbf{A},\mathbf{T}) \leftarrow \mathsf{KeyGen}(1^\kappa)$
  / satisfying $\mathbf{AT}=\mathbf{G}$
  / $\mathbf{A} \in \mathcal{R}_q^{n\times m}, \mathbf{T} \in \mathcal{R}_q^{m\times(n+\ell)}$
For $(i,j) \in [N]\times[N]$ do
  $\mathsf{seed}_{i,j} \leftarrow \{0,1\}^{2\kappa}$
$\tilde{\mathbf{T}}_1,\ldots,\tilde{\mathbf{T}}_{T-1} \leftarrow \mathcal{R}_q^{m\times(n+\ell)}$
For $i \in [N]$ do
  $\mathbf{T}_i = \mathbf{T} + \sum_{j=1}^{T-1} \tilde{\mathbf{T}}_j i^j$
  $\mathsf{st}_i.\mathsf{sk} = (\mathbf{T}_i, (\mathsf{seed}_{i,j}, \mathsf{seed}_{j,i})_{j\in[N]})$
  $\mathsf{st}_i.\mathsf{mapPP} = () ; \mathsf{st}_i.\mathsf{validPP} = ()$
Return $\mathsf{pk} = \mathbf{A}, (\mathsf{st}_i)_{i\in[N]}$.

$\underline{\mathsf{Sign}_1(i,\mathsf{st}_i):}$
Require: $i \in [N]$
$\mathbf{r}_i \leftarrow \mathcal{D}_{\mathcal{R}^m,\sigma_r}; \mathbf{w}_i = \mathbf{A}\mathbf{r}_i$
$\mathsf{st}_i.\mathsf{mapPP}(\mathbf{w}_i) = \mathbf{r}_i$
$\mathsf{st}_i.\mathsf{validPP}(\mathbf{w}_i) = 1$
Return $\mathbf{w}_i, \mathsf{st}_i$

$\underline{\mathsf{Ver}(\mathsf{pk},\mu,(\mathbf{z},\mathsf{salt})):}$
Return $\|\mathbf{z}\| \le L_1 \wedge (\mathbf{Az} = H_1(\mu,\mathsf{salt}))$

$\underline{\mathsf{Sign}_2(i,\mathsf{st}_i,\mu,\mathrm{SS},\mathcal{M}):}$
Require: $i \in \mathrm{SS} \subseteq [N]$ and $|\mathrm{SS}| \ge T$
$(\mathbf{T}_i, (\mathsf{seed}_{i,j},\mathsf{seed}_{j,i})_{j\in[N]}) = \mathsf{st}_i.\mathsf{sk}$
$\{(j,\mathbf{w}_j)\}_{j\in\mathrm{SS}} = \mathcal{M}$
If $\mathsf{st}_i.\mathsf{validPP}(\mathbf{w}_i) \neq 1$ then return $\perp$
$\mathsf{st}_i.\mathsf{validPP}(\mathbf{w}_i) = 0$
$\mathbf{r}_i = \mathsf{st}_i.\mathsf{mapPP}(\mathbf{w}_i)$
$\mathbf{w} = \sum_{i\in\mathrm{SS}} \mathbf{w}_i$
$\mathsf{salt} = H_2(\mu,\mathrm{SS},\mathcal{M})$
$\mathsf{mask}_i \leftarrow \sum_{j\in\mathrm{SS}} H_3(\mathsf{seed}_{i,j},\mathsf{salt})$
$\mathsf{mask}'_i \leftarrow \sum_{j\in\mathrm{SS}} H_3(\mathsf{seed}_{j,i},\mathsf{salt})$
$\mathbf{z}_i = \mathbf{r}_i + L_{\mathrm{SS},i}\cdot\mathbf{T}_i\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w})$
  $+\mathsf{mask}_i - \mathsf{mask}'_i$
Return $\mathbf{z}_i, \mathsf{st}_i$

$\underline{\mathsf{Combine}(\mu,\mathrm{SS},\mathcal{M},\{(i,\mathbf{z}_i)\}_{i\in\mathrm{SS}}):}$
$\mathbf{z} = \sum_{i\in\mathrm{SS}} \mathbf{z}_i$, $\mathsf{salt} = H_2(\mu,\mathrm{SS},\mathcal{M})$
Return $(\mathbf{z},\mathsf{salt})$

$\underline{\mathsf{SVer}(\mathsf{pk},\mu,\mathrm{SS},\mathcal{M},(\mathbf{z},\mathsf{salt})):}$
Return $\mathsf{salt} = H_2(\mu,\mathrm{SS},\mathcal{M})$

Fig. 5. The lattice-based T-out-of-N threshold signature scheme Oriole. The instantiation of the algorithm KeyGen is provided in Section 4.2. We use H_i to denote $H(i,\cdot)$, where H is a specific hash function in the hash function family $\mathsf{Oriole.HF} := \{H \mid H_1 : \{0,1\}^* \to \mathcal{R}_q^n, H_2 : \{0,1\}^* \to \{0,1\}^{2\kappa}, H_3 : \{0,1\}^* \to \mathcal{R}_q^m\}$. Also, $\mathbf{G}$ denotes the gadget matrix, and $\mathbf{G}^{-1}$ refers to the deterministic inversion, both defined in Section 4.2. $L_{\mathrm{SS},i}$ denotes the Lagrange coefficient. The public key pk is implicitly given to all algorithms except KeyGen.

The mask term $\mathsf{mask}_i - \mathsf{mask}'_i$ in Figure 5 corresponds to $\mathsf{mask}_{i,\mathrm{SS},\mathsf{salt}}$ in the overview, and the partial signature $\mathbf{z}_i$ in Figure 5 corresponds to $\mathbf{z}'_i$ in the overview. The Lagrange coefficient $L_{U,j} \in \mathbb{Z}_q$ for any subset $U \subseteq [N]$ and $j \in U$ is defined as $L_{U,j} := \prod_{i\in U\setminus\{j\}} \frac{-i}{j-i}$. To define strong unforgeability of Oriole, we specify the extended verification algorithm SVer for the scheme, where a signature $(\mathbf{z},\mathsf{salt})$ is associated with a query $(\mu,\mathrm{SS},\mathcal{M})$ if and only if $H_2(\mu,\mathrm{SS},\mathcal{M}) = \mathsf{salt}$.

In the key generation algorithm KeyGen, we sample $(\mathbf{A},\mathbf{T})$ satisfying $\mathbf{AT}=\mathbf{G}$, where $\mathbf{G}$ denotes a gadget matrix defined below. Indeed, [MP12] already shows how to do this. Here, we describe a construction that yields better concrete efficiency, following [CGM19, YJW23, EEN+24].

- We first sample $\mathbf{A}_0 \leftarrow \mathcal{R}_q^{n\times k}$, $\mathbf{R} \leftarrow \mathcal{B}_{\beta_{\mathsf{lwe}}}^{k\times\ell}, \mathbf{E} \leftarrow \mathcal{B}_{\beta_{\mathsf{lwe}}}^{n\times\ell}$, where $\mathcal{B}_\beta := \{x \in \mathcal{R} : \|x\|_\infty \le \beta\}$, and define $\mathbf{A} = [\mathbf{I}_n \mid \mathbf{A}_0 \mid \mathbf{A}_0\mathbf{R}+\mathbf{E}+\mathbf{G}_1] \in \mathcal{R}_q^{n\times m}$, where $\mathbf{G}_1 = \lceil q/p^t \rceil \cdot [\mathbf{I}_n \mid p\mathbf{I}_n \mid \cdots \mid p^{t-1}\mathbf{I}_n] \in \mathcal{R}_q^{n\times tn}$, and $m = k+n+\ell$, $\ell = tn$.
- Next, we define $\mathbf{G} := [\mathbf{I}_n \mid \mathbf{G}_1]$ so that $\mathbf{A}\cdot\overbrace{\begin{bmatrix}\mathbf{I}_n & -\mathbf{E}\\ \mathbf{0} & -\mathbf{R}\\ \mathbf{0} & \mathbf{I}_\ell\end{bmatrix}}^{\mathbf{T}} = \mathbf{G}$.
- Given $\mathbf{u} \in \mathcal{R}_q^n$, we can express $\mathbf{u}$ uniquely as $\mathbf{u} = \mathbf{u}_0 + \mathbf{G}_1\mathbf{u}_1$ such that $\|\mathbf{u}_0\|_\infty < \lceil q/p^t \rceil$, $\|\mathbf{u}_1\|_\infty < p$. We define $\mathbf{G}^{-1}(\mathbf{u}) := \begin{bmatrix}\mathbf{u}_0\\ \mathbf{u}_1\end{bmatrix}$.

We summarize all the parameters used in the protocol in Figure 6. For convenience in the analysis of the scheme, we denote $\rho_0(\varepsilon) := \eta_\varepsilon(\Lambda_q^\perp(\mathbf{A}))$ and $\rho_1 := \max_{\mathbf{u}\in\mathcal{R}_q^n} \|\mathbf{TG}^{-1}(\mathbf{u})\|$. We bound $\rho_0(\varepsilon)$ and ρ_1 in Section 4.5. The following theorem establishes the correctness of our scheme.

Theorem 1 (correctness). *For any parameter selection as specified in Figure 6 and any $\varepsilon \in (0,1)$, given $\sigma_r \ge \max\{2\rho_0(\varepsilon),\rho_1\}$ and $L_1 \ge (\sqrt{mdN}+1)\sigma_r$,* Oriole *(Figure 5) is correct with a correctness error $\varepsilon_{\mathsf{cor}} \le 2^{-md}$.*

Notations	Description
κ	Security parameter.
N	Number of signers.
T	Threshold for signing.
d	$d \geq 2\kappa$, power of two defining the ring $\mathcal{R}$.
q	Prime modulus.
n	Number of rows of the public matrix $\mathbf{A}$.
k	Auxiliary dimension used in the HNF-style form of $\mathbf{A}$.
t	Gadget-decomposition length.
ℓ	Number of gadget columns, where $\ell = tn$.
m	Number of columns of $\mathbf{A}$, where $m = k + n + \ell$.
p	Gadget base used in $\mathbf{G}_1$ and in the deterministic inversion $\mathbf{G}^{-1}$.
σ_r	Standard deviation used to sample the nonce $\mathbf{r}_i \leftarrow D_{\mathcal{R}^m,\sigma_r}$.
L_1	Euclidean norm bound for a valid signature $\mathbf{z}$.
β_{lwe}	Coefficient bound of the MLWE noise used to sample the matrices $\mathbf{R}$ and $\mathbf{E}$ in KeyGen.
β_{sis}	Norm bound in the underlying HNF-MSIS′ assumption.
$\rho_0(\varepsilon)$	$\eta_\varepsilon(\Lambda_q^\perp(\mathbf{A}))$.
ρ_1	$\max_{\mathbf{u}\in\mathcal{R}_q^n} \|\mathbf{T}\mathbf{G}^{-1}(\mathbf{u})\|$.

Fig. 6. Parameters and notations for Oriole. In particular, $\rho_0(\varepsilon)$ and ρ_1 are convenient notations used in the proof, instead of parameters for the scheme.

Proof. Fix any message $\mu \in \{0,1\}^*$ and any signing set $\mathrm{SS} \subseteq [N]$ satisfying $|\mathrm{SS}| \geq T$.

In the correctness experiment, each signer $i \in \mathrm{SS}$ invokes Sign_1 exactly once, obtaining

$$\mathbf{r}_i \leftarrow \mathcal{D}_{\mathcal{R}^m,\sigma_r}\ , \qquad \mathbf{w}_i = \mathbf{A}\mathbf{r}_i\ .$$

The algorithm then sets $\mathsf{st}_i.\mathsf{validPP}(\mathbf{w}_i) = 1$, and the corresponding Sign_2 invocation uses this output exactly once. Therefore, no honest invocation of Sign_2 aborts.

From Figure 5,

$$\begin{aligned}\mathbf{z} &= \sum_{i\in\mathrm{SS}} \mathbf{r}_i + L_{\mathrm{SS},i}\cdot\mathbf{T}_i\cdot\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w}) + \mathsf{mask}_i - \mathsf{mask}_i' \\ &= \sum_{i\in\mathrm{SS}} \mathbf{r}_i + \mathbf{T}\cdot\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w})\ ,\end{aligned}$$

where the second equality uses

$$\mathbf{T} = \sum_{i\in\mathrm{SS}} L_{\mathrm{SS},i}\cdot\mathbf{T}_i, \quad \sum_{i\in\mathrm{SS}} \mathsf{mask}_i = \sum_{i\in\mathrm{SS}} \mathsf{mask}_i'\ .$$

Therefore,

$$\begin{aligned}\mathbf{A}\mathbf{z} &= \sum_{i\in\mathrm{SS}} \overbrace{\mathbf{A}\mathbf{r}_i}^{=\mathbf{w}_i} + \overbrace{\mathbf{A}\mathbf{T}}^{=\mathbf{G}}\cdot\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w}) \\ &= \mathbf{w} + (H_1(\mu,\mathsf{salt}) - \mathbf{w}) = H_1(\mu,\mathsf{salt})\ .\end{aligned}$$

Also, it is clear that $\mathsf{salt} = H_2(\mu, \mathrm{SS}, \mathcal{M})$.

We proceed to bound $\|\mathbf{z}\|$. By Lemma 3 and independence, $\phi(\sum_{i\in\mathrm{SS}}\mathbf{r}_i)$ satisfies the conditions of Lemma 4 with dimension md and $s^2 = |\mathrm{SS}|\sigma_r^2/(2\pi)$. Taking $t = md$ and using $5/(2\pi) < 1$, we obtain

$$\Pr\left[\left\|\sum_{i\in\mathrm{SS}}\mathbf{r}_i\right\| > \sqrt{md|\mathrm{SS}|}\,\sigma_r\right] \leq e^{-md} \leq 2^{-md}\ .$$

Game adp-IUF$^{\mathcal{A}}(\kappa)$:

$(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{KeyGen}(1^\kappa)$; $\mathsf{pk} = \mathbf{A}$
$H \leftarrow$ Oriole.HF
$\mathsf{Q} = \emptyset$; $\mathsf{CS} = \emptyset$
For $i \in [N]$:
  $\mathsf{st}_i.\mathsf{mapPP} = ()$
  $\mathsf{st}_i.\mathsf{validPP} = ()$
$(\mu^*, sig^*, sig') \leftarrow \mathcal{A}^{\text{oSign}_1, \text{oSign}_2, \text{RO}, \text{Cor}}(\mathsf{pk})$
$(\mathbf{z}^*, \mathsf{salt}^*) = sig^*$; $(\mathbf{z}', \mathsf{salt}') = sig'$
If $\exists\ (\mu, \mathrm{SS}, \mathcal{M}) \in \mathsf{Q} : \mu^* = \mu$
  $\wedge\ \mathsf{salt}^* = \mathsf{salt}' = H_2(\mu, \mathrm{SS}, \mathcal{M})$
  $\wedge\ sig^* \neq sig'$ then
  Return $\mathsf{Ver}(\mathsf{pk}, \mu, sig^*)\ \wedge\ \mathsf{Ver}(\mathsf{pk}, \mu, sig')$
If $\exists\ (\mu, \mathrm{SS}, \mathcal{M}) \in \mathsf{Q} : \mu^* = \mu$
  $\wedge\ \mathsf{salt}^* = H_2(\mu, \mathrm{SS}, \mathcal{M})$ then
  Return 0
Return $\mathsf{Ver}(\mathsf{pk}, \mu^*, sig^*)$

Oracle $\text{Cor}(i)$:

