# BMRaccoon: Blind Multi-Signatures from Lattices

Keerthi Aiswarya Varshini[1*], Kanchan Bisht[1*], Palkesh Jain[1], Maria Francis[1], and R. Kabaleeshwaran[2]

[1] Indian Institute of Technology Hyderabad
keerthi.av@prjt.cse.iith.ac.in
kanchan.bisht@prjt.cse.iith.ac.in
cs25resch12002@iith.ac.in
mariaf@cse.iith.ac.in

[2] Indian Institute of Information Technology Design and Manufacturing, Kurnool
kabaleesh@iiitk.ac.in

**Abstract.** Blind multi-signatures (BMS) allow a user to obtain a compact signature on a hidden message through the joint participation of multiple signers, providing a basis for decentralized issuance of anonymous tokens.We present BMRaccoon, the first lattice-based post-quantum BMS construction, with a two-round signing protocol secure under concurrent sessions and signatures syntactically compatible with Raccoon under the sum of the signers' verification keys.

As a building block, we introduce MRaccoon, the first Raccoon-based multi-signature and prove its strong unforgeability under the Algebraic One-More MISIS (AOM-MISIS) and MSIS assumptions in the random oracle and and knowledge-of-secret-key (KOSK) model. Our modular security analysis establishes one-more unforgeability of BMRaccoon via a straight-line reduction to the strong unforgeability of MRaccoon, modelling the user's blinding contribution as an auxiliary signer with a zero verification key. We additionally prove blindness against malicious, colluding signers. For up to $q_s = 2^{32}$ signing sessions and $n = 32$ signers, our concrete instantiation yields signatures of approximately 67 KB and verification keys of approximately 15 KB.

**Keywords:** Concurrent security, Blind multi-signatures, Lattices, Raccoon signatures.

## 1 Introduction

*Blind signatures.* Blind signatures [Cha83] allow a user to obtain a signature on a message from a signer such that the signer learns nothing about the message and cannot later link the resulting signature to the session in which it was produced (*blindness*). At the same time, the user must not be able to output more valid signatures than the number of signing sessions it has completed (*one-more unforgeability*). These properties make blind signatures a fundamental primitive for privacy-preserving applications such as electronic voting, anonymous credentials, and anonymous tokens.

*Multi-signatures.* A multi-signature scheme [BN06] enables a group of $n$ signers, each holding an independently generated key pair $(sk_i, vk_i)$, to jointly produce a single compact signature $\sigma$ on a common message $m$. This compactness makes multi-signatures attractive for blockchains, where a single signature replaces $n$ individual signatures, reducing on-chain storage, verification costs, and transaction fees.

A central security challenge in the multi-signer setting is the *rogue-key attack* [BN06], in which a malicious signer chooses its verification key as a function of the honest signers' keys so that it can forge a multi-signature without their participation. Security against such attacks is analyzed under different key-registration models. In the *knowledge-of-secret-key* (KOSK) model [RY07, LOS+06, Bol02], the adversary must know the secret key corresponding to every verification key it controls. In practice, this is enforced by requiring a proof of possession at key registration, which is natural wherever signers must enroll in a registry before they can sign. In contrast, the *plain public-key* (PPK) model [BN06] places no such requirement on adversarially generated keys, making it suitable for open settings such as blockchains, where trusted key registration may be unavailable.

* Equal contribution.

*Blind multi-signatures (BMS).* A standard blind signature scheme has a single point of trust. In decentralized settings such as decentralized autonomous organizations (DAOs), authorization may instead require the approval of every member of a designated group. Karantaidou et al. [KRB+24] formalized this functionality as *blind multi-signatures* (BMS), which combine the privacy of blind signatures with the collective authorization of multi-signatures. In a BMS, a user interacts with a group of signers to obtain a single compact signature on a message that remains hidden from all of them. Its security is captured by two properties: (*i*) *blindness*, which ensures that even colluding signers cannot link a signature to the session in which it was generated, and (*ii*) *one-more unforgeability* (OMUF), which prevents a malicious user controlling all but one signer from producing more valid signatures than the number of signing sessions completed with the honest signer.

*Concurrent security.* Ensuring security under concurrent execution is a fundamental challenge for interactive signing protocols, as an adversary may interleave multiple sessions to forge signatures even against protocols that are provably secure under sequential execution. It is therefore a minimal requirement for real-world deployments where sessions execute concurrently. For blind signatures, this threat is particularly significant in schemes derived from identification protocols, in which the signer commits to a nonce before the challenge is determined. An adversary that collects the commitments of $l$ concurrent sessions can choose correlated challenges for them, and then combine the $l$ responses into $l + 1$ valid signatures. In blind Schnorr signatures [CP92], the problem of finding such suitable challenges is formalized by the ROS problem [Sch01]. Wagner's generalized birthday algorithm [Wag02] solves ROS in subexponential time. Benhamouda et al. [BLL+22] later gave a polynomial-time attack against ROS when the number of concurrent sessions exceeds the bit length of the group order. Katsumata et al. [KLR24] extended this line of attack to blind signatures based on parallel repetition, including lattice and isogeny-based constructions.

Multi-signatures are vulnerable to an analogous attack that exploits nonce aggregation. In two-round protocols, a malicious co-signer can choose its nonces after observing the honest signers' commitments, thereby biasing the aggregate nonce and the resulting challenge. Drijvers et al. [DEF+19] exploited this vulnerability using Wagner's algorithm to break several two-round multi-signature schemes whose security analyses did not account for concurrent sessions. Subsequent constructions address this attack by having each signer commit to multiple nonces and combining them using hash-derived coefficients, as in MuSig2 [NRS21] and its lattice-based counterpart MuSig-L [BTT22].

Blind multi-signatures must address both vulnerabilities simultaneously. An adversary acting as the user may also control all but one of the $n$ signers, allowing it to correlate challenges across concurrent sessions, as in ROS attacks [BLL+22], while adaptively choosing the corrupted signers' nonces to manipulate the aggregate nonce, as in multi-signature attacks [DEF+19]. Achieving concurrent security for BMS therefore requires a careful design and security analysis that account for both attack strategies and their interaction.

### 1.1 Related work

The only formally analyzed constructions of BMS are by [KRB+24] and BMuSig2 [BVS+26]. Karantaidou et al. [KRB+24], motivated by anonymous tokens with decentralized issuance, gave two constructions, one based on BLS signatures and one pairing-free, both proven secure in the algebraic group model. Subsequently, BMuSig2 [BVS+26] proposed a BMS whose signatures are standard Schnorr signatures. It makes the two-round Schnorr multi-signature MuSig2 [NRS21] blind by building on the techniques of concurrently secure blind Schnorr signatures of Fuchsbauer and Wolf [FW24], and reduces its one-more unforgeability (OMUF) to the strong unforgeability of MuSig2. We note, however, that the blindness argument of BMuSig2 contains an error. The nonce coefficient is computed as a deterministic hash of the signers' nonce commitments and the message. A signer that later sees a message–signature pair can therefore recompute this coefficient using the nonce commitments from each session it took part in, and check which session is consistent with the signature. This links the signature to the session in which it was issued, and so breaks blindness.

*The post-quantum gap.* All existing BMS constructions [KRB+24, BVS+26] rely on classical hardness assumptions and are therefore insecure against quantum attacks. Lattices are a leading foundation for post-quantum cryptography, with established lines of work on both blind signatures [AKSY22, dPK22, LNP22, BLNS23, JS25, HMM+26] and multi-signatures [DOTT21, BTT22, Che23, LCL26]. Nevertheless, no lattice-based BMS in the model of [KRB+24] is known to date. As we discuss below, efficient

lattice-based blind signatures and multi-signatures follow different design paradigms, which makes it non-trivial to combine blindness with compact signature aggregation.

*Lattice-based blind signatures.* Most round-optimal lattice-based blind signatures follow Fischlin's paradigm [Fis06], in which the final signature is a non-interactive zero-knowledge (NIZK) proof that the user knows a valid signature on the message under an underlying signature scheme. Such constructions are not *syntactically compatible* with the underlying signature scheme.[2] Verification therefore requires support for the underlying proof system rather than relying solely on the signature verification algorithm. Moreover, these constructions do not naturally extend to the multi-signer setting.[3]

A second line of work blinds Fiat–Shamir with aborts (FSwA) signatures, which use rejection sampling to hide the secret key. As in blind Schnorr signatures, the user shifts the signer's commitment and the challenge by blinding values and later unblinds the response. In the lattice setting, however, these blinding values must be short and therefore cannot hide perfectly, so the user must also rejection-sample, first the blinded challenge and then the unblinded response. The first rejection only requires local resampling, but the second occurs after the signer has responded and forces a restart with a fresh commitment. The user must then prove that the failure is genuine; otherwise, it could claim failure after obtaining a valid signature and thereby obtain an extra one. Rückert [Rüc10] proposed the first scheme of this type, followed by more efficient variants such as BLAZE [AAEBB20] and BlindOR [AAHJ21]. However, Hauck et al. [HKLN20] identified a gap in the proof of [Rüc10], and their corrected framework is secure only for a polylogarithmic number of sessions.

BRaccoon [HMM+26] builds on the Fiat–Shamir-style lattice signature Raccoon [dPKPR24] to obtain concurrently secure blind signatures. Raccoon replaces rejection sampling with noise flooding, eliminating signing aborts. Unlike Schnorr-style blinding, which shifts the challenge and may violate Raccoon's shortness requirement, BRaccoon hides the challenge using linearly *homomorphic encryption* (HE). The signer computes its response on the encrypted challenge, and the user decrypts and unblinds the result to obtain a standard Raccoon signature. BRaccoon also employs commitments and NIZK arguments, following Fuchsbauer and Wolf [FW24], to bind the hidden challenge to a message fixed at the start of each session. The resulting scheme is concurrently secure and syntactically compatible with Raccoon, but supports only a single signer.

*Lattice-based multi-signatures.* Efficient two-round lattice-based multi-signatures [DOTT21,BTT22, Che23, LCL26] follow the FSwA paradigm, whose linear structure allows the partial signatures of the signers to be combined into a single compact signature. To resist the concurrent attacks of Drijvers et al. [DEF+19], most of these protocols build on ideas from the Schnorr-based multi-signature MuSig2 [NRS21].

In particular, the challenge is computed from the aggregated nonce, and each signer forms its response by combining its own nonces with the same coefficients. The final signature is the sum of the partial signatures. However, lattice-based signing requires each partial signature to be rejection-sampled, and since the challenge depends on all signers' nonces, a rejection by any one signer restarts the entire session with fresh nonces. The blind setting makes this harder still. Because the signing request is encrypted under the user's blinding, signers compute their responses without seeing them in the clear and cannot perform the rejection sampling that keeps their secret keys hidden. Nor can this step be delegated to the user, who sees every response, including those that should have been rejected and that therefore leak information about the signers' secret keys. Therefore, incorporating blindness into these protocols may further complicate the design, as it must preserve the privacy of the signing request while securely handling aborts.

This leads to the following question of this paper:

> *Q: Can we construct a concurrently secure lattice-based blind multi-signature that is syntactically compatible with its underlying signature scheme?*

[2] Following [HMM+26], we call a blind signature scheme *syntactically compatible* with its underlying signature scheme if its outputs have the same algebraic form as signatures under that scheme and are accepted by its verification algorithm, possibly with enlarged parameters.

[3] Generic aggregation is possible, for instance with a succinct non-interactive argument of knowledge (SNARK) of $n$ such proofs, but it incurs substantial overhead. A related approach has been used to aggregate Falcon signatures [AAB+24, CDBT26].

### 1.2 Our Contributions

We answer this question affirmatively. Our contributions are as follows.

- **Raccoon multi-signature.** We construct MRaccoon, the first two-round Raccoon-based multi-signature and prove it strongly unforgeable in the KOSK model under Algebraic One-More Module Inhomogeneous Short Integer Solution (AOM-MISIS) [ZT25] and Module Short Integer Solution (MSIS) in the random oracle model (ROM).
- **Lattice-based BMS.** By blinding MRaccoon, we obtain BMRaccoon, the first lattice-based blind multi-signature. Its two-round (four-move) signing protocol is concurrently secure and outputs a syntactically compatible Raccoon signature under the sum of the signers' verification keys.
- **Modular security analysis.** We capture the user's blinding by an intermediate multi-signature scheme MRaccoon$^\star$ and reduce the one-more unforgeability of BMRaccoon to the strong unforgeability of MRaccoon via MRaccoon$^\star$, without rewinding or programming the signing random oracles. We also prove blindness against malicious, colluding signers.
- **Concrete parameters.** We instantiate at 128-bit computational and 40-bit statistical security. With $n = 32$ signers and $q_s = 2^{32}$ signing sessions, the resulting signatures and verification keys are 67.92 KB and 15.46 KB, respectively. Increasing the number of signers to $n = 64$ incurs only a marginal increase of approximately 1% in signature size and 4% in verification-key size.

## 2 Technical Overview

Our construction follows the blueprint of two-round (four-move) BMS protocol BMuSig2 [BVS$^+$26], which adapts the two-round MuSig2 multi-signature scheme [NRS21] to the blind setting using the technique of Fuchsbauer and Wolf [FW24]. The central idea is for the user to commit to the message and blinding factors in the first round, before receiving any signer nonces. Upon receiving the nonces, the user aggregates them to obtain a joint nonce and randomizes it using the committed blinding randomness. The user then computes the signing challenge with respect to the randomized nonce and the committed message, and masks the challenge using the blinding factor. Along with the masked challenge, the user provides a zero-knowledge proof demonstrating that the challenge and all other protocol messages sent to the signers are correctly derived from the initially committed values. The signers use the challenge to generate their respective partial signatures, which the user then aggregates and randomizes with the corresponding blinding factor to obtain the final signature.

Intuitively, concurrent security follows because the initial commitment prevents a malicious user from adaptively choosing challenges across concurrent sessions, which is precisely the freedom exploited by ROS-style attacks. The user's only remaining freedom lies in the nonces of the co-signers it corrupts, and the concurrent security of the underlying multi-signature scheme already accounts for this behavior.

*Why Raccoon?* A lattice analogue of this blueprint requires a blind signature and a multi-signature built on the *same* lattice signature scheme, whose algebraic structure supports both blinding and aggregation. Raccoon [dPKPR24] is a natural choice for three reasons. First, it is abort-free: it relies on noise flooding instead of rejection sampling, so signing never aborts. This property is particularly important in interactive settings, where an abort by any single party would force all participants to restart the protocol. Second, its response is linear in the secret key and the nonce, so partial responses from multiple signers can be summed directly. Third, a blind variant of Raccoon, i.e. BRaccoon [HMM$^+$26] already exists, so what remains is a compatible multi-signature. Existing Raccoon-based multi-party schemes are threshold signatures [DPKM$^+$24, DPKN$^+$25, BKL$^+$24], in which the signers hold shares of a single secret key generated by a trusted dealer or a distributed key generation protocol. To our knowledge, no Raccoon-based multi-signature has been proposed in which each signer generates its own key pair and any ad-hoc group of signers can sign jointly.

*Raccoon.* We first briefly recall the Raccoon signature scheme [dPKPR24]. Let $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N + 1)$ and let $\mathbf{A} = [\hat{\mathbf{A}} \,\|\, \mathbf{I}_k] \in \mathcal{R}_q^{k \times m}$ be a public matrix. The signing key is a short vector $\mathbf{s} \leftarrow \mathcal{D}^m_{\sigma_{\mathsf{sk}}}$ sampled from a discrete Gaussian distribution, and the verification key is $vk := \lfloor \mathbf{As} \rceil_{\nu_t}$, where $\lfloor \cdot \rceil_\nu$ drops the $\nu$ low-order bits of each coefficient. To sign a message $\mathbf{m}$, the signer samples $\mathbf{r} \leftarrow \mathcal{D}^m_{\sigma_{\mathsf{w}}}$, computes the rounded nonce $\tilde{\mathbf{w}} := \lfloor \mathbf{Ar} \rceil_{\nu_{\mathsf{w}}}$, and derives a short challenge vector $c := \mathsf{H}(\tilde{\mathbf{w}}, vk, \mathbf{m})$, where $\mathsf{H} : \{0,1\}^* \to \mathcal{S}_c$ is modelled as a random oracle and $\mathcal{S}_c$ denotes the set of ternary polynomials with exactly $\omega$ nonzero coefficients. The signer then computes the response $\mathbf{z} := (c\,\mathbf{s} + \mathbf{r})_{[m-k]}$,

consisting of the first $m-k$ coordinates of $c\,\mathbf{s}+\mathbf{r}$, together with a hint $\mathbf{h} := \tilde{\mathbf{w}} - \lfloor \hat{\mathbf{A}}\mathbf{z} - 2^{\nu_t} c\, vk \rceil_{\nu_w}$, which absorbs the omitted coordinates and the rounding errors. A signature $\sigma := (c, \mathbf{z}, \mathbf{h})$ is valid if $\big\|(\mathbf{z},\, 2^{\nu_w}\mathbf{h})\big\|_2 \le \beta$ and $c = \mathsf{H}\big(\lfloor \hat{\mathbf{A}}\mathbf{z} - 2^{\nu_t} c\, vk \rceil_{\nu_w} + \mathbf{h},\, vk,\, \mathbf{m}\big)$.

*Roadmap.* We first construct the multi-signature MRaccoon and prove it is strongly unforgeable. We then blind it to obtain BMRaccoon. Finally, we explain why the blinding prevents a direct reduction to MRaccoon, and how the intermediate scheme MRaccoon$^\star$ bridges this gap.

### 2.1 MRaccon: Raccoon-based multi-signature

We extend Raccoon to a two-round multi-signature scheme, MRaccoon for a set of $n$ signers with verification key set $K = \{vk_1, \ldots, vk_n\}$. Each signer $\mathcal{S}_i$ independently generates a Raccoon key pair $(sk_i, vk_i) := (\mathbf{s}_i, \lfloor \mathbf{t}_i \rceil_{\nu_t})$, where $\mathbf{t}_i = \mathbf{A}\mathbf{s}_i$. To resist the concurrent attacks we aggregate the signers' nonces through a linear combination with signed-monomial coefficients following the nonce aggregation approach of [EKT24].