Require: $i \in [N] \setminus \mathsf{CS}$ and $|\mathsf{CS}| < T - 1$
$\mathsf{CS} = \mathsf{CS} \cup \{i\}$
Return st_i

Oracle $\text{oSign}_1(i)$:

Require: $i \in [N] \setminus \mathsf{CS}$
$\mathbf{r}_i \leftarrow \mathcal{D}_{\mathcal{R}^m, \sigma_r}$, $\mathbf{w}_i = \mathbf{A}\mathbf{r}_i$
$\mathsf{st}_i.\mathsf{mapPP}(\mathbf{w}_i) = \mathbf{r}_i$
$\mathsf{st}_i.\mathsf{validPP}(\mathbf{w}_i) = 1$
Return $\mathbf{w}_i$

Oracle $\text{oSign}_2(\mu, \mathrm{SS}, \mathcal{M})$:

Require: $\mathrm{SS} \subseteq [N]$ and $|\mathrm{SS}| \geq T$
$\{(i, \mathbf{w}_i)\}_{i \in \mathrm{SS}} = \mathcal{M}$
If $\exists\ i \in \mathrm{SS} \setminus \mathsf{CS}$:
  $\mathsf{st}_i.\mathsf{validPP}(\mathbf{w}_i) \neq 1$ then
  Return $\perp$
$\mathsf{Q} = \mathsf{Q} \cup \{(\mu, \mathrm{SS}, \mathcal{M})\}$
$\mathbf{w} = \sum_{i \in \mathrm{SS}} \mathbf{w}_i$
$\mathsf{salt} = H_2(\mu, \mathrm{SS}, \mathcal{M})$
For $i \in \mathrm{SS} \setminus \mathsf{CS}$:
  $\mathbf{r}_i = \mathsf{st}_i.\mathsf{mapPP}(\mathbf{w}_i)$
  $\mathsf{st}_i.\mathsf{validPP}(\mathbf{w}_i) = 0$
$\mathbf{z} = \sum_{i \in \mathrm{SS} \setminus \mathsf{CS}} \mathbf{r}_i + \mathbf{T}\mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w})$
Return $\mathbf{z}$

Oracle $\text{RO}(i, x)$:

Return $H_i(x)$

Fig. 7. The adaptive-intermediate game adp-IUF. The algorithm KeyGen is defined in Section 4.2 and $(\mathbf{A}, \mathbf{T})$ satisfies $\mathbf{A}\mathbf{T} = \mathbf{G}$, where $\mathbf{A} \in \mathcal{R}_q^{n \times m}$, $\mathbf{T} \in \mathcal{R}_q^{m \times (n+\ell)}$, and $\mathbf{G} \in \mathcal{R}_q^{n \times (n+\ell)}$ denotes the gadget matrix. The hash function is sampled uniformly from the hash function family Oriole.HF, which is defined in Figure 5.

Thus, except with probability at most 2^{-md},

$$\|\mathbf{z}\| \leq \left\| \sum_{i \in \mathrm{SS}} \mathbf{r}_i \right\| + \|\mathbf{T}\mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w})\| \leq (\sqrt{mdN} + 1)\sigma_r \leq L_1 \ ,$$

where we used $|\mathrm{SS}| \leq N$ and the parameter conditions. This proves correctness with error at most 2^{-md}.

□

4.3 Adaptive Security

We show that Oriole achieves adaptive strong unforgeability (adp-TS-sUF-4) under the MSIS and MLWE assumptions in the random oracle model. We refer to Section 2 for a proof overview.

Proof structure. The proof follows a two-step approach, similar to the prior works [ZT25, JTWZ26]. We consider an intermediate security game, referred to as the adp-IUF game (defined in Fig. 7), where-in for the second-round message corresponding to oSign_2, the adversary only sees the sum of (honest) partial signatures, i.e., $\hat{\mathbf{z}}$ in Equation (2), instead of the individual partial signatures. Note that the oracle oSign_2 now takes as input $(\mu, \mathrm{SS}, \mathcal{M})$ instead of $(i, \mu, \mathrm{SS}, \mathcal{M})$. Also, each signer no longer has a secret key.

- In the first step (Theorem 2), we reduce the adp-TS-sUF-4 game of the threshold scheme to the adp-IUF game. This step is similar to the previous proofs in [ZT25, JTWZ26] (which in turn build on [EKT24]), and uses masking in an essential way. The proof is deferred to Section 4.4.
- In the second step (Theorem 3), we establish the hardness of the adp-IUF game based on the hardness of the HNF-MSIS′ problem, which is implied by the MSIS and MLWE assumptions (Lemma 11). This step is where the main novelty lies.

Theorem 2 (From adp-TS-sUF-4 to adp-IUF). *For any parameter selection as specified in Figure 6 and any adp-TS-sUF-4 adversary $\mathcal{A}$ making at most Q_{s1} queries to the oSIGN_1 oracle, Q_{s2} queries to the oSIGN_2 oracle, and Q_h random oracle queries, there exist an adp-IUF adversary $\mathcal{B}$ making at most Q_{s1} queries to the oSIGN_1 oracle, Q_{s2} queries to the oSIGN_2 oracle, and $Q_h + Q_{s2}$ random oracle queries such that*

$$\mathsf{Adv}^{\mathsf{adp\text{-}ts\text{-}suf\text{-}4}}_{\mathsf{Oriole}}(\mathcal{A}, \kappa) \leq \mathsf{Adv}^{\mathsf{adp\text{-}IUF}}(\mathcal{B}, \kappa) + Q(Q + N^2) \cdot 2^{-2\kappa} ,$$

where $Q = Q_h + Q_{s_2}$ and N denotes the number of signers.

Theorem 3 (From adp-IUF to HNF-MSIS′). *For any parameter selection as specified in Figure 6, any $\epsilon \in (0, 1/3)$, and any an adp-IUF adversary $\mathcal{A}$ making at most Q_s queries to oSIGN_1 and at most Q_h random-oracle queries, given $(1+\varepsilon)^{N-1} < 2$, $L_1 \geq 2\sqrt{md} \cdot \rho_0(\varepsilon)$, $\sigma_r \geq \sqrt{Q_s}(\sqrt{md} \cdot \rho_0(\varepsilon) + \rho_1)$, and $\beta_{\mathsf{sis}} \geq 2L_1$, there exists an HNF-MSIS′ adversary $\mathcal{B}$ running in time that of $\mathcal{A}$ plus polynomial overhead such that*

$$\mathsf{Adv}^{\mathsf{adp\text{-}IUF}}(\mathcal{A}, \kappa) \leq (Q_H + 1)\left(e^{\pi}\sqrt{2\delta \mathsf{Adv}^{\mathsf{hnf\text{-}msis'}}_{q,d,n,k,\ell,\beta_{\mathsf{sis}},\beta_{\mathsf{lwe}}}(\mathcal{B}) + 2\delta\tau} + \Delta\right),$$

where $Q_H = Q_h + 2Q_s + 1$, $\delta = (1+\bar{\varepsilon})^{3Q_s}\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{2Q_s}$, $\bar{\varepsilon} = \max\left\{\frac{2\varepsilon}{1-\varepsilon}, \frac{2\left((1+\varepsilon)^{N-1}-1\right)}{2-(1+\varepsilon)^{N-1}}\right\}$, $\tau = \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-md}$, and $\Delta = Q_H(\tau + 2\varepsilon + Q_H \cdot 2^{-2\kappa})$.

In particular, if N, Q_s, and Q_h are polynomial in κ, ϵ and 2^{-md} are negligible in κ, then $\delta = 1 + \mathsf{negl}(\kappa)$ and $\Delta = \mathsf{negl}(\kappa)$.

Proof. We first normalize the random oracle queries.

- The adversary accesses the random oracle through $\text{RO}(i, x)$, while H_i denotes the corresponding truth table.
- Before every valid query $(\mu, \text{SS}, \mathcal{M})$ to oSIGN_2, we make on behalf of $\mathcal{A}$ any missing queries $\text{RO}(2, \mu, \text{SS}, \mathcal{M})$, and $\text{RO}(1, \mu, \mathsf{salt})$.
- Before checking the final output, we likewise make the query $\text{RO}(1, \mu^*, \mathsf{salt}^*)$ if it has not already been made.
- Repeated queries are answered from the tables.

Throughout the proof, a valid oSIGN_2 query means one that passes the validity checks of the original adp-IUF game, before any proof-specific abort. An abort terminates the entire experiment with output 0, whereas returning $\perp$ on an invalid oSIGN_2 query only rejects that query. Every valid oSIGN_2 query contains at least one honest signer, and consumes at least one previously unused honest oSIGN_1 output. Since at most Q_s such outputs are generated, there are at most Q_s valid oSIGN_2 queries. Hence, after normalization, there are at most $Q_H = Q_h + 2Q_s + 1$ fresh random oracle queries in total.

Also, there are two ways for $\mathcal{A}$ to win the adp-IUF game: (1) $\mathcal{A}$ outputs two distinct valid signatures associated with the same signing query $(\mu, \text{SS}, \mathcal{M}) \in \mathsf{Q}$; (2) the first output signature is valid and is not associated with any query in Q. Let WinDiff and WinFresh denote the events corresponding to the two cases, respectively. These two events are disjoint and together constitute the winning event of the original adp-IUF game.

We now define a sequence of games and let Adv_j denote the probability that $\mathcal{A}$ wins in G_j.

G_0: This is the original adp-IUF game defined in Figure 7. In particular, $\mathsf{Adv}_0 = \mathsf{Adv}^{\mathsf{adp\text{-}IUF}}(\mathcal{A}, \kappa)$.

G_1: Same as G_0 except that we additionally sample $j_1 \leftarrow \{0, 1, \ldots, Q_H\}$ at the beginning of the game and the winning condition is strengthened to: (1) WinDiff occurs and $j_1 = 0$, or (2) WinFresh occurs and (μ^*, salt^*) output by the adversary is the j_1-th query to $\text{RO}(1, \cdot)$.[8] In addition, G_1 aborts if the adversary makes a query $(\mu, \text{SS}, \mathcal{M})$ to oSIGN_2 such that $(\mu, H_2(\mu, \text{SS}, \mathcal{M}))$ is the j_1-th query to $\text{RO}(1, \cdot)$. The differences are also shown in Figure 8.

It is clear that G_1 is identical to G_0 until G_1 aborts. To analyze the probability that $\mathcal{A}$ wins G_1, first let us ignore this additional abort event. There are two cases. (1) If WinDiff occurs in G_0, since j_1 is uniformly sampled from $\{0, 1, \ldots, Q_H\}$, the probability $\mathcal{A}$ wins G_1 is exactly $1/(Q_H + 1)$. (2) Similarly, if WinFresh occurs in G_0, the probability $\mathcal{A}$ wins G_1 is also $1/(Q_H + 1)$.

The additional abort event removes none of these winning executions. For the first case, the abort event never occurs when $j_1 = 0$. For the second case, since WinFresh occurs, the adversary never made a query $(\mu, \text{SS}, \mathcal{M})$ such that $(\mu, H_2(\mu, \text{SS}, \mathcal{M})) = (\mu^*, \mathsf{salt}^*)$, which is the j_1-th query to $\text{RO}(1, \cdot)$, and thus the abort event did not occur. Therefore, $\mathsf{Adv}_1 = \mathsf{Adv}_0/(Q_H + 1)$.

[8] In particular, $\mathcal{A}$ does not win G_1 if WinFresh occurs and $j_1 = 0$.

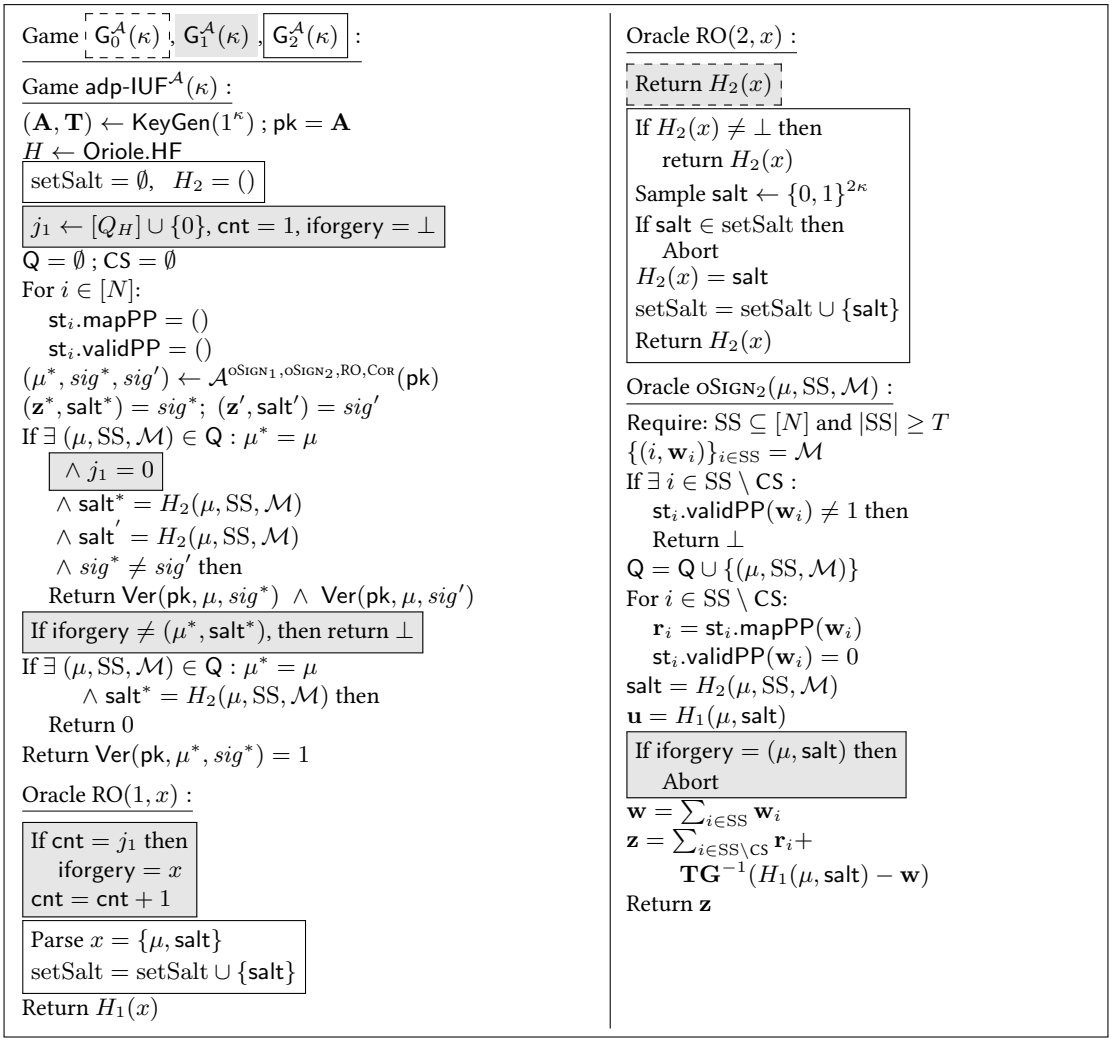

Fig. 8. The main procedure and the oracles RO$(1, \cdot)$, RO$(2, \cdot)$, oSign$_2$ of the games G_0, G_1, and G_2, where G_0 only contains the dashed box, G_1 only contains the highlighted boxes, and G_2 only contains the solid boxes. The rest of each game is the same as Fig. 7

G_2**:** Same as G_1 except that the generation of RO$(2, \cdot)$ queries is changed as follows. We initialize H_2 as an empty table and maintain a set $\mathrm{setSalt}$ containing all outputs returned by RO$(2, \cdot)$ and every salt appearing in a random oracle query of the form RO$(1, (\mu, \mathsf{salt}))$. On a random oracle query RO$(2, x)$, we sample $\mathsf{salt} \leftarrow \{0,1\}^{2\kappa}$. If $\mathsf{salt} \in \mathrm{setSalt}$, the game aborts with the event CollSalt. Otherwise, we add salt to $\mathrm{setSalt}$ and return $H_2(x) := \mathsf{salt}$. The differences are also shown in Figure 8.