In the first round, each signer $\mathcal{S}_i$ samples $\mathbf{r}_i^{(0)}, \ldots, \mathbf{r}_i^{(\ell)}$ and broadcasts commitment containing $\ell+1$ nonces as $com_i := (\mathbf{A}\mathbf{r}_i^{(j)})_{j\in[0,\ell]}$. In the second round, each signer obtains the ordered list of commitments $W := (com_i)_{i\in[n]}$ and independently derives the common nonce coefficients $(b^{(1)}, \ldots, b^{(\ell)}) := \mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, K) \in \mathcal{S}_{\mathsf{b}}^{\ell}$, where $\mathcal{S}_{\mathsf{b}} := \{\pm 1, \pm X, \ldots, \pm X^{N-1}\}$. With $b^{(0)} := 1$, they compute the aggregated nonce $\tilde{\mathbf{w}} := \Big\lfloor \sum_{j=0}^{\ell} b^{(j)} \sum_{i\in[n]} \mathbf{A}\mathbf{r}_i^{(j)} \Big\rceil_{\nu_w}$, and the challenge $c := \mathsf{H}_{\mathsf{sig}}(\tilde{\mathbf{w}}, \mathbf{m}, K)$, and each signer returns $\mathbf{z}_i := \big(c\,\mathbf{s}_i + \sum_{j=0}^{\ell} b^{(j)} \mathbf{r}_i^{(j)}\big)_{[m-k]}$. The aggregator sets $\mathbf{z} := \sum_i \mathbf{z}_i$ and computes the hint $\mathbf{h}$ with respect to the aggregated key $\tilde{\mathbf{t}} := \sum_i vk_i \bmod q_{\nu_t}$. The final signature $\sigma = (c, \mathbf{z}, \mathbf{h})$ is a syntactically compatible Raccoon signature under $\tilde{\mathbf{t}}$.

We make two observations about the construction. First, verification keys are rounded, and the sum of rounded keys is not the rounding of the sum. We show that $2^{\nu_t}\tilde{\mathbf{t}} = \sum_i \mathbf{t}_i + \boldsymbol{\varepsilon}_t \pmod q$ with $\|\boldsymbol{\varepsilon}_t\|_\infty \le (3n-1)2^{\nu_t - 1}$. The hint absorbs this error, at the cost of a verification bound $\beta_{\mathsf{z}}$ that grows linearly in number of signers $n$. Second, the nonce coefficients are drawn from the signed monomials $\mathcal{S}_{\mathsf{b}}$ rather than from the challenge set $\mathcal{S}_{\mathsf{c}}$. The difference of two distinct signed monomials is invertible up to a factor of 2 with a ternary inverse (Lemma 1), which keeps the extracted solution short in the security proof. However, since a signed monomial carries only $\log N$ bits of entropy, we use $\ell = \lceil 2\lambda / \log N \rceil$ coefficients, so that distinct $\mathsf{H}_{\mathsf{non}}$ inputs receive distinct coefficient vectors except with negligible probability.

*Security.* We analyze the security of MRaccoon in the KOSK model, assuming a single honest signer. The adversary controls all remaining signers and may initiate arbitrarily many concurrent signing sessions with the honest signer. To win the game, it must output a valid forgery together with the secret keys corresponding to all corrupted verification keys in the forged key set. We prove MRaccoon existentially unforgeable under the algebraic one-more MISIS (AOM-MISIS) assumption [ZT25] (Theorem 2). In AOM-MISIS, the adversary is given target vectors $\mathbf{p}_i = \mathbf{A}\mathbf{s}_i$ for short secrets $\mathbf{s}_i$, and access to an oracle $\mathsf{PI}$ that returns $\sum_i d_i \mathbf{s}_i$ on input any vector $\mathbf{d} = (d_i)_i$. Each answer is a short preimage of $\sum_i d_i \mathbf{p}_i$, so preimages of any combination in the span of the queried vectors can be obtained trivially. To win, the adversary must instead output a triple $(\hat{\mathbf{s}}, \hat{\mathbf{b}}, \mathbf{u})$ such that $\hat{\mathbf{s}}$ is short and satisfies $\mathbf{A}\hat{\mathbf{s}} = \sum_i \hat{b}_i \mathbf{p}_i$ for a coefficient vector $\hat{\mathbf{b}} = (\hat{b}_i)_i$ outside this span. This is certified by the short vector $\mathbf{u}$, which is orthogonal to every queried $\mathbf{d}$ but not to $\hat{\mathbf{b}}$.

This assumption is a natural fit for MRaccoon, since each partial signature is a linear combination of the signer's secret key and its nonces. The reduction embeds the first target as the honest signer's key, setting $vk_1 := \lfloor \mathbf{p}_1 \rceil_{\nu_t}$, and uses the remaining targets as the $\ell+1$ nonces of each signing session. To answer a signing query, it makes a single $\mathsf{PI}$ query whose coefficient vector contains $c$ at the key position and $(1, b^{(1)}, \ldots, b^{(\ell)})$ at the session's nonce positions. The answer is precisely the honest partial signature, so signing queries are simulated without the secret key.

To extract a solution, the reduction forks the adversary at the $\mathsf{H}_{\mathsf{sig}}$ query of its forgery and obtains two forgeries $(c^*, \mathbf{z}^*, \mathbf{h}^*)$ and $(c^{*\prime}, \mathbf{z}^{*\prime}, \mathbf{h}^{*\prime})$ on the same aggregated nonce, message and key set, with $c^* \ne c^{*\prime}$. Let $\bar{c} := c^* - c^{*\prime}$ and $\bar{\mathbf{z}} := \mathbf{z}^* - \mathbf{z}^{*\prime}$. Since both forgeries reconstruct the same nonce, $\hat{\mathbf{A}}\bar{\mathbf{z}}$ is close to $\bar{c}\,(\mathbf{p}_1 + \sum_{i\ge 2} \mathbf{t}_i)$: the difference consists only of hint and rounding terms. Hence $\big(\bar{\mathbf{z}},\, \bar{c}\,(\mathbf{p}_1 + \sum_{i\ge 2} \mathbf{t}_i) - \hat{\mathbf{A}}\bar{\mathbf{z}}\big)$ is a short preimage of $\bar{c}\,(\mathbf{p}_1 + \sum_{i\ge 2} \mathbf{t}_i)$. Subtracting $\bar{c}\sum_{i\ge 2} sk_i$, which the

reduction knows in the KOSK model, yields a short $\hat{\mathbf{s}}$ with $\mathbf{A}\hat{\mathbf{s}} = \bar{c}\,\mathbf{p}_1$, that is, $\hat{\mathbf{b}} = (\bar{c}, 0, \ldots, 0)$. In the PPK model, the adversarially chosen $sk_i$'s are unknown and cannot be removed directly, so the extracted vector may not yield an AOM-MISIS solution.

It remains to construct a short vector $\mathbf{u}$ that is orthogonal to the $\mathsf{PI}$ queries in both executions, but not to the extracted coefficient vector $\hat{\mathbf{b}}$. The reduction sets the key entry of $\mathbf{u}$ to 2, ensuring that $\hat{\mathbf{b}}^\top \mathbf{u} = 2\bar{c} \neq 0$, and constructs the remaining entries session by session to cancel the contribution $2c$ from each query. A difficulty arises when the same session is queried in both executions with different coefficient vectors, since a single session block must be orthogonal to both. If the vectors differ in some nonce coefficient $b^{(k)} \neq \tilde{b}^{(k)}$, a short inverse of $b^{(k)} - \tilde{b}^{(k)}$ (Lemma 1) allows the reduction to construct a short block orthogonal to both queries. The remaining case, identical coefficients but different challenges, cannot occur as the reduction programs each session's $\mathsf{H}_{\mathsf{sig}}$ value already when answering the corresponding $\mathsf{H}_{\mathsf{non}}$ query, so identical coefficients imply that the session's challenge was fixed before the fork, and this session cannot be the one forked on because the forged message was never signed.

Strong unforgeability (Theorem 3) of $\mathsf{MRaccoon}$ follows with one additional step. If two of the adversary's signatures on the same pair reconstruct the same nonce, they share a challenge, and their difference is a short non-zero vector in the kernel of $\mathbf{A}$, i.e., an MSIS solution. Otherwise, some signature corresponds to a triple $(K, \mathbf{m}, \tilde{\mathbf{w}})$ that no signing session produced, and the argument above applies to it.

### 2.2 BMRaccoon : Blinding MRaccoon

We obtain our BMS scheme $\mathsf{BMRaccoon}$ by blinding $\mathsf{MRaccoon}$ in the style of $\mathsf{BRaccoon}$ [HMM+26]. The user acts as the aggregator and must prevent the signers from linking the final signature to their session. There are specifically three values computed by the user that can help signers link a signature to its session.

- *The aggregated nonce.* Every signer sees all nonce lists $W$ of the session, so in $\mathsf{MRaccoon}$ it knows the aggregated nonce $\tilde{\mathbf{w}}$. Since anyone can recompute $\tilde{\mathbf{w}}$ from a signature $(c, \mathbf{z}, \mathbf{h})$ during verification, a signer could match a published signature to the session with the same nonce. To prevent this, the user samples a blinding vector $\mathbf{r}' \leftarrow D^m_{\sigma_{\mathsf{w}'}}$ from a wider Gaussian, shifts the aggregated nonce by $\mathbf{A}\mathbf{r}'$ before deriving the challenge, and shifts the aggregated response $\mathbf{z}$ by $\mathbf{r}'_{[m-k]}$. The signature remains valid, since both shifts correspond to the same additional nonce randomness $\mathbf{r}'$. The published nonce and response now depend on the session only through $\mathbf{r}_{agg} + \mathbf{r}'$, where $\mathbf{r}_{agg}$ is the signers' combined nonce randomness. Since $\sigma_{\mathsf{w}'}$ is much larger than $\|\mathbf{r}_{agg}\|$, this sum is statistically close to $\mathbf{r}'$ alone and reveals nothing about the session.
- *The challenge $c$.* The challenge appears in the clear in the final signature, so a signer that saw $c$ during signing could identify the session immediately. Blind Schnorr signatures avoid this by sending the signer a shifted challenge $c + \beta$ for a uniformly random $\beta$. However, this approach fails in lattices as the challenge must lie in the set $\mathcal{S}_{\mathsf{c}}$ and a shifted challenge is no longer short. Therefore the user instead sends $c$ encrypted under a linearly homomorphic encryption scheme, and each signer computes its response on the ciphertext without learning $c$.
- *The coefficients* $\mathbf{b} = (b^{(1)}, \ldots, b^{(\ell)})$. This issue is specific to the multi-signer setting. Since $\mathbf{b} = \mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \hat{K})$ is a deterministic hash of the transcript and the message, a signer that saw $\mathbf{b}$ could link any published $(\mathbf{m}, \sigma)$ to its session by recomputing $\mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \hat{K})$ for each stored transcript $W$. The user therefore encrypts $\mathbf{b}$ alongside $c$.