Since the two games are identical until CollSalt occurs and $\Pr[\mathsf{CollSalt}] \leq Q_H^2 \cdot 2^{-2\kappa}$, $\mathsf{Adv}_1 \leq \mathsf{Adv}_2 + Q_H^2 \cdot 2^{-2\kappa}$.

G_3**:** Same as G_2 except the generation of fresh RO$(1, \cdot)$ queries is changed as follows. We initialize H_1 as an empty table and maintain an auxiliary table Aux. Whenever a fresh query to RO$(2, \mu, \mathrm{SS}, \mathcal{M})$ is made, we sample $\tilde{\mathbf{y}} \leftarrow \mathcal{D}_{\mathcal{R}^m, \sigma_0}$, where $\sigma_0 = \rho_0(\varepsilon)$, compute $\mathbf{v} = \mathbf{A}\tilde{\mathbf{y}} + \sum_{i \in \mathrm{SS}} \mathbf{w}_i$, where $\{(j, \mathbf{w}_j)\}_{j \in \mathrm{SS}} = \mathcal{M}$, and store $\mathsf{Aux}(\mu, \mathsf{salt}) := (\tilde{\mathbf{y}}, \mathbf{v})$, where $\mathsf{salt} = H_2(\mu, \mathrm{SS}, \mathcal{M})$. On a fresh query to RO$(1, x)$ with $x = (\mu, \mathsf{salt})$, we proceed as follows:

- If this is the j_1-th query to RO$(1, \cdot)$, sample $\mathbf{z}_0 \leftarrow \mathcal{D}_{\mathcal{R}^m, \sigma_1}$, where $\sigma_1 = 2\sigma_0$, and return $H_1(x) := \mathbf{A}\mathbf{z}_0$.
- Otherwise, if $\mathsf{Aux}(x) \neq \perp$ is defined, let $(\tilde{\mathbf{y}}, \mathbf{v}) = \mathsf{Aux}(x)$ and return $H_1(x) := \mathbf{v}$.
- For every remaining fresh query, sample $\mathbf{u} \leftarrow \mathcal{R}_q^n$ uniformly and return $H_1(x) := \mathbf{u}$.

In particular, if $j_1 = 0$, no vector $\mathbf{z}_0$ is sampled and no H_1 entry is programmed as $\mathbf{A}\mathbf{z}_0$. The differences are also shown in Figure 9.

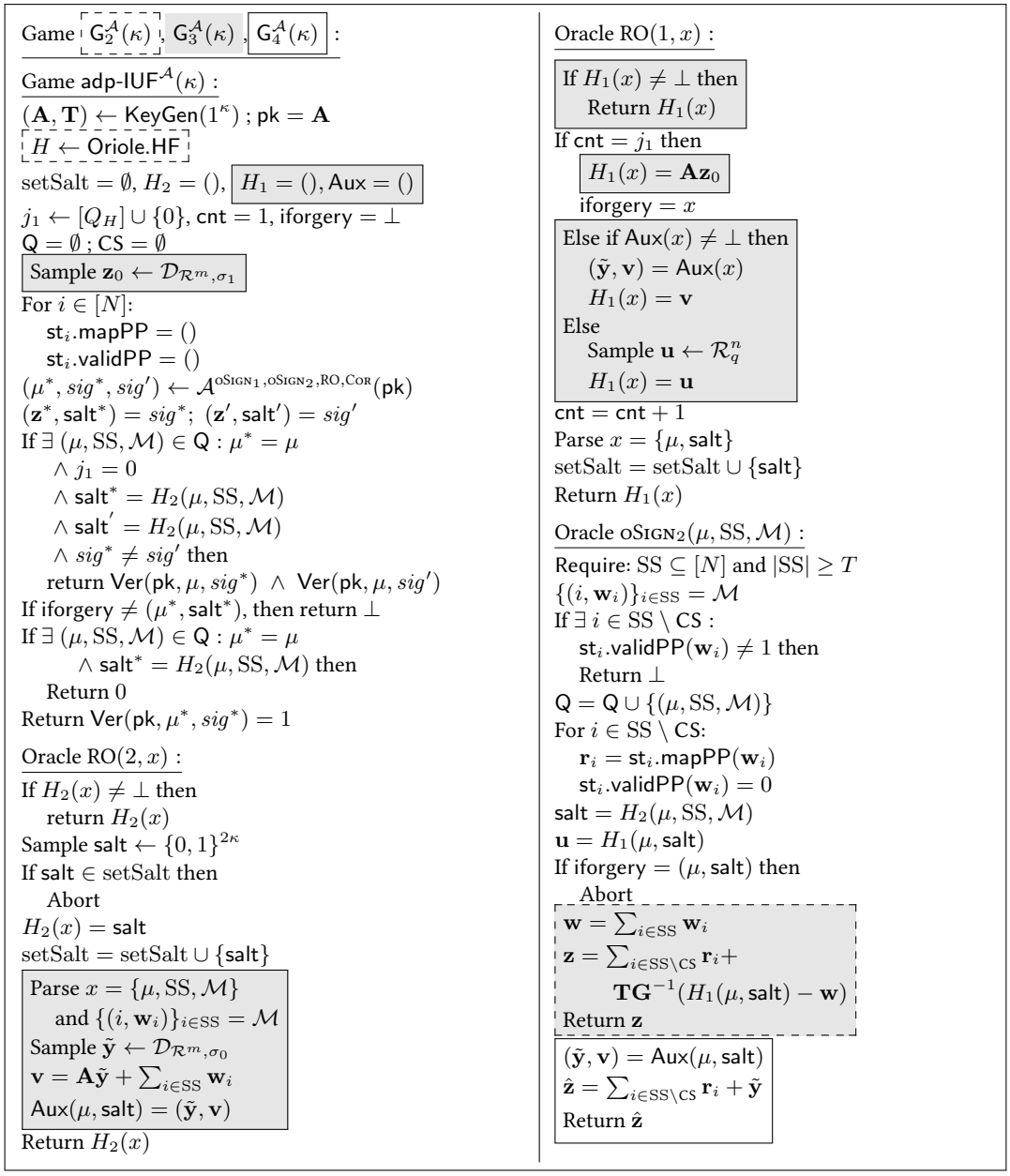

Fig. 9. Initialization, the oracles RO$(1, \cdot)$, RO$(2, \cdot)$ and oSIGN$_2$ of the games, G_2, G_3, G_4, where G_2 only contains the dashed box, G_3 only contains the highlighted boxes, and G_4 only contains the solid boxes. The rest of each game is the same as G_2.

By Lemma 6, both $\mathbf{A}\tilde{\mathbf{y}} + \mathbf{w}$ and $\mathbf{A}\mathbf{z}_0$ are within statistical distance at most 2ε of uniform over $\mathcal{R}_q^n$. Since at most Q_H fresh H_1 entries are returned, a standard hybrid argument gives $|\mathsf{Adv}_2 - \mathsf{Adv}_3| \leq 2Q_H \cdot \varepsilon$.

G_4**:** Same as G_3 except that when answering a valid query $(\mu, \text{SS}, \mathcal{M})$ to oSIGN$_2$, instead of returning $\mathbf{z} = \sum_{i \in \text{SS} \setminus \mathsf{CS}} \mathbf{r}_i + \mathbf{T}\mathbf{G}^{-1}\big(H_1(\mu, \mathsf{salt}) - \mathbf{w}\big)$, it returns $\hat{\mathbf{z}} = \sum_{i \in \text{SS} \setminus \mathsf{CS}} \mathbf{r}_i + \tilde{\mathbf{y}}$, where $\mathsf{Aux}(\mu, \mathsf{salt}) = (\tilde{\mathbf{y}}, \mathbf{v})$, $\mathsf{salt} = H_2(\mu, \text{SS}, \mathcal{M})$, $\mathcal{M} = \{(i, \mathbf{w}_i)\}_{i \in \text{SS}}$, and $\mathbf{w} = \sum_{i \in \text{SS}} \mathbf{w}_i$. The differences are also shown in Figure 9. The transition between these two games is analyzed in Lemma 15, which gives $\mathsf{Adv}_3 \leq e^\pi \sqrt{\delta \mathsf{Adv}_4} + Q_H \cdot \tau$.

Construction of $\mathcal{B}$. Let $\mathcal{B}$ be a HNF-MSIS′ adversary that, on input a challenge matrix $\mathbf{A}$, runs $\mathcal{A}$ by simulating G_4 faithfully except that $\mathcal{B}$ does not run KeyGen to generate $(\mathbf{A}, \mathbf{T})$. Then, $\mathcal{B}$ extracts a solution as follows. There are two cases.

Case $j_1 = 0$. If $\mathcal{A}$ wins G_4 simulated by $\mathcal{B}$, then WinDiff occurs. The two output signatures have the same salt salt and satisfy $\mathbf{A}\mathbf{z}^* = H_1(\mu^*, \mathsf{salt}) = \mathbf{A}\mathbf{z}'$ and $\mathbf{z}^* \neq \mathbf{z}'$. Thus, $\mathcal{B}$ outputs $\mathbf{z}^* - \mathbf{z}'$.

Case $j_1 > 0$. If $\mathcal{A}$ wins G_4 simulated by $\mathcal{B}$, then WinFresh occurs and $\mathsf{iforgery} = (\mu^*, \mathsf{salt}^*)$. Hence $\mathbf{A}\mathbf{z}^* = H_1(\mu^*, \mathsf{salt}^*) = \mathbf{A}\mathbf{z}_0$, and $\mathcal{B}$ outputs $\mathbf{z}_0 - \mathbf{z}^*$. Otherwise, $\mathcal{B}$ aborts.

Analysis of $\mathcal{B}$. It is clear that $\mathcal{B}$ simulates G_4 perfectly. It is left to bound the winning probability of $\mathcal{B}$. Denote $\mathsf{Adv}_{4,\mathsf{WinDiff}} := \Pr[\mathsf{G}_4 \text{ returns } 1 \wedge j_1 = 0]$ and $\mathsf{Adv}_{4,\mathsf{WinFresh}} := \Pr[\mathsf{G}_4 \text{ returns } 1 \wedge j_1 > 0]$, we have $\mathsf{Adv}_4 = \mathsf{Adv}_{4,\mathsf{WinDiff}} + \mathsf{Adv}_{4,\mathsf{WinFresh}}$.

For the first case, if $\mathcal{B}$ returns, the validity of the signatures gives $\|\mathbf{z}^* - \mathbf{z}'\| \leq \|\mathbf{z}^*\| + \|\mathbf{z}'\| \leq 2L_1 = \beta_{\mathsf{sis}}$, and thus $\mathbf{z}^* - \mathbf{z}'$ is a valid solution. Therefore, the probability that $\mathcal{B}$ wins in this case is $\mathsf{Adv}_{4,\mathsf{WinDiff}}$.

For the second case, the adversary's view depends on $\mathbf{z}_0$ only through $\mathbf{A}\mathbf{z}_0$, so conditioned on this view, $\mathbf{z}_0$ follows the Gaussian distribution of width σ_1 over the corresponding coset. Since $\sigma_1 = 2\rho_0(\varepsilon)$, by Lemma 2, the min-entropy of $\mathbf{z}_0$ is at least $md - 1$. Thus, the extracted vector $\mathbf{z}_0 - \mathbf{z}^*$ is nonzero with probability at least $1/2$.

Also, by Lemma 1, the probability that $\|\mathbf{z}_0\| > \sigma_1\sqrt{md}$ is at most τ, which implies that $\|\mathbf{z}_0 - \mathbf{z}^*\| \leq 2\sqrt{md}\rho_0(\varepsilon) + L_1 \leq 2L_1 = \beta_{\mathsf{sis}}$ except with probability τ. Thus, the probability that $\mathcal{B}$ in this case wins is at least $\frac{1}{2}\mathsf{Adv}_{4,\mathsf{WinFresh}} - \tau$.

Combining the two cases yields $\mathsf{Adv}^{\mathsf{hnf\text{-}msis}'}_{q,d,n,k,\ell,\beta_{\mathsf{sis}},\beta_{\mathsf{lwe}}}(\mathcal{B}) \geq \frac{1}{2}\mathsf{Adv}_{4,\mathsf{WinFresh}} - \tau + \mathsf{Adv}_{4,\mathsf{WinDiff}} \geq \frac{1}{2}\mathsf{Adv}_4 - \tau$, which concludes the proof. □

The flooding transition. The next lemma is the technical core of the proof. It relies on the conditional distribution described in Lemma 8 to characterize the information revealed when the adversary corrupts a signer or issues a query to $\textsc{oSign}_2$. By combining the noise-flooding technique of Lemma 14 with the weak triangle inequality from Lemma 12, we show that the Rényi distance between the transcript distributions generated in the two games is small. It applies to events determined by the adversary's transcript and output, together with the sampled index j_1.

Lemma 15 (Transition from G_3 to G_4). *For any event $\mathcal{E}$ determined by the adversary's transcript and output, together with j_1, given $(1+\varepsilon)^{N-1} < 2$, $\sigma_0 := \rho_0(\varepsilon)$ and $\sigma_r \geq \sqrt{Q_s}(\sqrt{md}\rho_0(\varepsilon) + \rho_1)$, it holds that $\Pr_{\mathsf{G}_3}[\mathcal{E}] \leq e^{\pi}\sqrt{\delta \Pr_{\mathsf{G}_4}[\mathcal{E}] + Q_H\tau}$, where the probabilities are over all randomness of the games and the adversary.*

Proof. Recall that the two games are identical except that every answered query $(\mu, \mathrm{SS}, \mathcal{M})$ to $\textsc{oSign}_2$ returns $\mathbf{z} = \sum_{i\in\mathrm{SS}\setminus\mathsf{CS}} \mathbf{r}_i + \mathbf{T}\mathbf{G}^{-1}\big(H_1(\mu, \mathsf{salt}) - \mathbf{w}\big)$ in G_3, and $\hat{\mathbf{z}} = \sum_{i\in\mathrm{SS}\setminus\mathsf{CS}} \mathbf{r}_i + \tilde{\mathbf{y}}$ in G_4, where $\mathsf{Aux}(\mu, \mathsf{salt}) = (\tilde{\mathbf{y}}, \mathbf{v})$, $\mathsf{salt} = H_2(\mu, \mathrm{SS}, \mathcal{M})$, $\mathcal{M} = \{(i, \mathbf{w}_i)\}_{i\in\mathrm{SS}}$, and $\mathbf{w} = \sum_{i\in\mathrm{SS}} \mathbf{w}_i$.

We first show that $\mathbf{z}$ and $\hat{\mathbf{z}}$ lie in the same coset of $\Lambda_q^{\perp}(\mathbf{A})$, or equivalently, $\mathbf{A}\mathbf{z} = \mathbf{A}\hat{\mathbf{z}}$. In particular, we just need to show that $H_1(\mu, \mathsf{salt}) = \mathbf{A}\tilde{\mathbf{y}} + \mathbf{w}$, which implies $\mathbf{A}\hat{\mathbf{z}} = \sum_{i\in\mathrm{SS}\setminus\mathsf{CS}} \mathbf{w}_i + \mathbf{A}\tilde{\mathbf{y}} = \sum_{i\in\mathrm{SS}\setminus\mathsf{CS}} \mathbf{w}_i + H_1(\mu, \mathsf{salt}) - \mathbf{w} = \mathbf{A}\mathbf{z}$. When the corresponding $\mathrm{RO}(2, \cdot)$ query first returns salt, the game stores $\mathbf{v} = \mathbf{A}\tilde{\mathbf{y}} + \mathbf{w}$ in $\mathsf{Aux}(\mu, \mathsf{salt})$. The salt-collision abort ensures that this salt has not appeared in an earlier $\mathrm{RO}(1, \cdot)$ query or as another $\mathrm{RO}(2, \cdot)$ output. Thus $H_1(\mu, \mathsf{salt})$ is assigned $\mathbf{v}$, unless $(\mu, \mathsf{salt}) = \mathsf{iforgery}$, in which case the signing query aborts. Hence every answered signing query satisfies $H_1(\mu, \mathsf{salt}) = \mathbf{A}\tilde{\mathbf{y}} + \mathbf{w}$.

We now bound the Rényi divergence of the view of the adversary in G_3 from that in G_4. Denote

$$X_0 := \big(\mathbf{A}, \mathbf{T}, j_1, \mathbf{z}_0, (\mathbf{w}^{(j)})_{j\in[Q_s]}, (\tilde{\mathbf{y}}^{(j)})_{j\in[Q_H]}, \omega_{\mathcal{A}}, \omega_{\mathsf{RO}}\big) ,$$

where $\mathbf{w}^{(j)}$ is the j-th first-round output and $\tilde{\mathbf{y}}^{(j)}$ is the j-th sampled auxiliary vector, with distribution $\mathcal{D}_{\mathcal{R}^m,\sigma_0}$. The lists are indexed by sampling order and completed independently to the stated lengths when fewer samples are used. Here $\mathbf{z}_0$ is the Gaussian vector for the selected H_1 query, $\omega_{\mathcal{A}}$ contains the adversary's coins, and ω_{RO} contains the coins for the remaining uniform oracle samples.

Then, let us record, in chronological order, the signing responses $\mathbf{z}$ (resp. $\hat{\mathbf{z}}$) and the first-round nonce $\mathbf{r}_i$ revealed by $\textsc{Cor}$ in G_3 (resp. G_4), and denote X_j as the j-th recorded value, which is either $\mathbf{z}$ (resp. $\hat{\mathbf{z}}$) or $\mathbf{r}_i$. The total number of recorded values is bounded by Q_s. This is because different signing responses use disjoint samples, and each contains a sample of a signer who is never corrupted. Each signing response therefore corresponds to a distinct nonce sample that is never revealed. Since at most Q_s nonce samples are generated, the total number of signing responses and individually revealed samples is at most Q_s. We pad this sequence with $\perp$ to length Q_s. For $b \in \{0, 1\}$ and $j \in [0..Q_s]$, let $P_j^{(b)}$ denote the distribution of X_j in G_{3+b}, and let $P_{[0..j]}^{(b)}$ denote the joint distribution of $X_{[0..j]}$. Also, we use $P_{j|x_{[0..(j-1)]}}^{(b)}$ to denote the distribution of X_j conditioned on $X_{[0..(j-1)]} = x_{[0..(j-1)]}$.

Since the view of the adversary in G_3 (resp. G_4) is determined by $X_{[0..Q_s]}$, we now bound $R_\alpha\left(P^{(0)}_{[0..Q_s]}\middle\|P^{(1)}_{[0..Q_s]}\right)$, and, by Lemma 13, we just need to bound $R_\alpha\left(P^{(0)}_{j|x_{[0..(j-1)]}}\middle\|P^{(1)}_{j|x_{[0..(j-1)]}}\right)$ for any $j\in[0..Q_s]$ and $x_{[0..(j-1)]}$. It is clear that $P^{(0)}_0$ is the same as $P^{(1)}_0$. Denote $\mathsf{TailAux}$ as the event that $\max_{j\in[Q_H]}\|\tilde{\mathbf{y}}^{(j)}\|>\sqrt{md}\,\sigma_0$. For $j\geq 1$, considering all $x_{[0..(j-1)]}$ where $\mathsf{TailAux}$ does not occur, there are two cases.

Signing response $X_j=\mathbf{z}$. Let $t'=|\mathsf{SS}\setminus\mathsf{CS}|$. By the marginal statement of Lemma 8 with $k=1$ (the case $t'=1$ is exact),

$$\begin{aligned}P^{(0)}_{j|x_{[0..(j-1)]}}&\overset{\bar\varepsilon}{\approx}\mathcal{D}_{\Lambda_q^\perp(\mathbf{A})+\mathbf{a},\sqrt{t'}\sigma_r,\mathbf{T}\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})}\ ,\\ P^{(1)}_{j|x_{[0..(j-1)]}}&\overset{\bar\varepsilon}{\approx}\mathcal{D}_{\Lambda_q^\perp(\mathbf{A})+\mathbf{a},\sqrt{t'}\sigma_r,\tilde{\mathbf{y}}}\ ,\end{aligned}$$

where $\mathbf{A}\mathbf{a}=H_1(\mu,\mathsf{salt})-\sum_{i\in\mathsf{SS}\cap\mathsf{CS}}\mathbf{w}_i$. By Lemmas 12 and 14 with $\alpha=2$,

$$\begin{aligned}R_2\left(P^{(0)}_{j|x_{[0..(j-1)]}}\middle\|P^{(1)}_{j|x_{[0..(j-1)]}}\right)&\leq(1+\bar\varepsilon)^3\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\exp\left(\frac{2\pi\|\mathbf{T}\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})-\tilde{\mathbf{y}}\|^2}{t'\sigma_r^2}\right)\\&\leq(1+\bar\varepsilon)^3\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\exp\left(\frac{2\pi(\sqrt{md}\,\sigma_0+\rho_1)^2}{\sigma_r^2}\right).\end{aligned}\tag{4}$$

First-round nonce $X_j=\mathbf{r}_i$. If $\mathbf{r}_i$ has not been used in any signing session, then $P^{(0)}_{j|x_{[0..(j-1)]}}=P^{(1)}_{j|x_{[0..(j-1)]}}$. Otherwise, let $\mathbf{z}$ be the corresponding signing response among the preceding recorded values $x_{[j-1]}$, and let t' be its number of honest samples when it was returned. Let J index those samples already revealed, and set $k:=|J|+1\leq t'-1$, $s_k:=\sigma_r\sqrt{\frac{t'-k}{t'-k+1}}\geq\frac{\sigma_r}{\sqrt{2}}\geq\rho_0(\varepsilon)$. By Lemma 8,

$$\begin{aligned}P^{(0)}_{j|x_{[0..(j-1)]}}&\overset{\bar\varepsilon}{\approx}\mathcal{D}_{\Lambda_q^\perp(\mathbf{A})+\mathbf{s}_i,s_k,\frac{\mathbf{z}-\sum_{j\in J}\mathbf{r}_j-\mathbf{T}\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})}{t'-k+1}}\ ,\\ P^{(1)}_{j|x_{[0..(j-1)]}}&\overset{\bar\varepsilon}{\approx}\mathcal{D}_{\Lambda_q^\perp(\mathbf{A})+\mathbf{s}_i,s_k,\frac{\mathbf{z}-\sum_{j\in J}\mathbf{r}_j-\tilde{\mathbf{y}}}{t'-k+1}}\ ,\end{aligned}$$

where $\mathbf{A}\mathbf{s}_i=\mathbf{w}_i$. Applying Lemmas 12 and 14 with $\alpha=2$ again gives

$$\begin{aligned}R_2\left(P^{(0)}_{j|x_{[0..(j-1)]}}\middle\|P^{(1)}_{j|x_{[0..(j-1)]}}\right)&\leq(1+\bar\varepsilon)^3\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\exp\left(\frac{2\pi\|\mathbf{T}\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})-\tilde{\mathbf{y}}\|^2}{(t'-k)(t'-k+1)\sigma_r^2}\right)\\&\leq(1+\bar\varepsilon)^3\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\exp\left(\frac{\pi(\sqrt{md}\,\sigma_0+\rho_1)^2}{\sigma_r^2}\right),\end{aligned}\tag{5}$$

where $(t'-k)(t'-k+1)\geq 2$.

We denote $x_{[0..(j-1)]}\notin\mathsf{TailAux}$ if and only if $\mathsf{TailAux}$ does not occur on $x_{[0..(j-1)]}$ and denote $P^{(b)}_{[0..Q_s]|\neg\mathsf{TailAux}}$ as the distribution $P^{(b)}_{[0..Q_s]}$ conditioned on $\neg\mathsf{TailAux}$. Combining (4)–(5) with Lemma 13, we obtain

$$\begin{aligned}R_2\left(P^{(0)}_{[0..Q_s]|\neg\mathsf{TailAux}}\middle\|P^{(1)}_{[0..Q_s]|\neg\mathsf{TailAux}}\right)&\leq\prod_{j=0}^{Q_s}\sup_{x_{[j-1]}\notin\mathsf{TailAux}}R_2\left(P^{(0)}_{j|x_{[0..(j-1)]}}\middle\|P^{(1)}_{j|x_{[0..(j-1)]}}\right)\\&\leq(1+\bar\varepsilon)^{3Q_s}\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{2Q_s}\exp\left(\frac{2\pi Q_s(\sqrt{md}\,\sigma_0+\rho_1)^2}{\sigma_r^2}\right)\\&\leq\delta e^{2\pi}.\end{aligned}$$

By data processing and probability preservation in Lemma 12, $\Pr_{\mathsf{G}_3}[\mathcal{E}\wedge\neg\mathsf{TailAux}]\leq e^\pi\sqrt{\delta\Pr_{\mathsf{G}_4}[\mathcal{E}\wedge\neg\mathsf{TailAux}]}$. Therefore, we can conclude the lemma since $\Pr_{\mathsf{G}_4}[\mathcal{E}\wedge\neg\mathsf{TailAux}]\leq\Pr_{\mathsf{G}_4}[\mathcal{E}]$ and, by Lemma 1, $\Pr_{\mathsf{G}_3}[\mathsf{TailAux}]\leq Q_H\frac{1+\varepsilon}{1-\varepsilon}2^{-md}=Q_H\tau$. □

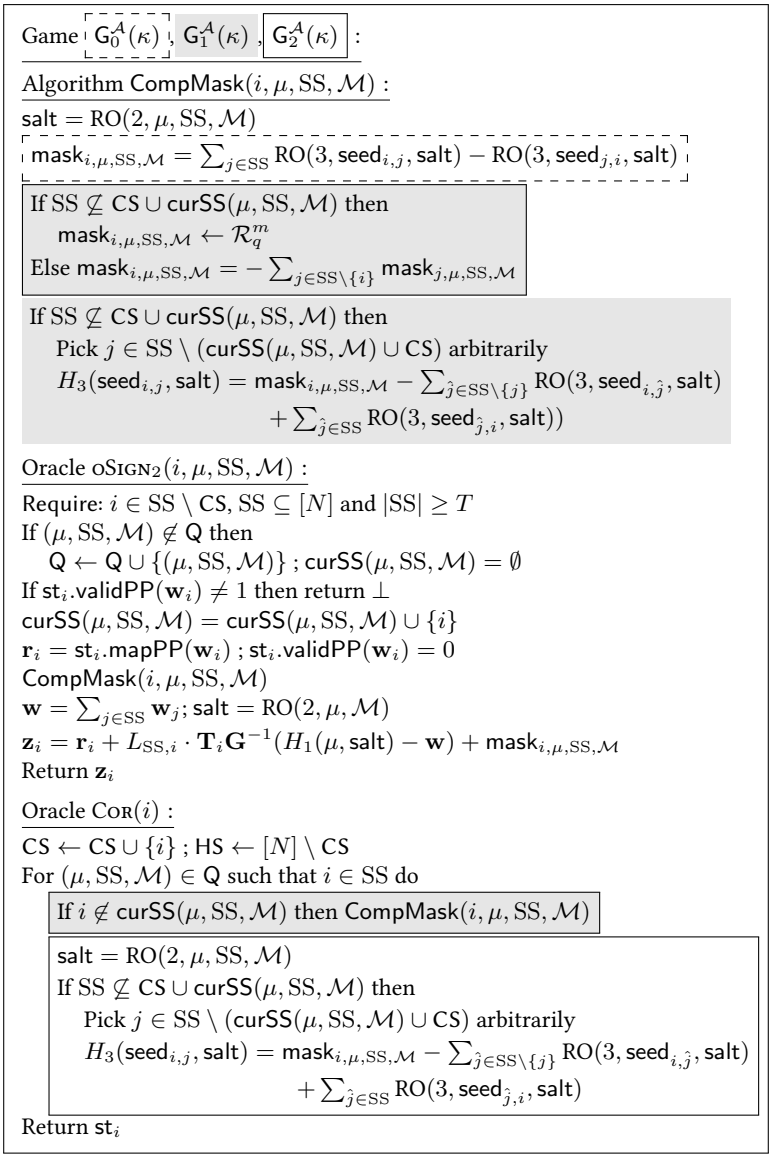

Fig. 10. The oracles $\textsc{oSign}_2$ and $\textsc{Cor}$ of the games G_0, G_1, and G_2, where G_0 only contains the dashed box, G_1 only contains the highlighted boxes, and G_2 only contains the solid boxes. The $\mathsf{CompMask}$ algorithm takes a second-round query $(\mu, \mathrm{SS}, \mathcal{M})$ and a signer index $i \in \mathrm{SS}$ as input and compute the mask $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$. Also, for a corrupted signer i, if $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$ has not been computed yet when referenced, it is computed as $\mathsf{mask}_{j,\mu,\mathrm{SS},\mathcal{M}} = \sum_{j\in\mathrm{SS}} \mathrm{RO}(3, \mathsf{seed}_{i,j}, \mathsf{salt}) - \mathrm{RO}(3, \mathsf{seed}_{j,i}, \mathsf{salt})$. The rest of each game is the same as $\mathsf{adp\text{-}TS\text{-}sUF\text{-}4}_{\mathsf{Oriole}}$.

4.4 Proof of Theorem 2

We show the theorem via the following series of games.

G_0: Same as the $\mathsf{adp\text{-}TS\text{-}sUF\text{-}4}_{\mathsf{Oriole}}$ game defined in Fig 4.

G_1: Same as G_0 except we change the way how the masks in the $\textsc{oSign}_2$ oracle are computed. We use $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$ to denote the mask computed by signer i when the adversary makes a second-round signing query $(\mu, \mathrm{SS}, \mathcal{M})$, and we use $\mathsf{CompMask}$ to denote the algorithm used to compute the mask. Also, we use a map $\mathsf{curSS}(\mu, \mathrm{SS}, \mathcal{M})$ to record the set of signers that already received the second-round query $(\mu, \mathrm{SS}, \mathcal{M})$. The changes from G_0 consist of the following two steps.

First, we additionally compute $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$ when signer i is corrupted even if signer i did not receive the query $(\mu, \mathrm{SS}, \mathcal{M})$ before. It is clear that this does not affect the view of $\mathcal{A}$ at all.