Encrypting both suffices because the partial signature is linear in $(c, \mathbf{b})$, with the signer's secrets as coefficients, so each signer can compute its response directly on the ciphertexts.

*Construction.* Signing is a two-round (four-move) protocol. In the first round, the user samples the blinding vector $\mathbf{r}'$ and an encryption key pair $(ek, dk)$, and sends $ek$ together with an extractable commitment $C$ to $(\mathbf{r}', \mathbf{m})$. Each signer $S_i$ replies with its nonce list $com_i$. In the second round, the user computes the coefficients $\mathbf{b}$, the shifted aggregated nonce $\tilde{\mathbf{w}}$ and the challenge $c$. It sends the encryptions $ctx_{\mathbf{b}}$ and $ctx_c$, together with a NIZK argument $\pi_1$ showing that these values are honestly derived from the committed $(\mathbf{r}', \mathbf{m})$ and the received nonces, and that the ciphertexts encrypt exactly these values. Each signer verifies $\pi_1$ and evaluates the response homomorphically on $ctx_c$, $ctx_{\mathbf{b}}$ and a

fresh encryption of $\mathbf{r}^{(0)}_{i,[m-k]}$, obtaining an encryption $ctx_{\mathbf{z}_i}$ of its partial signature. It returns $ctx_{\mathbf{z}_i}$ with a NIZK argument $\pi_{i,2}$ showing that its key is well formed, its nonces are short and the evaluation was performed honestly. Finally, the user verifies all arguments, decrypts and sums the partial signatures, adds $\mathbf{r}'_{[m-k]}$, and computes the hint $\mathbf{h}$. The resulting signature $(c, \mathbf{z}, \mathbf{h})$ is a Raccoon signature under $\tilde{\mathbf{t}} = \sum_i vk_i$, with a norm bound enlarged to absorb $\mathbf{r}'$.

On the user's side, the commitment and $\pi_1$ bind the challenge, the coefficients and the shift to a message fixed before any nonce is seen. As discussed before, this prevents a malicious user from adapting its challenges across concurrent sessions. On the signer's side, the evaluation must be *circuit private*, since the user holds $dk$, so that the evaluated ciphertext reveals $\mathbf{z}_i$ and nothing else about the signer's secrets. In addition, each signer proves with $\pi_{i,2}$ that it evaluated the response honestly. This prevents a malicious signer from returning a ciphertext that fails for only one candidate message and learning the message from whether the user aborts [GH26].

**Blindness.** We prove blindness against fully malicious, colluding signers (Theorem 8). By zero knowledge of $\pi_1$, hiding of the commitment and IND-CPA security of the encryption scheme, the user's protocol messages are independent of its message. By knowledge soundness of $\pi_{i,2}$, each signer's partial signature is the honest function of extracted secrets, so the user's abort decision leaks nothing. Finally, a signature depends on its session only through $\mathbf{r}_{agg}+\mathbf{r}'$, where $\|\mathbf{r}_{agg}\| \le n(1+\ell)\beta_{\mathsf{w}}$. Choosing $\sigma_{\mathsf{w}'}$ sufficiently larger than this bound makes the signatures statistically independent of their sessions.

**One-More Unforgeability (OMUF).** Following the blueprint, we would like to reduce the one-more unforgeability of BMRaccoon to the strong unforgeability of MRaccoon, translating each blind session into exactly one MRaccoon signing query. However, blinding prevents a direct reduction. Each blind session introduces an additional term $\mathbf{A}\mathbf{r}'$ in the aggregated nonce and $\mathbf{r}'_{[m-k]}$ in the response, which are absent in MRaccoon. Furthermore, the honest signers' coefficients are derived from $\mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \cdot)$, where $W$ excludes the blinding shift, whereas MRaccoon hashes every nonce entering the aggregate. Thus, the MRaccoon signing oracle cannot be used directly to simulate blind signing sessions.

*The intermediate scheme* MRaccoon$^\star$. We bridge this gap by introducing an intermediate multi-signature scheme MRaccoon$^\star$ whose signing oracle already includes the shift. We model the user's blinding as an auxiliary signer $\mathcal{S}_0$ with key pair $(\mathbf{0}, \mathbf{0})$, nonce $\mathbf{w}_0 = \mathbf{A}\mathbf{r}'$ and partial signature $\mathbf{z}_0 = \mathbf{r}'_{[m-k]}$. As in BMRaccoon , $\mathbf{w}_0$ is added to the aggregated nonce but not hashed. It is instead bound to the session by the round order, since each signer receives $\mathbf{w}_0$ before sampling its own nonces, just as the user in BMRaccoon commits before seeing any nonce. An MRaccoon$^\star$ signature on $K$ is then an MRaccoon signature on the extended key list $(\mathbf{0}, K)$ under the enlarged bound $\beta'_{\mathsf{z}}$. We prove OMUF via the following chain of reductions

$$\mathsf{AOM\text{-}MISIS} + \mathsf{MSIS} \xrightarrow{\text{Thms. 2, 3}} \mathsf{SUF}(\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]) \xrightarrow{\text{Thm. 5}} \mathsf{SUF}(\mathsf{MRaccoon}^\star) \xrightarrow{\text{Thm. 7}} \mathsf{OMUF}(\mathsf{BMRaccoon}),$$

*From* MRaccoon$^\star$ *to* MRaccoon. We reduce the strong unforgeability of MRaccoon$^\star$ tightly to that of MRaccoon with bound $\beta'_{\mathsf{z}}$, without programming any random oracle (Theorem 5). The reduction registers $(\mathbf{0}, \mathbf{0})$ as an additional corrupted key. This is a valid key pair in the KOSK model, since $\lfloor \mathbf{A} \cdot \mathbf{0} \rceil_{\nu_{\mathsf{t}}} = \mathbf{0}$. Because $\mathcal{S}_0$ holds no secret, the adversary runs it, and the reduction only has to make $\mathbf{w}_0$ visible to the MRaccoon game. To do so, it answers each $\mathsf{H}_{\mathsf{non}}$ query by inserting the auxiliary nonce list $(\mathbf{w}_0, \mathbf{0}, \ldots, \mathbf{0})$ of the corresponding session into $W$ and forwarding the result. This mapping is injective, so the adversary still sees a uniformly random function. Each query is associated with its session via the honest signer's commitment. As honest commitments have high min-entropy, the reduction fails only if the adversary guesses a commitment before it is issued, which occurs with negligible probability.

*From* BMRaccoon *to* MRaccoon$^\star$. This step is a straight-line reduction that neither rewinds the adversary nor programs $\mathsf{H}_{\mathsf{non}}$ or $\mathsf{H}_{\mathsf{sig}}$ (Theorem 7) as RO. The reduction generates the commitment key with a trapdoor, extracts $(\mathbf{r}', \mathbf{m})$ from each commitment $C$, and passes $\mathbf{w}_0 = \mathbf{A}\mathbf{r}'$ to its MRaccoon$^\star$ signing oracle. Soundness of $\pi_1$ ensures that the ciphertexts encrypt exactly the values $(c, \mathbf{b})$ determined by the extracted $(\mathbf{r}', \mathbf{m})$. Hence the honest signer's response is precisely the MRaccoon$^\star$ partial

signature on $((\mathbf{0}, K), \mathbf{m})$. The reduction obtains this partial signature from its $\mathsf{MRaccoon}^\star$ signing oracle and returns a fresh encryption of it, which is indistinguishable from an honest evaluation by circuit privacy, and it simulates $\pi_{1,2}$ using zero knowledge. Each completed blind session therefore corresponds to exactly one $\mathsf{MRaccoon}^\star$ signing query. Suppose the adversary completes $q_c$ sessions and outputs $f > q_c$ valid forgeries. Then, by the pigeonhole principle, some pair $((\mathbf{0}, K), \mathbf{m})$ receives more valid signatures than signing queries, which is a strong forgery against $\mathsf{MRaccoon}^\star$.

Since $\pi_1$ attests to evaluations of $\mathsf{H}_{\mathsf{non}}$ and $\mathsf{H}_{\mathsf{sig}}$, these functions cannot be modelled as random oracles inside the protocol and must be instantiated with a concrete hash function. Accordingly, Theorem 7 assumes that $\mathsf{MRaccoon}^\star$ is strongly unforgeable for this concrete instantiation. Our random-oracle analysis of $\mathsf{MRaccoon}$ and $\mathsf{MRaccoon}^\star$ serves as heuristic evidence for this assumption.