Second, we change how the masks are computed in CompMask. Concretely, in G_1, CompMask samples $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$ uniformly from $\mathcal{R}_q^m$, if $\mathrm{SS} \not\subseteq \mathsf{curSS}(\mu, \mathrm{SS}, \mathcal{M}) \cup \mathsf{CS}$, i.e., *not* all signers in SS is either corrupted or queried. Otherwise, since the sum of $\{\mathsf{mask}_{j,\mu,\mathrm{SS},\mathcal{M}}\}_{j\in\mathrm{SS}}$ is 0, CompMask computes $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}} \leftarrow -\sum_{j\in\mathrm{SS}\setminus\{i\}} \mathsf{mask}_{j,\mu,\mathrm{SS},\mathcal{M}}$. Then, CompMask programs the random oracle accordingly. The detailed differences are shown in Figure 10.

Denote BadSeed as the event that there is a RO query $H_3(\mathsf{seed}_{i,j}, x)$ for some $i, j \in [N]$ and x before signer i or j is corrupted, or there are two RO queries $H_2(x)$ and $H_2(x')$ such that $H_2(x) = H_2(x')$ and $x \neq x'$. In addition to the above changes, G_1 also aborts if BadSeed occurs.

We will show later that the number of random oracle queries is bounded by $Q = Q_h + Q_{s2}$. Since each $\mathsf{seed}_{i,j}$ is uniformly sampled from $\{0,1\}^{2\kappa}$, the probability that $\mathcal{A}$ guesses one of an honest seed is at most $N^2 \cdot 2^{-2\kappa}$ in a single RO query. Since the output of H_2 is 2κ bits, the probability that $\mathcal{A}$ finds a collision is at most $Q^2 \cdot 2^{-2\kappa}$. Therefore, the probability $\Pr[\mathsf{BadSeed}] \leq Q(Q+N^2)\cdot 2^{-2\kappa}$ in either G_0 or G_1. Since given BadSeed does not occur, G_1 provides the same view to $\mathcal{A}$ as G_0, $\mathsf{Adv}^{\mathsf{G}_1}(\mathcal{A},\kappa) \geq \mathsf{Adv}^{\mathsf{G}_0}(\mathcal{A},\kappa) - \Pr[\mathsf{BadSeed}] \geq \mathsf{Adv}^{\mathsf{G}_0}(\mathcal{A},\kappa) - Q(Q+N^2)\cdot 2^{-2\kappa}$.

G_2**:** Same as G_1 except that the programming of H_3 is delayed until the corresponding signer is corrupted, as shown in Figure 11. Similarly, given BadSeed does not occur, G_2 provides the same view to $\mathcal{A}$ as G_1. Therefore, $\mathsf{Adv}^{\mathsf{G}_2}(\mathcal{A},\kappa) = \mathsf{Adv}^{\mathsf{G}_1}(\mathcal{A},\kappa)$.

G_3**:** Same as G_2 except that, in $\mathsf{CompMask}(i,\mu,\mathrm{SS},\mathcal{M})$, we first compute $\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}}$ and then derive the corresponding mask $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$. Concretely, there are two cases: (1) CompMask is invoked by the $\textsc{oSign}_2$ oracle, in which case $i \notin \mathsf{CS}$; (2) CompMask is invoked by the $\textsc{Cor}$, in which case $i \in \mathsf{CS}$. To simplify notation, let us fix a second-round query $(\mu, \mathrm{SS}, \mathcal{M})$ for the following discussion and use mask_i, $\mathbf{z}_i$ and $\mathbf{r}_i$ to denote $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$, $\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}}$ and $\mathbf{r}_{i,\mu,\mathrm{SS},\mathcal{M}}$ respectively. For the first case, if i is not the last signer in SS such that $\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}}$ is not computed yet (equivalently, $\mathrm{SS} \not\subseteq \mathsf{CS} \cup \mathsf{curSS}(\mu,\mathrm{SS},\mathcal{M})$), we sample $\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}}$ uniformly from $\mathcal{R}_q^m$. Otherwise, denoting $\mathsf{hSet} = \mathrm{SS} \setminus \mathsf{CS}$ and $\mathsf{cSet} = \mathrm{SS} \cap \mathsf{CS}$, since in G_2,

$$\begin{aligned}\sum_{j\in\mathsf{hSet}} \mathbf{z}_j &= \sum_{j\in\mathsf{hSet}} \left(\mathbf{r}_j + L_{\mathrm{SS},j}\mathbf{T}_j\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w}) + \mathsf{mask}_j\right) \\ &= \sum_{j\in\mathsf{hSet}} \mathbf{r}_j + \left(\mathbf{T} - \sum_{j\in\mathsf{cSet}} L_{\mathrm{SS},j}\mathbf{T}_j\right)\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w}) \\ &\quad - \sum_{j\in\mathsf{cSet}} \mathsf{mask}_j\ ,\end{aligned} \tag{6}$$

we compute

$$\begin{aligned}\mathbf{z}_i = \sum_{j\in\mathsf{hSet}} \mathbf{r}_j + \left(\mathbf{T} - \sum_{j\in\mathsf{cSet}} L_{\mathrm{SS},j}\mathbf{T}_j\right)\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w}) \\ - \sum_{j\in\mathsf{cSet}} \mathsf{mask}_j - \sum_{j\in\mathsf{hSet}\setminus\{i\}} \mathbf{z}_j\ .\end{aligned} \tag{7}$$

Finally, we compute $\mathsf{mask}_i = \mathbf{z}_i - L_{\mathrm{SS},i}\mathbf{T}_i\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt}) - \mathbf{w}) - \mathbf{r}_i$.

For the second case, the only differences are that we set $\mathbf{r}_i = 0$ and in Equation (7), cSet is replaced by $\mathsf{cSet} \setminus \{i\}$. The changes are also shown in Figure 11.[9]

It is clear that the view of $\mathcal{A}$ remains the same in the first case. We now show that the view of $\mathcal{A}$ also remains the same in the second case. Note that in the second case, the value $\mathbf{z}_i$ is never used later in the game. Therefore, we just need to show that the distribution of mask_i remains the same. If $\mathrm{SS} \not\subseteq \mathsf{CS} \cup \mathsf{curSS}(\mu,\mathrm{SS},\mathcal{M})$, mask_i is uniformly distributed over $\mathcal{R}_q^m$, which is consistent with G_2. Otherwise, in

[9] Note in the first case, when computing $\mathbf{z}_i$ using Equation (7), we can also replace cSet with $\mathsf{cSet} \setminus \{i\}$ since $i \notin \mathsf{cSet}$, which makes the computation identical in both cases.

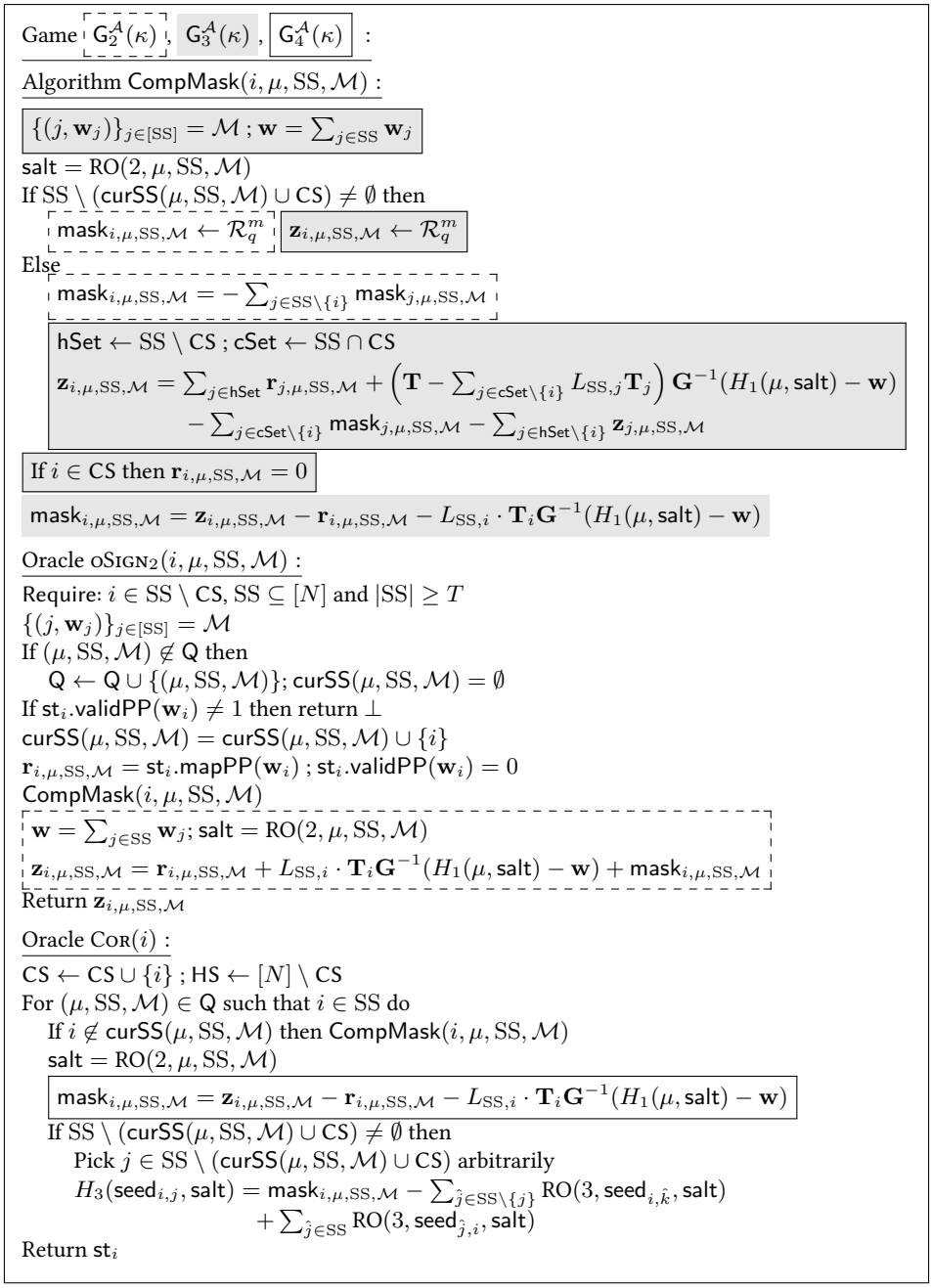

Fig. 11. The oracles oSign_2 and Cor of the games G_2, G_3, and G_4, where G_2 only contains the dashed box, G_3 only contains the highlighted boxes, and G_4 only contains the solid boxes. The rest of each game is the same as G_2.

G_3, by the above description, since $\mathbf{r}_i=0$,

$$\begin{aligned}\mathsf{mask}_i&=\mathbf{z}_i-\ \mathbf{r}_i-L_{\mathrm{SS},i}\mathbf{T}_i\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})\\&=\sum_{j\in\mathsf{hSet}}\mathbf{r}_j+\left(\mathbf{T}-\sum_{j\in\mathsf{cSet}\setminus\{i\}}L_{\mathrm{SS},j}\mathbf{T}_j\right)\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})\\&\quad-\sum_{j\in\mathsf{cSet}\setminus\{i\}}\mathsf{mask}_j-\sum_{j\in\mathsf{hSet}\setminus\{i\}}\mathbf{z}_j\\&\quad-L_{\mathrm{SS},i}\mathbf{T}_i\mathbf{G}^{-1}(H_1(\mu,\mathsf{salt})-\mathbf{w})\ ,\end{aligned}$$

which is consistent with Equation (6). (Note that in the above equation, $\mathsf{hSet} = \mathsf{hSet} \setminus \{i\}$ since $i \notin \mathsf{hSet}$.)

Therefore, the view of $\mathcal{A}$ remains the same after the changes, which means $\mathsf{Adv}^{\mathsf{G}_3}(\mathcal{A}, \kappa) = \mathsf{Adv}^{\mathsf{G}_2}(\mathcal{A}, \kappa)$.

G_4**:** Same as G_3 except that we further delay the computation of the masks in $\textsc{oSign}_2$ to the $\textsc{Cor}$ oracle. The differences are shown in Figure 11. We can delay the computation since $\mathsf{mask}_{i,\mu,\mathrm{SS},\mathcal{M}}$ is not used until signer i is corrupted. It is clear that the view of $\mathcal{A}$ remains the same. Therefore, $\mathsf{Adv}^{\mathsf{G}_4}(\mathcal{A}, \kappa) = \mathsf{Adv}^{\mathsf{G}_3}(\mathcal{A}, \kappa)$.

We now construct an adversary $\mathcal{B}$ playing the adp-IUF game that runs $\mathcal{A}$ by simulating the G_4 game.

Construction of $\mathcal{B}$. We denote CS, Q and st_i in G_4 as $\widetilde{\mathsf{CS}}$, $\widetilde{\mathsf{Q}}$ and $\widetilde{\mathsf{st}}_i$ in order to distinguish them from those defined in the adp-IUF game. After receiving $\mathbf{A}$ from the adp-IUF game, $\mathcal{B}$ initializes $\widetilde{\mathsf{CS}}$, $\widetilde{\mathsf{Q}}$, curSS and $\widetilde{\mathsf{st}}_i$ for each signer i following the game G_4 except that $\mathcal{B}$ samples $\mathbf{T}_i \leftarrow \mathcal{R}_q^{m\times(n+\ell)}$ when initializing $\widetilde{\mathsf{st}}_i.\mathsf{sk}$. Then, $\mathcal{B}$ runs $\mathcal{A}$ with input $\mathbf{A}$ and access to oracles $\widetilde{\textsc{oSign}_1}$, $\widetilde{\textsc{oSign}_2}$, $\widetilde{\mathrm{RO}}$, $\widetilde{\textsc{Cor}}$, which are simulated as follows.

$\widetilde{\mathbf{RO}}(i, x)$**:** For $i \in \{1, 2\}$, $\mathcal{B}$ directly forwards the query to its own oracle RO. For $i = 3$, if $H_3(x) = \perp$, $\mathcal{B}$ samples $H_3(x) \leftarrow \mathcal{R}_q^m$. Then, $\mathcal{B}$ returns $H_3(x)$.

$\widetilde{\textbf{oSign}}_1(i)$**:** Same as $\textsc{oSign}_1$ in G_4 except that $\mathcal{B}$ sets $\mathbf{w}_i \leftarrow \textsc{oSign}_1(i)$ and then sets $\widetilde{\mathsf{st}}_i.\mathsf{mapPP}(\mathbf{w}_i) = \perp$, and $\widetilde{\mathsf{st}}_i.\mathsf{validPP}(\mathbf{w}_i) = 1$, where $\textsc{oSign}_1(i)$ denotes a query to the $\textsc{oSign}_1$ oracle in the adp-IUF game.

$\widetilde{\textbf{oSign}}_2(i, \mu, \mathrm{SS}, \mathcal{M})$**:** Same as $\textsc{oSign}_2$ in G_4 except that we change the execution of CompMask as described below.

$\widetilde{\textbf{Cor}}(i)$**:** $\mathcal{B}$ obtains $\mathsf{st}_i \leftarrow \textsc{Cor}(i)$, where $\textsc{Cor}$ denotes the corruption oracle in the adp-IUF game, and sets $\widetilde{\mathsf{st}}_i.\mathsf{mapPP} = \mathsf{st}_i.\mathsf{mapPP}$. Then, for each $(\mu, \mathrm{SS}, \mathcal{M}) \in \widetilde{\mathsf{Q}}$ such that $i \in \mathsf{curSS}(\mu, \mathrm{SS}, \mathcal{M})$, $\mathcal{B}$ parse $\{(j, \mathbf{w}_j)\}_{j\in\mathrm{SS}}$ and sets $\mathbf{r}_{i,\mu,\mathrm{SS},\mathcal{M}} = \widetilde{\mathsf{st}}_i.\mathsf{mapPP}(\mathbf{w}_i)$. Finally, $\mathcal{B}$ follows $\textsc{Cor}(i)$ in G_4 except that $\mathcal{B}$ queries $\widetilde{\mathrm{RO}}(x)$ instead of $\mathrm{RO}(x)$, and the execution of the algorithms CompMask is changed as described below.