**Future Direction.** Our results hold in the KOSK model, which in practice requires each signer to register its key with a proof of possession. Moreover, verification takes the full list of signer keys as input. A natural direction is therefore to extend $\mathsf{BMRaccoon}$ to the plain public-key (PPK) model with compact key aggregation, in which the verifier uses a single aggregated key computed with hash-derived coefficients, as in MuSig-L [BTT22]. This would first require a PPK variant of $\mathsf{MRaccoon}$, since our proof of OMUF reduces to the security of the underlying multi-signature. Our reduction from $\mathsf{MRaccoon}$ to AOM-MISIS, however, does not carry over to the PPK setting. Without the corrupted signers' secret keys, the reduction can no longer subtract their contribution from the forgeries, and must additionally fork on the key-aggregation hash. This results in up to four related executions. The AOM-MISIS solution then requires a short vector $\mathbf{u}$ that is orthogonal to the signing queries of all these executions. Our construction of $\mathbf{u}$ handles at most two distinct queries per session, using the short inverse of a difference of signed monomials (Lemma 1). With up to four distinct queries per session, it no longer yields a short solution. Obtaining a security proof in the PPK model therefore requires new techniques, which we leave for future work.

## 3 Preliminaries

### 3.1 Notation

**General Notations.** We write $\lambda$ for the security parameter, $\mathsf{negl}$ for a negligible function and PPT for probabilistic polynomial time. For integers $a \le b$ we set $[a, b] := \{a, \ldots, b\}$ and $[n] := [1, n]$. For a vector $\mathbf{x} = (x_1, \ldots, x_\ell)$ we write $\mathbf{x}_{[i]} := (x_1, \ldots, x_i)$ and $\mathbf{x}_{[i,j]} := (x_i, \ldots, x_j)$. We write $x \xleftarrow{\$} S$ for uniform sampling from a finite set $S$, and $x \xleftarrow{\$} \mathcal{D}$ for sampling from a distribution $\mathcal{D}$ with support $\mathsf{Supp}(\mathcal{D})$. The statistical distance of two discrete distributions is $\Delta(P, Q) := \frac{1}{2}\sum_x |P(x) - Q(x)|$. For a scheme $\mathsf{Scheme}$, $\mathsf{Scheme.A}$ denotes its algorithm $\mathsf{A}$.

**Rounds vs. Moves.** We use the following terminology for interactive signing. In a multi-signature scheme, a *round* is a single phase of communication in which each signer exchanges messages with all other signers. In a blind multi-signature scheme, where a user interacts with signers, a *round* typically consists of two *moves*: the user's request to the signer and the signer's response. The last round, however, may consist of the signer's response alone.

**Rings and norms.** Let $\mathcal{R} := \mathbb{Z}[X]/(X^N + 1)$ with $N$ a power of two, and let $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$ for a prime $q$. Vectors are in bold lower case and matrices are in bold upper case. The elements of $\mathcal{R}_q$ are identified with their representatives with coefficients in $(-q/2, q/2]$. The norms of ring elements and vectors are those of the coefficient vector: for $\mathbf{z} = (z_1, \ldots, z_\ell)$ with $z_i = \sum_{j=0}^{N-1} a_{i,j} X^j$,

$$\|\mathbf{z}\|_\infty := \max_{i,j} |a_{i,j}|, \qquad \|\mathbf{z}\|_1 := \sum_{i,j} |a_{i,j}|, \qquad \|\mathbf{z}\|_2 := \Big(\sum_{i,j} a_{i,j}^2\Big)^{1/2},$$

and $\|\cdot\| := \|\cdot\|_2$.

We use two sets of challenge polynomials. The first is the set of sparse ternary polynomials of weight $\beta_{\mathsf{c}}$, defined as

$$\mathcal{S}_{\mathsf{c}} := \{c \in \mathcal{R}_q : \|c\|_\infty = 1,\ \|c\|_1 = \beta_{\mathsf{c}}\}.$$

Each polynomial has exactly $\beta_{\mathsf{c}}$ nonzero coefficients, each in $\{-1, 1\}$. We choose the parameters such that

$$|\mathcal{S}_{\mathsf{c}}| = 2^{\beta_{\mathsf{c}}} \binom{N}{\beta_{\mathsf{c}}} \geq 2^{2\lambda}.$$

The second is the set of signed monomials

$$\mathcal{S}_{\mathsf{b}} := \{\pm 1, \pm X, \ldots, \pm X^{N-1}\} \subseteq \mathcal{R}_q,$$

with $|\mathcal{S}_{\mathsf{b}}| = 2N$.

An important property of signed monomials is that the difference of any two distinct elements admits a short inverse up to a factor of 2, as stated in the following lemma.

**Lemma 1 ( [BCK+14, Lemma 3.1]).** *For any distinct $b, b' \in \mathcal{S}_{\mathsf{b}}$, there exists $\gamma \in \mathcal{R}$ such that*

$$\|\gamma\|_\infty \leq 1 \qquad \textit{and} \qquad (b - b')\gamma = 2$$

*in $\mathcal{R}$, and hence also modulo $q$.*

**Modulus Rounding** Let $\nu \geq 0$, $q > 2^\nu$, and $q_\nu = \lfloor q/2^\nu \rfloor$. For $x \in \mathbb{Z}_q$, let $\bar{x} \in \{0, \ldots, q-1\}$ be its canonical representative; likewise, for $u \in \mathbb{Z}_{q_\nu}$, let $\bar{u} \in \{0, \ldots, q_\nu - 1\}$ denote its canonical lift. Following [HMM+26, Section 2.1], define modulus rounding by

$$\lfloor x \rceil_\nu := \left\lfloor \frac{\bar{x}}{2^\nu} + \frac{1}{2} \right\rfloor \bmod q_\nu \in \mathbb{Z}_{q_\nu}.$$

The operation extends coefficient-wise to ring elements and vectors.

**Lemma 2 ([HMM+26, Lemma 11]).** *Let $\nu \in \mathbb{Z}_{\geq 0}$ and $q \in \mathbb{Z}_{>0}$ with $q > 2$, and set $q_\nu = \lfloor q/2^\nu \rfloor$. Assume that $q_\nu = \lfloor q/2^\nu \rceil$, that is,*

$$q = 2^\nu q_\nu + q_\perp, \qquad q_\perp \in \left[0,\, 2^{\max(\nu-1,0)} - 1\right].$$

*Then, for any $x \in \mathbb{Z}_q$, $\left| x - 2^\nu \cdot \overline{\lfloor x \rceil_\nu} \bmod q \right| \leq 2^\nu - 1$. Moreover, for all $\mathbf{x}, \boldsymbol{\delta} \in \mathbb{Z}_q^n$,*

$$\left\| 2^\nu \cdot \left( \overline{\lfloor \mathbf{x} + \boldsymbol{\delta} \rceil_\nu} - \overline{\lfloor \mathbf{x} \rceil_\nu} \right) \bmod q \right\| \leq \left\| 2^\nu \cdot \overline{\lfloor \boldsymbol{\delta} \rceil_\nu} \bmod q \right\| + \|\mathbf{1}\| \cdot f(\nu),$$

*where $f(0) = 0$, $f(1) = 3$, $f(2) = 6$, $f(3) = 12$, and $f(\nu) = 2^\nu$ for $\nu \geq 4$.*

Throughout, $q_\nu$ and $q_\perp$ are as in Lemma 2.

**Lemma 3 (Rounding with carry).** *Let $\nu \geq 1$. For every $x \in \mathbb{Z}_q$ there are unique $d \in \{0, 1\}$ and $e \in [-2^{\nu-1}, 2^{\nu-1} - 1]$ such that*

$$\overline{x} = 2^\nu \big( \overline{\lfloor x \rceil_\nu} + q_\nu d \big) + e \quad \textit{over } \mathbb{Z}, \qquad \textit{and hence} \qquad 2^\nu \, \overline{\lfloor x \rceil_\nu} \equiv x - e + q_\perp d \pmod{q}.$$

*Proof.* Let $u := \lfloor \overline{x}/2^\nu + 1/2 \rfloor$, so that $e := \overline{x} - 2^\nu u \in [-2^{\nu-1}, 2^{\nu-1} - 1]$, and $u$ is the only integer with this property since the interval has length $2^\nu$. As $\overline{x} \leq 2^\nu q_\nu + q_\perp - 1$ and $q_\perp < 2^{\nu-1}$, we have $0 \leq u \leq q_\nu$. Since $\overline{\lfloor x \rceil_\nu} = u \bmod q_\nu \in [0, q_\nu - 1]$, it follows that $u = \overline{\lfloor x \rceil_\nu} + q_\nu d$ for a unique $d \in \{0, 1\}$. The congruence follows from $2^\nu q_\nu = q - q_\perp$.

**Corollary 1.** *Let $\mathbf{u}_1, \ldots, \mathbf{u}_\ell \in \mathcal{R}_{q_\nu}^k$ and $a_1, \ldots, a_\ell \in \mathbb{Z}$. If $\sum_{i=1}^\ell a_i \, \overline{\mathbf{u}_i} = q_\nu \boldsymbol{\kappa}$ for some integer vector $\boldsymbol{\kappa}$, then $\sum_{i=1}^\ell a_i \, 2^\nu \, \overline{\mathbf{u}_i} \equiv -q_\perp \boldsymbol{\kappa} \pmod{q}$.*

*Proof.* Multiply by $2^\nu$ and use $2^\nu q_\nu = q - q_\perp \equiv -q_\perp \pmod{q}$. □

### 3.2 Module Lattices and Discrete Gaussian Distributions

**Module Lattices.** Let $\mathcal{R}$ be a ring and let $\mathbf{A} \in \mathcal{R}^{k \times \ell}$. The module lattice generated by $\mathbf{A}$ is defined as

$$\mathcal{L}(\mathbf{A}) := \{\mathbf{A}\mathbf{z} \mid \mathbf{z} \in \mathcal{R}^\ell\} \subseteq \mathcal{R}^k.$$

The corresponding kernel module lattice is defined as

$$\mathcal{L}^\perp(\mathbf{A}) := \{\mathbf{z} \in \mathcal{R}^\ell \mid \mathbf{A}\mathbf{z} = \mathbf{0}\} \subseteq \mathcal{R}^\ell.$$

**Discrete Gaussians.** For a parameter $\sigma > 0$ and a center $\mathbf{c} \in \mathcal{R}^m$, define the Gaussian function as $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) := \exp(-\pi\|\mathbf{x}-\mathbf{c}\|^2/\sigma^2)$ and, for a countable set $S$, let $\rho_{\sigma,\mathbf{c}}(S) := \sum_{\mathbf{y}\in S} \rho_{\sigma,\mathbf{c}}(\mathbf{y})$. Note that $\sigma$ is the Gaussian width and the corresponding standard deviation is approximately $\sigma/\sqrt{2\pi}$. The discrete Gaussian distribution over a coset $\mathcal{L}+\mathbf{a}$ assigns each $\mathbf{x} \in \mathcal{L}+\mathbf{a}$ the probability

$$\mathcal{D}_{\mathcal{L}+\mathbf{a},\sigma,\mathbf{c}}(\mathbf{x}) := \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{x})}{\rho_{\sigma,\mathbf{c}}(\mathcal{L}+\mathbf{a})}.$$

We omit $\mathbf{c}$ when $\mathbf{c} = \mathbf{0}$ and omit the coset when $\mathcal{L}+\mathbf{a} = \mathcal{R}^m$, writing $\mathcal{D}^m_\sigma$. Reducing a sample modulo $q$ induces a distribution over $\mathcal{R}^m_q$, which we denote the same way.

**Definition 1 (Smoothing Parameter [MR07]).** *Let $\mathcal{L} \subseteq \mathcal{R}^m$ be a full-rank lattice with dual $\mathcal{L}^*$, and let $\varepsilon > 0$. The smoothing parameter $\eta_\varepsilon(\mathcal{L})$ is the smallest $\sigma > 0$ such that $\sum_{\mathbf{y}\in\mathcal{L}^*\setminus\{\mathbf{0}\}} \exp(-\pi\sigma^2\|\mathbf{y}\|^2) \le \varepsilon$.*

**Lemma 4 ( [MR07], Lemmas 3.2 and 3.3).** *For any $n$-dimensional lattice $\mathcal{L}$,*

$$\eta_{1/2^n}(\mathcal{L}) \le \min\left\{\frac{\sqrt{n}}{\lambda_1(\mathcal{L}^*)}, \sqrt{\frac{\ln(2n(1+2^n))}{\pi}}\,\lambda_n(\mathcal{L})\right\}.$$

The following lemma bounds the tail probability of a one-dimensional discrete Gaussian whose width $\sigma$ is at least the smoothing parameter.

**Lemma 5 ( [MR07], Lemma 4.4).** *Let $x \leftarrow \mathcal{D}_{\mathbb{Z}^d,\sigma}$. For any $\varepsilon \in (0, 1/2]$, $d \ge 1$, if $\sigma \ge \eta_\varepsilon(\mathbb{Z}^d)$, then, ,*

$$\Pr\left[\|x\| \ge \sigma\sqrt{d}\right] \le 2^{-d}\frac{1+\varepsilon}{1-\varepsilon}.$$

We also use the following tail bound for the Euclidean norm of a continuous Gaussian vector.

**Lemma 6 ( [DPKN+25], Lemma 2.1).** *Let $\sigma \ge \eta_\varepsilon(\mathbb{Z}^{Nm})$, where $\varepsilon \le 1/2$ and let $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^{Nm},\sigma}$ where $Nm \ge 2$. Then*

$$\Pr\left[\|\mathbf{x}\|_2 \ge e^{1/4}\sigma\sqrt{Nm}\right] \le 2^{-Nm/10}.$$

**Rényi Divergence.** Rényi divergence quantifies the difference between two probability distributions and is used in lattice-based cryptography to analyze the statistical closeness of discrete Gaussians.

**Definition 2 (Rényi Divergence).** *Let $P, Q$ be discrete probability distributions with $\mathsf{Supp}(P) \subseteq \mathsf{Supp}(Q)$. For $\alpha \in (1, \infty)$, the Rényi divergence of order $\alpha$ is*

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

For $\alpha = 2$, the quantity $R_2(P \,\|\, Q) - 1$ is the $\chi^2$-divergence of $P$ from $Q$, which bounds the statistical distance $\Delta(P, Q)$ as follows.

**Lemma 7.** *Let $P, Q$ be discrete probability distributions with $\mathsf{Supp}(P) \subseteq \mathsf{Supp}(Q)$. Then*

$$\Delta(P, Q) \le \tfrac{1}{2}\sqrt{R_2(P \,\|\, Q) - 1}.$$

The following lemma bounds the Rényi divergence between two discrete Gaussians with different centers.

**Lemma 8 ( [BLRL+18, Lemma 4.2]).** *Let $\mathcal{L} \subseteq \mathcal{R}^m$ be a module lattice, let $\varepsilon \in (0, 1)$, and let $\sigma \ge \eta_\varepsilon(\mathcal{L})$. Then, for every $\mathbf{c} \in \mathcal{L}$,*

$$R_2\big(\mathbf{c} + \mathcal{D}_{\mathcal{L},\sigma} \,\big\|\, \mathcal{D}_{\mathcal{L},\sigma}\big) \le \Big(\frac{1+\varepsilon}{1-\varepsilon}\Big)^2 \exp\Big(\frac{2\pi\|\mathbf{c}\|^2}{\sigma^2}\Big).$$

Combining Lemmas 7 and 8 bounds the statistical distance between a discrete Gaussian and its shift.

**Corollary 2.** *Let $\mathcal{L} \subseteq \mathcal{R}^m$ be a module lattice, let $\varepsilon \le 1/4$, and let $\sigma \ge \eta_\varepsilon(\mathcal{L})$. Then, for every $\mathbf{c} \in \mathcal{L}$ with $2\pi\|\mathbf{c}\|^2 \le \sigma^2$,*

$$\Delta\big(\mathbf{c} + \mathcal{D}_{\mathcal{L},\sigma},\, \mathcal{D}_{\mathcal{L},\sigma}\big) \le \sqrt{\pi}\,\frac{\|\mathbf{c}\|}{\sigma} + \sqrt{6\varepsilon}.$$

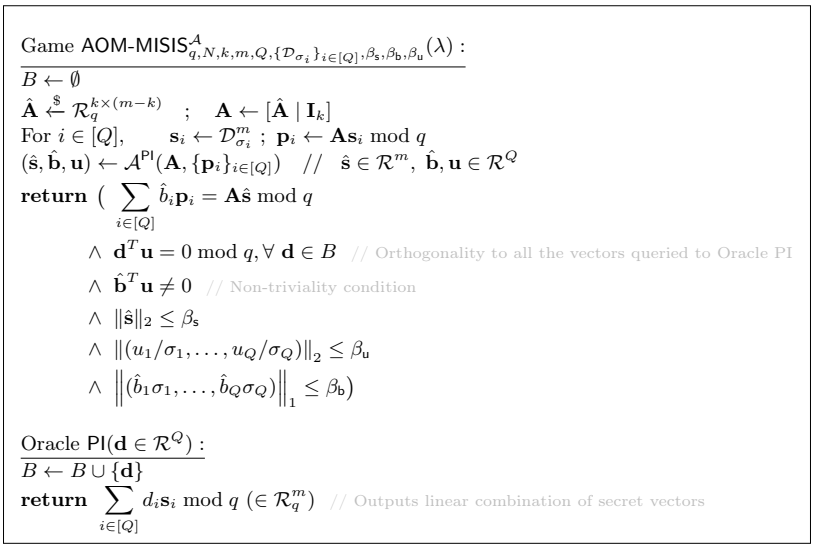



**Fig. 1.** he AOM-MISIS game with parameters $par = (q, N, k, m, Q, (\sigma_i)_{i\in[Q]}, \beta_{\mathsf{s}}, \beta_{\mathsf{b}}, \beta_{\mathsf{u}})$.

**Hardness Assumptions.** We recall the module short integer solution (MSIS) and module learning with errors (MLWE) assumptions, which are standard lattice hardness assumptions, as well as the algebraic one-more module inhomogeneous SIS (AOM-MISIS) assumption. The latter is recently introduced in [ZT25] for which reductions to MSIS and MLWE have been established.