$\mathsf{CompMask}(i, \mu, \mathrm{SS}, \mathcal{M})$**:** Same as G_4 except when $\mathrm{SS} \subseteq \widetilde{\mathsf{CS}} \cup \mathsf{curSS}(\mu, \mathrm{SS}, \mathcal{M})$,[10] $\mathcal{B}$ queries $\bar{\mathbf{z}} \leftarrow \textsc{oSign}_2(\mu, \mathrm{SS}, \mathcal{M})$, where $\textsc{oSign}_2$ denotes the second-round signing oracle in the adp-IUF game, and computes

$$\begin{aligned}\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}} =& \bar{\mathbf{z}} - \sum_{j\in\mathsf{cSet}\setminus\{i\}} L_{\mathrm{SS},j} \cdot \mathbf{T}_j\mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w}) \\ & - \sum_{j\in\mathsf{cSet}\setminus\{i\}} \mathsf{mask}_{j,\mu,\mathrm{SS},\mathcal{M}} - \sum_{j\in\mathsf{hSet}\setminus\{i\}} \mathbf{z}_{j,\mu,\mathrm{SS},\mathcal{M}} \ ,\end{aligned}$$

where $\mathsf{cSet} = \mathrm{SS} \cap \widetilde{\mathsf{CS}}$ and $\mathsf{hSet} = \mathrm{SS} \setminus \widetilde{\mathsf{CS}}$.

After $\mathcal{A}$ returns (μ^*, sig^*, sig'), $\mathcal{B}$ outputs (μ^*, sig^*, sig').

Analysis of $\mathcal{B}$. We first show $\mathcal{B}$ simulates the oracles perfectly. It is clear that curSS, $\widetilde{\mathsf{st}}_i.\mathsf{validPP}$, the oracles $\widetilde{\mathrm{RO}}$ and $\widetilde{\textsc{oSign}_1}$ are simulated perfectly. For the oracle $\widetilde{\textsc{oSign}_2}$, we just need to show that $\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}}$ is computed consistent with the game G_4 when $\mathrm{SS} \subseteq \widetilde{\mathsf{CS}} \cup \mathsf{curSS}(\mu, \mathrm{SS}, \mathcal{M})$. By description of the $\textsc{oSign}_2$ oracle in the adp-IUF game,

$$\bar{\mathbf{z}} = \sum_{j\in\mathsf{hSet}} \mathbf{r}_{j,\mu,\mathrm{SS},\mathcal{M}} + \mathbf{T}\mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w}) \ ,$$

where $\{(j, \mathbf{w}_j)\}_{j\in\mathrm{SS}} = \mathcal{M}$, $\mathbf{w} = \sum_{j\in\mathrm{SS}} \mathbf{w}_j$ and $\mathbf{r}_{j,\mu,\mathrm{SS},\mathcal{M}} = \mathsf{st}_j.\mathsf{mapPP}(\mathbf{w}_j)$. Therefore,

$$\begin{aligned}\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}} =& \sum_{j\in\mathsf{hSet}} \mathbf{r}_{j,\mu,\mathrm{SS},\mathcal{M}} + \mathbf{T}\mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w}) \\ & - \sum_{j\in\mathsf{cSet}\setminus\{i\}} L_{\mathrm{SS},j} \cdot \mathbf{T}_j\mathbf{G}^{-1}(H_1(\mu, \mathsf{salt}) - \mathbf{w}) \\ & - \sum_{j\in\mathsf{cSet}\setminus\{i\}} \mathsf{mask}_{j,\mu,\mathrm{SS},\mathcal{M}} - \sum_{j\in\mathsf{hSet}\setminus\{i\}} \mathbf{z}_{j,\mu,\mathrm{SS},\mathcal{M}}\end{aligned}$$

which is consistent with G_4.

[10] Note that when $\mathrm{SS} \not\subseteq \widetilde{\mathsf{CS}} \cup \mathsf{curSS}(\mu, \mathrm{SS}, \mathcal{M})$, $\mathcal{B}$ can follow G_4 since $\mathbf{z}_{i,\mu,\mathrm{SS},\mathcal{M}}$ is just sampled uniformly from $\mathcal{R}_q^m$.

For the oracle $\widetilde{\text{COR}}$, we first show that $\widetilde{\mathsf{st}}_i.\mathsf{sk}$ is distributed identically to G_4. It is clear that $\mathsf{seed}_{i,j}$ is sampled following G_4. For $\mathbf{T}_i$, since throughout the simulation, only $\{\mathbf{T}_j\}_{j\in\widetilde{\mathsf{CS}}}$ are used and $|\widetilde{\mathsf{CS}}| < t$, the distribution of $\{\mathbf{T}_j\}_{j\in\widetilde{\mathsf{CS}}}$ is uniformly random in G_4, which is consistent with the simulation.

Also, by the simulation, $\mathsf{st}_i.\mathsf{mapPP}$ maintained in the adp-IUF game is identical to that in G_4. Therefore, after $\widetilde{\text{COR}}(i)$, $\widetilde{\mathsf{st}}_i.\mathsf{mapPP}$ is also identical to that in G_4. We then show that the computation of $\mathbf{r}_{i,\mu,\text{SS},\mathcal{M}}$ is identical to G_4 for each $(\mu,\text{SS},\mathcal{M}) \in \widetilde{\mathsf{Q}}$ such that $i \in \text{SS}$. If $i \notin \mathsf{curSS}(\mu,\text{SS},\mathcal{M})$, $\mathbf{r}_{i,\mu,\text{SS},\mathcal{M}}$ is set to 0 in both G_4 and the simulation. Otherwise, in G_4, $\mathbf{r}_{i,\mu,\text{SS},\mathcal{M}}$ is set to $\widetilde{\mathsf{st}}_i.\mathsf{mapPP}(\mathbf{w}_i)$ with $\{(j,\mathbf{w}_j)\}_{j\in\text{SS}} = \mathcal{M}$ in the oracle oSIGN_2, which is identical to the simulation.

Finally, by the same argument as the simulation of oSIGN_2, the simulation of CompMask and $\mathsf{mask}_{i,\mu,\text{SS},\mathcal{M}}$ is also perfect. Therefore, $\mathcal{B}$ simulates the G_4 game perfectly. The number of oSIGN_1 queries made by $\mathcal{B}$ is at most Q_{s1}. Also, since for each $(\mu,\text{SS},\mathcal{M}) \in \widetilde{\mathsf{Q}}$, there is at most one additional random oracle query to $\widetilde{\text{RO}}(2,\mu,\text{SS},\mathcal{M})$, the number of random oracle queries made by $\mathcal{B}$ is at most $Q = Q_h + Q_{s2}$.

It is left to show that if $\mathcal{A}$ wins G_4 simulated by $\mathcal{B}$, $\mathcal{B}$ wins the adp-IUF game. There are two ways $\mathcal{A}$ wins G_4. First, $\mathcal{A}$ wins if $sig^* = (\mathbf{z}^*,\mathsf{salt}^*)$ and $sig' = (\mathbf{z}',\mathsf{salt}')$ are two distinct valid signatures for μ^* and there exists $(\mu^*,\text{SS},\mathcal{M}) \in \widetilde{\mathsf{Q}}$ such that $\mathsf{salt}^* = \mathsf{salt}' = H_2(\mu^*,\text{SS},\mathcal{M})$. Since $\mathcal{B}$ forwards the random oracle queries $\widetilde{\text{RO}}(1,x)$ and $\widetilde{\text{RO}}(2,x)$ directly to adp-IUF and uses the public key $\mathbf{A}$ from adp-IUF, the signatures are also valid and associated with the query in the adp-IUF game. It is clear that if $(\mu^*,\text{SS},\mathcal{M}) \in \mathsf{Q}$, $\mathcal{B}$ wins. Otherwise, since BadSeed does not occur, we know that for all $(\mu^*,\text{SS}',\mathcal{M}') \in \mathsf{Q}$, $\mathsf{salt}^* \neq H_2(\mu',\text{SS}',\mathcal{M}')$. Therefore, sig^* is not a trivial forgery, which means that $\mathcal{B}$ wins adp-IUF.

Second, $\mathcal{A}$ wins if sig^* is a valid signature for μ^* and for all $(\mu^*,\text{SS},\mathcal{M}) \in \widetilde{\mathsf{Q}}$, either $\text{SS} \not\subseteq \mathsf{CS}\cup\mathsf{curSS}(\mu,\text{SS},\mathcal{M})$ or $\mathsf{salt}^* \neq H_2(\mu,\text{SS},\mathcal{M})$. From the simulation of CompMask, a query $(\mu,\text{SS},\mathcal{M}) \in \widetilde{\mathsf{Q}}$ is added to Q in adp-IUF only if $\mathcal{B}$ queries $\text{oSIGN}_2(\mu,\text{SS},\mathcal{M})$, which only occurs if $\text{SS} \subseteq \mathsf{CS}\cup\mathsf{curSS}(\mu,\text{SS},\mathcal{M})$. Therefore, for all $(\mu^*,\text{SS},\mathcal{M}) \in \mathsf{Q}$, we know $\mathsf{salt}^* \neq H_2(\mu,\text{SS},\mathcal{M})$. Therefore, sig^* is not a trivial forgery, which means that $\mathcal{B}$ wins adp-IUF. Therefore, $\mathsf{Adv}^{\mathsf{adp\text{-}IUF}}(\mathcal{B},\kappa) \geq \mathsf{Adv}^{\mathsf{G}_4}(\mathcal{A},\kappa)$, which concludes the theorem. □

4.5 Relevant bounds in key generation

Bounding $\rho_1 = \max_{\mathbf{u}\in\mathcal{R}_q^n} \|\mathbf{T}\mathbf{G}^{-1}(\mathbf{u})\|$. Recall that $\mathbf{T} = \begin{bmatrix}\mathbf{I}_n & -\mathbf{E}\\ \mathbf{0} & -\mathbf{R}\\ \mathbf{0} & \mathbf{I}_\ell\end{bmatrix}$, where $\mathbf{R} \in \mathscr{B}_{\beta_{\mathsf{lwe}}}^{k\times(nt)}, \mathbf{E} \in \mathscr{B}_{\beta_{\mathsf{lwe}}}^{n\times(nt)}$, and $\mathbf{G}^{-1}(\mathbf{u}) := \begin{bmatrix}\mathbf{u}_0\\ \mathbf{u}_1\end{bmatrix}$, where $\mathbf{u}_0 \in \mathscr{B}_{\lceil q/p^t\rceil}^n$ and $\mathbf{u}_1 \in \mathscr{B}_p^{nt}$. Therefore, $\|\mathbf{u}_0\| \leq \sqrt{nd}\cdot\lceil q/p^t\rceil$ and $\|\mathbf{u}_1\| \leq \sqrt{tnd}\cdot p$. Since $\mathbf{R}\mathbf{u}_1 \in \mathscr{B}_{ndtp\beta_{\mathsf{lwe}}}^k$ and $\mathbf{E}\mathbf{u}_1 \in \mathscr{B}_{ndtp\beta_{\mathsf{lwe}}}^n$, we obtain $\|(\mathbf{E}\mathbf{u}_1,\mathbf{R}\mathbf{u}_1)\| \leq tndp\beta_{\mathsf{lwe}}\sqrt{(n+k)d}$. Since $\mathbf{T}\mathbf{G}^{-1}(\mathbf{u}) = \begin{bmatrix}\mathbf{u}_0 - \mathbf{E}\mathbf{u}_1\\ -\mathbf{R}\mathbf{u}_1\\ \mathbf{u}_1\end{bmatrix}$, we have $\|\mathbf{T}\mathbf{G}^{-1}(\mathbf{u})\| \leq \|\mathbf{u}_0\| + \|(\mathbf{E}\mathbf{u}_1,\mathbf{R}\mathbf{u}_1)\| + \|\mathbf{u}_1\|$, which implies

$$\rho_1 = \max_{\mathbf{u}\in\mathcal{R}_q^n} \|\mathbf{T}\mathbf{G}^{-1}(\mathbf{u})\| \leq \sqrt{nd}\left(\lceil q/p^t\rceil + tdp\beta_{\mathsf{lwe}}\sqrt{n(n+k)} + p\sqrt{t}\right) . \tag{8}$$

Bounding $\rho_0(\varepsilon) = \eta_\varepsilon(\Lambda_q^\perp(\mathbf{A}))$. By Lemma 5, it suffices to construct a basis of $\Lambda_q^\perp(\mathbf{A})$ with bounded Gram–Schmidt lengths. Define $\mathbf{g} := (1, \lceil q/p^t\rceil, \lceil q/p^t\rceil\cdot p, \ldots, \lceil q/p^t\rceil\cdot p^{t-1})$, so that $\mathbf{G} = \mathbf{g}\otimes\mathbf{I}_n$. Adapting the construction in [MP12, Section 4.2], let $\mathbf{S}_\mathbf{g} = (\lceil q/p^t\rceil\cdot\mathbf{e}_1 - \mathbf{e}_2, p\mathbf{e}_2 - \mathbf{e}_3, \ldots, p\mathbf{e}_t - \mathbf{e}_{t+1}, q\mathbf{e}_1)$, where $\mathbf{e}_i, i \in [t+1]$ are the unit vectors of $\mathbb{Z}^{t+1}$. Since $\mathbf{g}\mathbf{S}_\mathbf{g} = (0,\ldots,0,q)$ and $\det(\mathbf{S}_\mathbf{g}) = q$, these columns form a basis of $\Lambda_q^\perp(\mathbf{g})$. The first t columns span $\ker(\mathbf{g})$ and have norms at most $\max\{\sqrt{\lceil q/p^t\rceil^2+1}, \sqrt{p^2+1}\}$. The last Gram–Schmidt vector has norm $q/\|\mathbf{g}\| \leq p$, since $q \leq \lceil q/p^t\rceil\cdot p^t$. Consequently, setting $\mathbf{S}_\mathbf{G} := \mathbf{S}_\mathbf{g}\otimes\mathbf{I}_n$ gives $\phi_M(\mathbf{S}_\mathbf{G}) = \mathbf{S}_\mathbf{g}\otimes\mathbf{I}_{nd}$, $\|\widetilde{\mathbf{S}}_\mathbf{G}\| = \|\widetilde{\mathbf{S}}_\mathbf{g}\| \leq \max\{2p, \lceil q/p^t\rceil + 1\}$. (Recall from Sections 3.1 and 3.2, for a basis $\mathbf{B}$, $\widetilde{\mathbf{B}}$ denotes its Gram–Schmidt orthogonalization, $\|\widetilde{\mathbf{B}}\|$ denotes the largest Euclidean norm of its columns, and ϕ_M denotes the coefficient embedding for matrices.)