**Definition 3 (MSIS [LS15]).** *Let $par = (q, N, k, m, \beta)$, where $q$ is a prime, $N$ is a power of two, $k, m$ are positive integers, and $\beta > 0$, all functions of $\lambda$. The advantage of an adversary $\mathcal{A}$ is*

$$\mathsf{Adv}^{\mathsf{MSIS}}_{\mathsf{par}}(\mathbf{A}, \lambda) := \Pr\left[0 < \|\mathbf{s}\| \leq \beta \ \wedge\ [\hat{\mathbf{A}} \mid \mathbf{I}_k]\,\mathbf{s} = \mathbf{0} \bmod q : \hat{\mathbf{A}} \xleftarrow{\$} \mathcal{R}_q^{k\times(m-k)},\ \mathbf{s} \in \mathcal{R}_q^m \leftarrow \mathcal{A}(\hat{\mathbf{A}})\right].$$

*The $\mathsf{MSIS}_{\mathsf{par}}$ assumption holds if this advantage is negligible for every PPT adversary $\mathcal{A}$.*

**Definition 4 (MLWE [LS15]).** *Let $par = (q, N, k, m, \mathcal{D})$, where $q$ is a prime, $N$ is a power of two, $k, m$ are positive integers, and $\mathcal{D}$ is a distribution over $\mathcal{R}$, all functions of $\lambda$. The advantage of an adversary $\mathcal{A}$ is*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{MLWE}}_{\mathsf{par}}(\mathcal{A}, \lambda) := \Big|&\Pr\left[\mathcal{A}(\hat{\mathbf{A}}, \mathbf{t}) = 1 : \hat{\mathbf{A}} \xleftarrow{\$} \mathcal{R}_q^{k\times(m-k)},\ \mathbf{s} \xleftarrow{\$} \mathcal{D}^m,\ \mathbf{t} := [\hat{\mathbf{A}} \mid \mathbf{I}_k]\,\mathbf{s} \bmod q\right]\\ &- \Pr\left[\mathcal{A}(\hat{\mathbf{A}}, \mathbf{t}) = 1 : \hat{\mathbf{A}} \xleftarrow{\$} \mathcal{R}_q^{k\times(m-k)},\ \mathbf{t} \xleftarrow{\$} \mathcal{R}_q^k\right]\Big|.\end{aligned}$$

*The $\mathsf{MLWE}_{\mathsf{par}}$ assumption holds if this advantage is negligible for every PPT adversary $\mathcal{A}$.*

**Definition 5 (AOM-MISIS [ZT25]).** *Let $par = \big(q, N, k, m, Q, (\mathcal{D}_{\sigma_i})_{i\in[Q]}, \beta_{\mathsf{s}}, \beta_{\mathsf{b}}, \beta_{\mathsf{u}}\big)$, where $q$ is an odd prime, $N$ is a power of two, $k, m, Q$ are positive integers, $\{\mathcal{D}_{\sigma_i}\}_{i\in Q}$ is a distribution over $\mathcal{R}$ with gaussian widths $\{\sigma_i\}_{i\in Q}$ respectively and $\beta_{\mathsf{s}}, \beta_{\mathsf{b}}, \beta_{\mathsf{u}} > 0$, all functions of $\lambda$. For the game $\mathsf{AOM\text{-}MISIS}^{\mathcal{A}}_{\mathsf{par}}(\lambda)$ in Figure 1, the advantage of an adversary $\mathcal{A}$ is*

$$\mathsf{Adv}^{\mathsf{AOM\text{-}MISIS}}_{\mathsf{par}}(\mathcal{A}, \lambda) := \Pr\left[\mathsf{AOM\text{-}MISIS}^{\mathcal{A}}_{\mathsf{par}}(\lambda) = 1\right].$$

*The $\mathsf{AOM\text{-}MISIS}_{\mathsf{par}}$ assumption holds if this advantage is negligible for every PPT adversary $\mathcal{A}$.*

[ZT25] showed that the hardness of AOM-MISIS can be reduced to MSIS and MLWE for suitable parameter choices.

### 3.3 Non-Interactive Zero-Knowledge Arguments

Let $\mathsf{R}$ be an NP relation. It defines the language $\mathsf{L} = \{\,\mathsf{stmt} \mid \exists\,\mathsf{wit}$ such that $(\mathsf{stmt}, \mathsf{wit}) \in \mathsf{R}\,\}$, where $\mathsf{stmt}$ is the *statement* and $\mathsf{wit}$ is the *witness*. A non-interactive zero-knowledge argument $\mathsf{NIZK}_{\mathsf{L}}$ for $\mathsf{R}$ is a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$.[4]

[4] We consider only transparent arguments, which do not require a trusted setup, as in [BFM+26].

- $\mathsf{NIZK_L.Setup}(1^\lambda) \to \mathsf{pp}$: Given the security parameter $\lambda$, outputs public parameters $\mathsf{pp}$.
- $\mathsf{NIZK_L.Prove}(\mathsf{pp}, \mathsf{stmt}, \mathsf{wit}) \to \pi$: Given public parameters $\mathsf{pp}$, a statement $\mathsf{stmt}$ and a witness $\mathsf{wit}$ such that $(\mathsf{stmt}, \mathsf{wit}) \in \mathsf{R}$, outputs a proof $\pi$.
- $\mathsf{NIZK_L.Verify}(\mathsf{pp}, \pi, \mathsf{stmt}) \to \{0,1\}$: Given public parameters $\mathsf{pp}$, a proof $\pi$ and a statement $\mathsf{stmt}$, outputs 1 (accept) or 0 (reject).

The argument satisfies the following properties.

- **Completeness.** An honestly generated proof for a true statement is accepted, except with negligible probability. Formally, for all $(\mathsf{stmt}, \mathsf{wit}) \in \mathsf{R}$,

$$\Pr\left[\mathsf{NIZK_L.Verify}(\mathsf{pp}, \pi, \mathsf{stmt}) = 1 : \pi \leftarrow \mathsf{NIZK_L.Prove}(\mathsf{pp}, \mathsf{stmt}, \mathsf{wit})\right] \geq 1 - \mathsf{negl}(\lambda).$$

- **Zero-knowledge.** A proof reveals nothing about the witness beyond the validity of the statement: it can be simulated without the witness. Formally, there exists a PPT simulator $\mathsf{NIZK_L.Sim}$, which may program the random oracle, such that for all $(\mathsf{stmt}, \mathsf{wit}) \in \mathsf{R}$ and every PPT distinguisher $\mathcal{A}$, the advantage

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{NIZK_L},\mathcal{A}}(\lambda) := \Big| &\Pr\left[\mathcal{A}(\mathsf{pp}, \pi) = 1 : \pi \leftarrow \mathsf{NIZK_L.Prove}(\mathsf{pp}, \mathsf{stmt}, \mathsf{wit})\right] \\ &- \Pr\left[\mathcal{A}(\mathsf{pp}, \pi) = 1 : \pi \leftarrow \mathsf{NIZK_L.Sim}(\mathsf{pp}, \mathsf{stmt})\right]\Big| \leq \mathsf{negl}(\lambda)\end{aligned}$$

- **Soundness.** No false statement has an accepting proof. For every PPT $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{SND}}_{\mathsf{NIZK_L},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{NIZK_L.Verify}(\pi, \mathsf{stmt}) = 1 \ \wedge \ \mathsf{stmt} \notin \mathsf{L} : (\mathsf{stmt}, \pi) \leftarrow \mathcal{A}(\mathsf{pp})\right] \leq \mathsf{negl}(\lambda)$$

- **Knowledge soundness.** A prover that outputs an accepting proof must know a valid witness, which can be extracted from it. Formally, there exists a PPT extractor $\mathsf{NIZK_L.Ext}$, with rewinding access to the prover, such that for every PPT prover $\mathsf{Prove}^*$, the advantage

$$\mathsf{Adv}^{\mathsf{KS}}_{\mathsf{NIZK_L},\mathsf{Prove}^*}(\lambda) := \Pr\left[\begin{array}{l}\mathsf{NIZK_L.Verify}(\mathsf{pp}, \pi, \mathsf{stmt}) = 1 \\ \wedge\ (\mathsf{stmt}, \mathsf{wit}) \notin \mathsf{R}\end{array} : \begin{array}{l}(\mathsf{stmt}, \pi) \leftarrow \mathsf{Prove}^*(\mathsf{pp}), \\ \mathsf{wit} \leftarrow \mathsf{NIZK_L.Ext}^{\mathsf{Prove}^*}(\mathsf{pp}, \mathsf{stmt}, \pi)\end{array}\right] \leq \mathsf{negl}(\lambda)$$

### 3.4 Extractable Commitment Schemes

An extractable commitment scheme $\mathsf{CS}$ with message space $\mathcal{M}$ is a tuple of PPT algorithms ($\mathsf{Setup}$, $\mathsf{SetupExt}, \mathsf{Com}, \mathsf{Ver}$).

- $\mathsf{CS.Setup}(1^\lambda) \to ck$: Given the security parameter $\lambda$, outputs a commitment key $ck$, which defines the message space $\mathcal{M}$.
- $\mathsf{CS.Com}(ck, \mathsf{msg}; \theta) \to (C, \mathsf{aux})$: Given a commitment key $ck$ , a message $\mathsf{msg} \in \mathcal{M}$, and a random coin $\theta$ outputs a commitment $C$ and opening information $\mathsf{aux}$.
- $\mathsf{CS.Ver}(ck, \mathsf{msg}, C, \mathsf{aux}) \to \{0,1\}$: Given a commitment key $ck$, a message $\mathsf{msg}$, a commitment $C$ and opening information $\mathsf{aux}$, outputs 1 if $(C, \mathsf{aux})$ is a valid opening to $\mathsf{msg}$, and 0 otherwise.

We omit $\mathsf{ck}$ and the coins $\theta$ when they are clear from context. The scheme satisfies the following properties.