Choose $\mathbf{W} \in \mathcal{R}^{(n+\ell)\times k}$ satisfying $\mathbf{A}_0 + \mathbf{G}\mathbf{W} = \mathbf{0} \bmod q$, and define $\hat{\mathbf{T}} := \begin{bmatrix}\mathbf{0} & \mathbf{I}_n & -\mathbf{E}\\ \mathbf{I}_k & \mathbf{0} & -\mathbf{R}\\ \mathbf{0} & \mathbf{0} & \mathbf{I}_\ell\end{bmatrix}$, $\mathbf{B} := \begin{bmatrix}\mathbf{0} & \mathbf{I}_k\\ \mathbf{S}_\mathbf{G} & \mathbf{W}\end{bmatrix}$, and $\mathbf{S}_\mathbf{A} := \hat{\mathbf{T}}\mathbf{B}$. Then, the matrix $\phi_M(\hat{\mathbf{T}})$ is unimodular and $\mathbf{A}\hat{\mathbf{T}} = [\mathbf{A}_0 \mid \mathbf{G}] \bmod q$. Since $\mathbf{B}$ is a basis of $\Lambda_q^\perp([\mathbf{A}_0 \mid \mathbf{G}])$, $\mathbf{S}_\mathbf{A}$ is a basis of $\Lambda_q^\perp(\mathbf{A})$. In the displayed column order, the columns coming from $\mathbf{S}_\mathbf{G}$ are

κ	$\log_2(q)$	n	k	t	p	σ_r	L_1	$\lvert\mathsf{pk}\rvert$	$\lvert sig\rvert$	Comm.
128	93	9	9	2	2^{28}	2^{79}	2^{91}	1.92MB	160KB	214KB

Fig. 12. The concrete parameters and estimated efficiency of Oriole for $\kappa = 128$ and $N = 1024$. We set $(d, \beta_{\mathsf{lwe}}) = (512, 2)$. The last column denotes the communication complexity per signer.

orthogonalized first. The remaining columns then project to the identity block, giving $\|\widetilde{\mathbf{B}}\| = \max\{1, \|\widetilde{\mathbf{S}}_{\mathbf{G}}\|\}$. As in [MP12, Lemma 5.3], multiplication by $\hat{\mathbf{T}}$ increases each Gram–Schmidt length by at most $\sigma_{\max}(\hat{\mathbf{T}})$. Hence

$$\|\widetilde{\mathbf{S}}_{\mathbf{A}}\| \le \sigma_{\max}(\hat{\mathbf{T}})\|\widetilde{\mathbf{B}}\| \le \sigma_{\max}(\hat{\mathbf{T}}) \max\{2p, \lceil q/p^t \rceil + 1\}$$

For a sampled key, $\sigma_{\max}(\hat{\mathbf{T}})$ can be evaluated numerically. For concrete parameter estimates, we use the heuristic $\sigma_{\max}(\hat{\mathbf{T}}) \lesssim \beta_{\mathsf{lwe}}(\sqrt{k} + \sqrt{n} + 2\sqrt{\ell})\sqrt{d}$ following the random-matrix estimates in [MP12, Section 5.2]. Finally, by Lemma 5, we obtain the bound

$$\rho_0(\varepsilon) \lesssim \sqrt{\frac{\log(2md(1+1/\varepsilon))}{\pi}} \cdot \beta_{\mathsf{lwe}}(\sqrt{k} + \sqrt{n} + 2\sqrt{\ell})\sqrt{d} \cdot \max\{2p, \lceil q/p^t \rceil + 1\} \,. \tag{9}$$

4.6 Parameter selection

Given $(\kappa, d, q, n, k, t, m, \beta_{\mathsf{lwe}})$ as specified in Figure 6, the rest of the parameters are set as follows.

- By Theorem 3 with $\varepsilon = 2^{-2\kappa}$ and Equations (8) and (9), we should set $\sigma_r = \sqrt{Q_s}\,(a_1 \cdot \max\{2p, \lceil q/p^t \rceil + 1\} + a_2 \cdot \lceil q/p^t \rceil + a_3 \cdot p)$, where $a_1 = \sqrt{\frac{\log(2md(1+1/\varepsilon))}{\pi}}\beta_{\mathsf{lwe}}(\sqrt{k} + \sqrt{n} + 2\sqrt{\ell})\sqrt{md}$, $a_3 = ntd\beta_{\mathsf{lwe}}\sqrt{(n+k)d} + \sqrt{ntd}$, and $a_2 = \sqrt{nd}$ are values already determined. To minimize σ_r, we first observe that $2p$ should be smaller than $\lceil q/p^t \rceil + 1$, since otherwise we can always reduce σ_r by reducing p until $2p < \lceil q/p^t \rceil + 1$. This gives $\sigma_r = \sqrt{Q_s}((a_1 + a_2) \cdot \lceil q/p^t \rceil + a_3 \cdot p + a_1)$, which is optimized when $p \approx ((a_1 + a_2)tq/a_3)^{1/(t+1)}$. Therefore, we set $p = \lceil ((a_1 + a_2)tq/a_3)^{1/(t+1)} \rceil$ and set σ_r accordingly. Note here that since $a_3 \gg (a_1 + a_2)t$, we indeed have $2p < \lceil q/p^t \rceil + 1$.
- To guarantee correctness (Theorem 1), we set $L_1 = (\sqrt{mdN} + 1)\sigma_r$.
- By Theorem 3, we set $\beta_{\mathsf{sis}} = 2L_1$.

Finally, we optimize over $(d, q, n, k, t, m, \beta_{\mathsf{lwe}})$ such that for any HNF-MSIS adversary $\mathcal{A}$ and MLWE adversary $\mathcal{B}$, $\mathsf{Adv}^{\mathsf{hnf\text{-}msis}}_{q,d,n,m,\beta_{\mathsf{sis}}}(\mathcal{A}) \le 2^{-\kappa}$ and $\mathsf{Adv}^{\mathsf{mlwe}}_{q,d,n,n+k,\beta_{\mathsf{lwe}}}(\mathcal{B}) \le 2^{-\kappa}$. Note that here, following common practice, we do not choose parameters according to the concrete bounds from the reduction, but rather we view the loss as an artifact of the proof and choose parameters so that MSIS and MLWE gives κ bits of security.

Optimizations for $\mathbf{z}$. Let $\mathbf{A} = [\mathbf{I}_n \mid \mathbf{A}']$, $sig = (\mathsf{salt}, \mathbf{z})$ and denote $\mathbf{z} = \binom{\overline{\mathbf{z}}}{\underline{\mathbf{z}}}$, where $\overline{\mathbf{z}} \in \mathcal{R}_q^n$ and $\underline{\mathbf{z}} \in \mathcal{R}_q^{k+\ell}$. Then, the signature only needs to contain $\underline{\mathbf{z}}$ instead of all of $\mathbf{z}$. This is because $\mathbf{A}\mathbf{z} = \overline{\mathbf{z}} + \mathbf{A}'\underline{\mathbf{z}}$. So instead of checking $\mathbf{A}\mathbf{z} = H_1(\mu, \mathsf{salt})$, we simply compute $\overline{\mathbf{z}} := H_1(\mu, \mathsf{salt}) - \mathbf{A}'\underline{\mathbf{z}}$ and check $\sqrt{\|\overline{\mathbf{z}}\|^2 + \|\underline{\mathbf{z}}\|^2} \le L_1$. Given this optimization, the final signature size is $d(k+tn)\log_2(q)$ bits, and the communication complexity per signer is $d(n+k+tn)\log_2(q)$ bits.

Concrete parameters. We give a set of candidate concrete parameters in Figure 12 based on the parameter selection described. Here we consider $\kappa = 128$, $Q_s = 2^{60}$ and $N = 1024$. The hardness of MLWE and HNF-MSIS is estimated using the state-of-the-art tool, lattice estimator.[11]

Acknowledgments. KJ and HW thank Divesh Aggarwal for hosting their visits at NUS and CQT, where part of this work was done. KJ is supported by NRF investigatorship grant NRF-NRFI09-0005) and the Breakthrough Plan of the Ministry of Education, China (No. JYB2025XDXM114). AI tools were used solely for grammar correction and minor language editing. All technical content, results, and proofs are the authors' own.

[11] https://github.com/malb/lattice-estimator

References

ABHR25. Karen Azari, Cecilia Boschini, Kristina Hostáková, and Michael Reichle. Security amplification of threshold signatures in the standard model. In Benny Applebaum and Huijia (Rachel) Lin, editors, *TCC 2025, Part III*, volume 16270 of *LNCS*, pages 201–234. Springer, Cham, December 2025.

AGHS13. Shweta Agrawal, Craig Gentry, Shai Halevi, and Amit Sahai. Discrete Gaussian leftover hash lemma over infinite domains. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 97–116. Springer, Berlin, Heidelberg, December 2013.

ALLW25. Martin R. Albrecht, Russell W. F. Lai, Oleksandra Lapiha, and Ivy K. Y. Woo. Partial lattice trapdoors: How to split lattice trapdoors, literally. In Goichiro Hanaoka and Bo-Yin Yang, editors, *ASIACRYPT 2025, Part III*, volume 16247 of *LNCS*, pages 265–296. Springer, Singapore, December 2025.

ASY22. Shweta Agrawal, Damien Stehlé, and Anshu Yadav. Round-optimal lattice-based threshold signatures, revisited. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *ICALP 2022*, volume 229 of *LIPIcs*, pages 8:1–8:20. Schloss Dagstuhl, July 2022.

BCdP+25. Giacomo Borin, Sofía Celi, Rafael del Pino, Thomas Espitau, Guilhem Niot, and Thomas Prest. Threshold signatures reloaded: ML-DSA and enhanced raccoon with identifiable aborts. Cryptology ePrint Archive, Paper 2025/1166, 2025.

BCdP+26. Giacomo Borin, Sofía Celi, Rafael del Pino, Thomas Espitau, Shuichi Katsumata, Guilhem Niot, Thomas Prest, and Kaoru Takemure. Hermine: An efficient lattice-based FROST-like threshold signature. Cryptology ePrint Archive, Paper 2026/419, 2026.

BCK+22. Mihir Bellare, Elizabeth C. Crites, Chelsea Komlo, Mary Maller, Stefano Tessaro, and Chenzhi Zhu. Better than advertised security for non-interactive threshold signatures. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part IV*, volume 13510 of *LNCS*, pages 517–550. Springer, Cham, August 2022.

BCL+26. Renas Bacho, Yanbo Chen, Julian Loss, Stefano Tessaro, and Chenzhi Zhu. Adaptively secure partially non-interactive threshold schnorr signatures in the AGM. In Joan Daemen and Emmanuel Thomé, editors, *EUROCRYPT 2026, Part I*, volume 16541 of *LNCS*, pages 33–62. Springer, Cham, May 2026.

BD21. Mihir Bellare and Wei Dai. Chain reductions for multi-signatures and the HBMS scheme. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 650–678. Springer, Cham, December 2021.

BdCE+25. Alexander Bienstock, Leo de Castro, Daniel Escudero, Antigoni Polychroniadou, and Akira Takahashi. Efficient, scalable threshold ML-DSA signatures: An MPC approach. Cryptology ePrint Archive, Paper 2025/1163, 2025.

BDLR25a. Renas Bacho, Sourav Das, Julian Loss, and Ling Ren. Adaptively secure three-round threshold Schnorr signatures from DDH. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VI*, volume 16005 of *LNCS*, pages 390–422. Springer, Cham, August 2025.

BDLR25b. Renas Bacho, Sourav Das, Julian Loss, and Ling Ren. Glacius: Threshold schnorr signatures from DDH with full adaptive security. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part II*, volume 15602 of *LNCS*, pages 304–334. Springer, Cham, May 2025.

BGG+18. Dan Boneh, Rosario Gennaro, Steven Goldfeder, Aayush Jain, Sam Kim, Peter M. R. Rasmussen, and Amit Sahai. Threshold cryptosystems from threshold fully homomorphic encryption. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 565–596. Springer, Cham, August 2018.

BGGK17. Dan Boneh, Rosario Gennaro, Steven Goldfeder, and Sam Kim. A lattice-based universal thresholdizer for cryptographic systems. Cryptology ePrint Archive, Report 2017/251, 2017.

BKL+25. Cecilia Boschini, Darya Kaviani, Russell W. F. Lai, Giulio Malavolta, Akira Takahashi, and Mehdi Tibouchi. Ringtail: Practical two-round threshold signatures from learning with errors. In Marina Blanton, William Enck, and Cristina Nita-Rotaru, editors, *2025 IEEE Symposium on Security and Privacy*, pages 149–164. IEEE Computer Society Press, May 2025.

BL22. Renas Bacho and Julian Loss. On the adaptive security of the threshold BLS signature scheme. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 193–207. ACM Press, November 2022.

BLT+24. Renas Bacho, Julian Loss, Stefano Tessaro, Benedikt Wagner, and Chenzhi Zhu. Twinkle: Threshold signatures from DDH with full adaptive security. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part I*, volume 14651 of *LNCS*, pages 429–459. Springer, Cham, May 2024.

BLT25. Dan Boneh, Evan Laufer, and Ertem Nusret Tas. Threshold batch identity-based encryption without epochs. Cryptology ePrint Archive, Paper 2025/1254, 2025.

Bol03. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46. Springer, Berlin, Heidelberg, January 2003.

BTZ22. Mihir Bellare, Stefano Tessaro, and Chenzhi Zhu. Stronger security for non-interactive threshold signatures: BLS and FROST. Cryptology ePrint Archive, Report 2022/833, 2022.

CATZ24. Rutchathon Chairattana-Apirom, Stefano Tessaro, and Chenzhi Zhu. Partially non-interactive two-round lattice-based threshold signatures. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part IV*, volume 15487 of *LNCS*, pages 268–302. Springer, Singapore, December 2024.

CGM19. Yilei Chen, Nicholas Genise, and Pratyay Mukherjee. Approximate trapdoors for lattices and smaller hash-and-sign signatures. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 3–32. Springer, Cham, December 2019.

Che25a. Yanbo Chen. Dazzle: Improved adaptive threshold signatures from DDH. In Tibor Jager and Jiaxin Pan, editors, *PKC 2025, Part III*, volume 15676 of *LNCS*, pages 233–261. Springer, Cham, May 2025.

Che25b. Yanbo Chen. Round-efficient adaptively secure threshold signatures with rewinding. *CiC*, 2(2):16, 2025.

Che26. Yanbo Chen. Partially non-interactive two-round threshold and multi-signatures with tighter and adaptive security. Cryptology ePrint Archive, Paper 2026/373, 2026.

CKGW22. Deirdre Connolly, Chelsea Komlo, Ian Goldberg, and Christopher A. Wood. Two-Round Threshold Schnorr Signatures with FROST. Internet-Draft draft-irtf-cfrg-frost-10, Internet Engineering Task Force, September 2022. Work in Progress.

CKK+25. Elizabeth C. Crites, Jonathan Katz, Chelsea Komlo, Stefano Tessaro, and Chenzhi Zhu. On the adaptive security of FROST. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VI*, volume 16005 of *LNCS*, pages 480–511. Springer, Cham, August 2025.

DDFY94. Alfredo De Santis, Yvo Desmedt, Yair Frankel, and Moti Yung. How to share a function securely. In *26th ACM STOC*, pages 522–533. ACM Press, May 1994.

Des88. Yvo Desmedt. Society and group oriented cryptography: A new concept. In Carl Pomerance, editor, *CRYPTO'87*, volume 293 of *LNCS*, pages 120–127. Springer, Berlin, Heidelberg, August 1988.

DF90. Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 307–315. Springer, New York, August 1990.

DKM+24. Rafaël Del Pino, Shuichi Katsumata, Mary Maller, Fabrice Mouhartem, Thomas Prest, and Markku-Juhani O. Saarinen. Threshold raccoon: Practical threshold signatures from standard lattice assumptions. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 219–248. Springer, Cham, May 2024.

dPENP25. Rafael del Pino, Thomas Espitau, Guilhem Niot, and Thomas Prest. Simple and efficient lattice threshold signatures with identifiable aborts. Cryptology ePrint Archive, Paper 2025/871, 2025.

dPKPR24. Rafaël del Pino, Shuichi Katsumata, Thomas Prest, and Mélissa Rossi. Raccoon: A masking-friendly signature proven in the probing model. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part I*, volume 14920 of *LNCS*, pages 409–444. Springer, Cham, August 2024.

DR24. Sourav Das and Ling Ren. Adaptively secure BLS threshold signatures from DDH and co-CDH. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part VII*, volume 14926 of *LNCS*, pages 251–284. Springer, Cham, August 2024.

EEN+24. Muhammed F. Esgin, Thomas Espitau, Guilhem Niot, Thomas Prest, Amin Sakzad, and Ron Steinfeld. Plover: Masking-friendly hash-and-sign lattice signatures. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part VII*, volume 14657 of *LNCS*, pages 316–345. Springer, Cham, May 2024.

EKT24. Thomas Espitau, Shuichi Katsumata, and Kaoru Takemure. Two-round threshold signature from algebraic one-more learning with errors. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part VII*, volume 14926 of *LNCS*, pages 387–424. Springer, Cham, August 2024.

ENP24. Thomas Espitau, Guilhem Niot, and Thomas Prest. Flood and submerse: Distributed key generation and robust threshold signature from lattices. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part VII*, volume 14926 of *LNCS*, pages 425–458. Springer, Cham, August 2024.

EY24. Ehsan Ebrahimi and Anshu Yadav. Strongly secure universal thresholdizer. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part III*, volume 15486 of *LNCS*, pages 207–239. Springer, Singapore, December 2024.

FKL18. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Cham, August 2018.

FNR25. Sebastian Faller, Guilhem Niot, and Michael Reichle. Lattice-based threshold blind signatures. Cryptology ePrint Archive, Paper 2025/1566, 2025.

GCRS25. Paul Gerhart, Davide Li Calsi, Luigi Russo, and Dominique Schröder. Fully-adaptive two-round threshold schnorr signatures from DDH. Cryptology ePrint Archive, Paper 2025/1478, 2025.

GKS24. Kamil Doruk Gür, Jonathan Katz, and Tjerand Silde. Two-round threshold lattice-based signatures from threshold homomorphic encryption. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *PQCrypto 2024, Part II*, pages 266–300. Springer, Cham, June 2024.

GMPW20. Nicholas Genise, Daniele Micciancio, Chris Peikert, and Michael Walter. Improved discrete gaussian and subgaussian analysis for lattice cryptography. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 623–651. Springer, Cham, May 2020.

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

GRJK07. Rosario Gennaro, Tal Rabin, Stanislaw Jarecki, and Hugo Krawczyk. Erratum: Robust and efficient sharing of RSA functions. *Journal of Cryptology*, 20(3):393, July 2007.

GTJ+26. Liming Gao, Guofeng Tang, Dingding Jia, Yijian Liu, Bingqian Liu, Xianhui Lu, Kunpeng Wang, and Yongjian Yin. TalonG: Bandwidth-efficient two-round threshold signatures from lattices. In *EUROCRYPT*, 2026.

HKZ12. Daniel Hsu, Sham M. Kakade, and Tong Zhang. A tail inequality for quadratic forms of subgaussian random vectors. *Electronic Communications in Probability*, 17(52):1–6, 2012.

JTWZ26. Kaijie Jiang, Stefano Tessaro, Hoeteck Wee, and Chenzhi Zhu. Tweed: Adaptively secure lattice-based two-round threshold signatures. In Joan Daemen and Emmanuel Thomé, editors, *EUROCRYPT 2026, Part I*, volume 16541 of *LNCS*, pages 124–153. Springer, Cham, May 2026.

KG20. Chelsea Komlo and Ian Goldberg. FROST: Flexible round-optimized Schnorr threshold signatures. In Orr Dunkelman, Michael J. Jacobson, Jr., and Colin O'Flynn, editors, *SAC 2020*, volume 12804 of *LNCS*, pages 34–65. Springer, Cham, October 2020.

KLL25. John Kelsey, Nathalie Lang, and Stefan Lucks. Turning hash-based signatures into distributed signatures and threshold signatures: Delegate your signing capability, and distribute it among trustees. *CiC*, 2(2):24, 2025.

KRT24. Shuichi Katsumata, Michael Reichle, and Kaoru Takemure. Adaptively secure 5 round threshold signatures from MLWE/MSIS and DL with rewinding. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part VII*, volume 14926 of *LNCS*, pages 459–491. Springer, Cham, August 2024.

Mak22. Nikolaos Makriyannis. On the classic protocol for MPC schnorr signatures. Cryptology ePrint Archive, Paper 2022/1332, 2022.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Berlin, Heidelberg, April 2012.

MP13. Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with small parameters. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 21–39. Springer, Berlin, Heidelberg, August 2013.

Natnt. National Institute of Standards and Technology. Multi-Party Threshold Cryptography, 2018–Present. https://csrc.nist.gov/Projects/threshold-cryptography.

NT24. Sela Navot and Stefano Tessaro. One-more unforgeability for multi - and threshold signatures. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part I*, volume 15484 of *LNCS*, pages 429–462. Springer, Singapore, December 2024.

Pei06. Chris Peikert. On error correction in the exponent. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 167–183. Springer, Berlin, Heidelberg, March 2006.

Pei15. Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015.

PKN+25. Rafaël Del Pino, Shuichi Katsumata, Guilhem Niot, Michael Reichle, and Kaoru Takemure. Unmasking TRaccoon: A lattice-based threshold signature with an efficient identifiable abort protocol. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part VI*, volume 16005 of *LNCS*, pages 423–456. Springer, Cham, August 2025.

Ros20. Mélissa Rossi. *Extended security of lattice-based cryptography*. PhD thesis, Université Paris sciences et lettres, 2020.

Sho00. Victor Shoup. Practical threshold signatures. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 207–220. Springer, Berlin, Heidelberg, May 2000.

TT15. Katsuyuki Takashima and Atsushi Takayasu. Tighter security for efficient lattice cryptography via the Rényi divergence of optimized orders. In Man Ho Au and Atsuko Miyaji, editors, *ProvSec 2015*, volume 9451 of *LNCS*, pages 412–431. Springer, Cham, November 2015.

TZ23. Stefano Tessaro and Chenzhi Zhu. Threshold and multi-signature schemes from linear hash functions. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 628–658. Springer, Cham, April 2023.

YJW23. Yang Yu, Huiwen Jia, and Xiaoyun Wang. Compact lattice gadget and its applications to hash-and-sign signatures. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 390–420. Springer, Cham, August 2023.

ZT25. Chenzhi Zhu and Stefano Tessaro. The algebraic one-more MISIS problem and applications to threshold signatures. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part I*, volume 16000 of *LNCS*, pages 548–581. Springer, Cham, August 2025.

A Proof of Lemma 8

We first show the statement on the distribution $\mathcal{D}$. Since the vectors $\mathbf{r}_1, \ldots, \mathbf{r}_t$ are sampled independently with $\mathbf{r}_i \leftarrow \mathcal{D}_{\Lambda+\mathbf{s}_i,\sigma_r}$.By Lemma 7, If $t = k+1$, then $\mathbf{P} \otimes \mathbf{I}_m$ is invertible, so the statement on $\mathcal{D}$ holds exactly. For the remainder of this part, assume $t > k+1$, we just need to show that:

- $\phi_M(\mathbf{P} \otimes \mathbf{I}_m)$ has full-row-rank, and $\ker(\mathbf{P} \otimes \mathbf{I}_m)$ is a Λ^t-subspace.
- $\eta_{\varepsilon'}(\Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)) \le \sigma_r$, where $\varepsilon' = (1+\varepsilon)^{t-k-1} - 1$.

By the definition of $\mathbf{P}$, $\phi_M(\mathbf{P} \otimes \mathbf{I}_m)$ has full-row-rank. In order to show $\ker(\mathbf{P} \otimes \mathbf{I}_m)$ is a Λ^t-subspace, we just need to show $\mathsf{Span}(\Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)) = \ker(\mathbf{P} \otimes \mathbf{I}_m)$. For any $t \ge j > k+1$, we have $\mathbf{v}_j := \mathbf{e}_{k+1} - \mathbf{e}_j \in \ker(\mathbf{P}) \cap \mathcal{R}^t$. Let Λ_0 be the lattice generated by $(\mathbf{v}_j)_{j \in [k+2,t]}$, then we have $\Lambda_0 \otimes \Lambda \subseteq \Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)$. Note that $\ker(\mathbf{P} \otimes \mathbf{I}_m) = \mathsf{Span}(\Lambda_0 \otimes \Lambda) \subseteq \mathsf{Span}(\Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)) \subseteq \ker(\mathbf{P} \otimes \mathbf{I}_m)$, thus $\mathsf{Span}(\Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)) = \ker(\mathbf{P} \otimes \mathbf{I}_m)$.

For the second condition, we just need to show $\eta_{\varepsilon'}(\Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)) \le \eta_{\varepsilon'}(\Lambda_0 \otimes \Lambda) \le \sigma_r$, note that $\Lambda_0 \otimes \Lambda = \sum_{j=k+2}^{t} \mathbf{v}_j \otimes \Lambda$ and $\tilde{\mathbf{v}}_i \otimes \Lambda$ is the projection of $\mathbf{v}_i \otimes \Lambda$ orthogonal to $\mathsf{Span}(\mathbf{v}_{k+2} \otimes \Lambda + \cdots + \mathbf{v}_{i-1} \otimes \Lambda)$. By applying Lemma 9 $t-k-1$ times, we have $\eta_{\varepsilon'}(\Lambda_0 \otimes \Lambda) \le \max_{j \in [k+2,t]} \eta_\varepsilon(\tilde{\mathbf{v}}_j \otimes \Lambda)$. Note that for any $j \in [k+2, t]$ $\tilde{\mathbf{v}}_j = \frac{1}{j-k-1} \cdot \sum_{i=k+1}^{j-1} \mathbf{e}_i - \mathbf{e}_j$, thus we have $\|\tilde{\mathbf{v}}_j\|_1 = 2$. By Lemma 10, $\eta_\varepsilon(\tilde{\mathbf{v}}_j \otimes \Lambda) \le 2 \cdot \eta_\varepsilon(\Lambda) \le \sigma_r$, which implies $\eta_{\varepsilon'}(\Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)) \le \sigma_r$.

For the statement on $\mathcal{D}_1$ following the similar proof as above, where we used $(\mathbf{P}_1 \otimes \mathbf{I}_m)\Lambda^t = \Lambda^k$.

For $\mathcal{D}_{2|\mathbf{x}_0}$, let $\mathcal{Q} := \mathcal{D}_{(\mathbf{P} \otimes \mathbf{I}_m)\Lambda^t + \mathbf{c}, \Sigma \otimes \mathbf{I}_m}$. We first compute the conditional distribution of $\mathbf{y}_1$ under $\mathcal{Q}$ given $\mathbf{x} = \mathbf{x}_0$, for any $\mathbf{x}_0 \in \Lambda^k + \mathbf{a}_1$, and then relate it to $\mathcal{D}_{2|\mathbf{x}_0}$.

Firstly, we know the support of $\mathbf{y}_1$ is $\mathsf{Supp} := \{\mathbf{y}_1 : (\mathbf{y}_1, \mathbf{x}_0)^\top \in \Lambda^{k+1} + \mathbf{c}\} = \Lambda + \mathbf{s}_1$, since $(\mathbf{P} \otimes \mathbf{I}_m)\Lambda^t = \Lambda^{k+1}$. It is left to show the probability of $\mathbf{y}_1$ conditioning on $\mathbf{x} = \mathbf{x}_0$. The probability of $\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{x}_0 \end{pmatrix}$ is proportional to

$$\begin{aligned}
\rho_{\Sigma \otimes \mathbf{I}_m}((\mathbf{y}_1, \mathbf{x}_0)^\top) &= \exp(-\pi \langle (\mathbf{y}_1, \mathbf{x}_0)^\top, (\Sigma^{-1} \otimes \mathbf{I}_m) \cdot (\mathbf{y}_1, \mathbf{x}_0)^\top \rangle) \\
&= \exp(-\pi \big(\langle \mathbf{y}_1, V_{11}\mathbf{y}_1 \rangle - 2\langle \mathbf{y}_1, -(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0 \rangle + \langle \mathbf{x}_0, (\mathbf{V}_{22} \otimes \mathbf{I}_m)\mathbf{x}_0 \rangle \big) \\
&= \exp(-\pi \big(\langle \mathbf{y}_1 + V_{11}^{-1}(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0), V_{11}(\mathbf{y}_1 + V_{11}^{-1}(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0) \rangle \\
&\quad + \langle \mathbf{x}_0, (\mathbf{V}_{22} \otimes \mathbf{I}_m)\mathbf{x}_0 \rangle - V_{11}\langle V_{11}^{-1}(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0, V_{11}^{-1}(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0 \rangle \big))
\end{aligned}$$

Thus the probability of $\mathbf{y}_1$ conditioning on $\mathbf{x} = \mathbf{x}_0$ is proportional to

$$\exp(-\pi \big(\langle \mathbf{y}_1 + V_{11}^{-1}(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0), V_{11}(\mathbf{y}_1 + V_{11}^{-1}(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0) \rangle \big),$$

which shows that $\mathcal{Q}_{2|\mathbf{x}_0}$ is exactly $\mathcal{D}_{\Lambda+\mathbf{s}_1, V_{11}^{-1}\mathbf{I}_m, -V_{11}^{-1}(\mathbf{V}_{21} \otimes \mathbf{I}_m)\mathbf{x}_0}$.

It remains to compare this distribution with $\mathcal{D}_{2|\mathbf{x}_0}$. Let $\Lambda_{\mathsf{ker}} := \Lambda^t \cap \ker(\mathbf{P} \otimes \mathbf{I}_m)$. For each $\mathbf{z} = (\mathbf{y}_1, \mathbf{x}_0)^\top$ in the same support, and we have

$$\mathcal{D}(\mathbf{z}) \propto \rho_{\Sigma \otimes \mathbf{I}_m}(\mathbf{z})\, \rho_{\sigma_r}(\Lambda_{\mathsf{ker}} + \mathbf{b}_{\mathbf{z}}) \ ,$$

where $\mathbf{b}_{\mathbf{z}} \in \ker(\mathbf{P} \otimes \mathbf{I}_m)$ and the proportionality constant is independent of $\mathbf{z}$. For $t > k+1$, the smoothing bound established above and Poisson summation imply $(1-\varepsilon')C \le \rho_{\sigma_r}(\Lambda_{\mathsf{ker}} + \mathbf{b}_{\mathbf{z}}) \le (1+\varepsilon')C$ for a constant $C > 0$ independent of $\mathbf{z}$, where $\varepsilon' = (1+\varepsilon)^{t-k-1} - 1 < (1+\varepsilon)^{t-k} - 1$. Therefore, fixing $\mathbf{x} = \mathbf{x}_0$ and normalizing over $\mathbf{y}_1 \in \Lambda + \mathbf{s}_1$ yields $\frac{1-\varepsilon'}{1+\varepsilon'} \le \frac{\mathcal{D}_{2|\mathbf{x}_0}(\mathbf{y}_1)}{\mathcal{Q}_{2|\mathbf{x}_0}(\mathbf{y}_1)} \le \frac{1+\varepsilon'}{1-\varepsilon'} \le 1 + \varepsilon''_{t,k}$. Hence $\mathcal{D}_{2|\mathbf{x}_0} \overset{\varepsilon''_{t,k}}{\approx} \mathcal{Q}_{2|\mathbf{x}_0}$. When $t = k+1$, $\Lambda_{\mathsf{ker}} = \{\mathbf{0}\}$ the conditional distributions are identical. This proves the statement. □