- **Correctness.** An honestly generated commitment verifies against the message it was computed on. Formally, for all $\mathsf{msg} \in \mathcal{M}$,

$$\Pr\left[\mathsf{CS.Ver}(ck, \mathsf{msg}, C, \mathsf{aux}) = 1 : ck \leftarrow \mathsf{CS.Setup}(1^\lambda),\ (C, \mathsf{aux}) \leftarrow \mathsf{CS.Com}(ck, \mathsf{msg})\right] = 1.$$

- **Hiding.** A commitment reveals nothing about the committed message. Formally, for every PPT $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the advantage

$$\mathsf{Adv}^{\mathsf{HID}}_{\mathsf{CS},\mathcal{A}}(\lambda) := \left|\Pr\left[b' = b : \begin{array}{l}ck \leftarrow \mathsf{CS.Setup}(1^\lambda),\ (\mathsf{msg}_0, \mathsf{msg}_1, \mathsf{st}) \leftarrow \mathcal{A}_1(ck), \\ b \leftarrow \{0,1\},\ (C, \mathsf{aux}) \leftarrow \mathsf{CS.Com}(ck, \mathsf{msg}_b),\ b' \leftarrow \mathcal{A}_2(\mathsf{st}, C)\end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

– **Binding.** A commitment cannot be validly opened to two distinct messages. Formally, for every PPT $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{BND}}_{\mathsf{CS},\mathcal{A}}(\lambda) := \Pr\left[\begin{matrix}\mathsf{msg}_0 \neq \mathsf{msg}_1 \wedge \\ \forall b \in \{0,1\} : \mathsf{CS.Ver}(ck, \mathsf{msg}_b, C, \mathsf{aux}_b) = 1\end{matrix} : \begin{matrix}ck \leftarrow \mathsf{CS.Setup}(1^\lambda), \\ (C, \mathsf{msg}_0, \mathsf{aux}_0, \mathsf{msg}_1, \mathsf{aux}_1) \leftarrow \mathcal{A}(ck)\end{matrix}\right]$$

is negligible.

– **Setup indistinguishability.** An extractable commitment scheme extends a standard commitment scheme with an alternative setup algorithm $\mathsf{CS.SetupExt}(1^\lambda)$, which outputs a commitment key $ck$ and a trapdoor $\tau$ which enables extraction. Formally, for every PPT $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{SetupInd}}_{\mathsf{CS},\mathcal{A}}(\lambda) := \left|\Pr\left[b' = b : \begin{matrix}\mathsf{ck}_0 \leftarrow \mathsf{CS.Setup}(1^\lambda),\ (\mathsf{ck}_1, \tau) \leftarrow \mathsf{CS.SetupExt}(1^\lambda), \\ b \leftarrow \{0,1\},\ b' \leftarrow \mathcal{A}(\mathsf{ck}_b)\end{matrix}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

– **Extractability.** Under a trapdoor key, the committed message can be recovered from the commitment alone. Formally, there exists a deterministic PPT extractor $\mathsf{CS.Ext}$ such that for every PPT $\mathcal{A}$, the advantage

$$\mathsf{Adv}^{\mathsf{Ext}}_{\mathsf{CS},\mathcal{A}}(\lambda) := \Pr\left[\begin{matrix}\mathsf{msg} \in \mathcal{M}\ \wedge\ \mathsf{CS.Ver}(ck, \mathsf{msg}, C, \mathsf{aux}) = 1 \\ \wedge\ \mathsf{msg} \neq \mathsf{CS.Ext}(\tau, C)\end{matrix} : \begin{matrix}(ck, \tau) \leftarrow \mathsf{CS.SetupExt}(1^\lambda), \\ (C, \mathsf{msg}, \mathsf{aux}) \leftarrow \mathcal{A}(ck)\end{matrix}\right] \leq \mathsf{negl}(\lambda)$$

### 3.5 Homomorphic Encryption

A homomorphic encryption scheme $\mathsf{HE}$ with message space $\mathcal{M}$, coin space $\Theta$ and admissible function class $\mathcal{F}$ is a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$.

– $\mathsf{HE.Setup}(1^\lambda) \to \mathsf{pp}$: Given the security parameter $\lambda$, outputs public parameters $\mathsf{pp}$, which include descriptions of $\mathcal{M}$, $\Theta$ and $\mathcal{F}$, an encryption-coin distribution $\mathcal{D}$ over $\Theta$, and a flooding distribution $\mathcal{D}_{\mathsf{fl}}$ over $\Theta$.
– $\mathsf{HE.KGen}(\mathsf{pp}) \to (ek, dk)$: Given public parameters $\mathsf{pp}$, outputs an encryption key $ek$ and a decryption key $dk$.
– $\mathsf{HE.Enc}(\mathsf{pp}, ek, \mathsf{msg}; \theta) \to \mathsf{ctx}$: Given public parameters $\mathsf{pp}$, an encryption key $ek$, a message $\mathsf{msg} \in \mathcal{M}$ and coins $\theta \in \Theta$, deterministically outputs a ciphertext $\mathsf{ctx}$. We write $\mathsf{HE.Enc}(\mathsf{pp}, ek, \mathsf{msg}; \mathcal{D}')$ for encryption with fresh coins $\theta \leftarrow \mathcal{D}'$, and $\mathsf{HE.Enc}(\mathsf{pp}, ek, \mathsf{msg})$ for the default case $\mathcal{D}' = \mathcal{D}$.
– $\mathsf{HE.Eval}(\mathsf{pp}, ek, f, (\mathsf{ctx}_1, \ldots, \mathsf{ctx}_n)) \to \mathsf{ctx}_f$: Given public parameters $\mathsf{pp}$, an encryption key $ek$, a function $f\colon \mathcal{M}^n \to \mathcal{M}$ in $\mathcal{F}$ and ciphertexts $\mathsf{ctx}_1, \ldots, \mathsf{ctx}_n$, deterministically outputs an evaluated ciphertext $\mathsf{ctx}_f$.
– $\mathsf{HE.Dec}(\mathsf{pp}, dk, \mathsf{ctx}) \to \mathsf{msg}/\perp$: Given public parameters $\mathsf{pp}$, a decryption key $dk$ and a ciphertext $\mathsf{ctx}$, outputs a message $\mathsf{msg} \in \mathcal{M}$, or $\perp$ if decryption fails.

*Coins.* Fresh ciphertexts use coins $\theta \leftarrow \mathcal{D}$. Flooded ciphertexts use coins from a distribution $\mathcal{D}_{\mathsf{fl}}$ with much larger noise, written $\mathsf{HE.Enc}(\mathsf{ek}, \mathsf{msg}; \mathcal{D}_{\mathsf{fl}})$. Let $\Theta_{\mathsf{adm}} \subseteq \Theta$ be the set of *admissible coins*, whose noise lies within the bounds the proofs enforce; coins drawn from $\mathcal{D}$ or $\mathcal{D}_{\mathsf{fl}}$ are admissible except with negligible probability.

For convenience, we omit $\mathsf{pp}$ and the coins. This also applies to the languages used by $\mathsf{BMRaccoon}$: their witnesses implicitly contain the coins of every ciphertext and the opening of every commitment they mention, and their relations require the coins to be admissible.

*Evaluated functions.* Our scheme only evaluates linear functions. Vectors are encrypted componentwise. For $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_\ell) \in (\mathcal{R}^{m-k})^{\ell+1}$ with $\max_j \|\boldsymbol{\alpha}_j\|_1 \leq B_\alpha$, let

$$f_{\boldsymbol{\alpha}}(c, b^{(1)}, \ldots, b^{(\ell)}, \mathbf{x}) := c\,\boldsymbol{\alpha}_0 + \sum_{j \in [\ell]} b^{(j)} \boldsymbol{\alpha}_j + \mathbf{x},$$

and let $\mathcal{F} := \{f_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}}$ (so $n = \ell + 2$). Privacy of the evaluation comes from the last input, which is a flooded encryption. We write $\mathsf{HE.Eval}_{\mathsf{priv}}(\mathsf{ek}, f_{\boldsymbol{\alpha}}, (ctx_i)_{i \in [\ell+1]}, \mathbf{x})$ for the resulting randomized procedure, which samples $ctx_{\mathbf{x}} \leftarrow \mathsf{HE.Enc}(\mathsf{ek}, \mathbf{x}; \mathcal{D}_{\mathsf{fl}})$ and outputs $\mathsf{HE.Eval}(\mathsf{ek}, f_{\boldsymbol{\alpha}}, (ctx_i)_{i \in [\ell+1]}, ctx_{\mathbf{x}})$.

The scheme satisfies the following properties.

– **Correctness (with error $\varepsilon_{\mathsf{HE}}$).** Decrypting an honestly generated ciphertext recovers the encrypted message, and decrypting an evaluated ciphertext yields the function applied to the underlying messages, both for default and for flooding coins. Formally, for all $\mathsf{msg} \in \mathcal{M}$ and $\mathcal{D}' \in \{\mathcal{D}, \mathcal{D}_{\mathsf{fl}}\}$,

$$\Pr\left[\mathsf{HE.Dec}(\mathsf{pp}, dk, \mathsf{ctx}) = \mathsf{msg} : (ek, dk) \leftarrow \mathsf{HE.KGen}(\mathsf{pp}),\ \theta \leftarrow \mathcal{D}', \mathsf{ctx} := \mathsf{HE.Enc}(\mathsf{pp}, ek, \mathsf{msg}; \theta)\right] \geq 1 - \varepsilon_{\mathsf{HE}},$$

and for every $f \in \mathcal{F}$ and all $\mathsf{msg}_1, \ldots, \mathsf{msg}_n \in \mathcal{M}$,

$$\Pr\left[\begin{array}{c}\mathsf{HE.Dec}(\mathsf{pp}, dk, \mathsf{ctx}_f) \\ = f(\mathsf{msg}_1, \ldots, \mathsf{msg}_n)\end{array} : \begin{array}{c}(ek, dk) \leftarrow \mathsf{HE.KGen}(\mathsf{pp}),\ \theta_i \leftarrow \mathcal{D}\ (i < n),\ \theta_n \leftarrow \mathcal{D}_{\mathsf{fl}}, \\ \mathsf{ctx}_i := \mathsf{HE.Enc}(\mathsf{pp}, ek, \mathsf{msg}_i; \theta_i)\ \forall i \in [n], \\ \mathsf{ctx}_f := \mathsf{HE.Eval}(\mathsf{pp}, ek, f, (\mathsf{ctx}_1, \ldots, \mathsf{ctx}_n))\end{array}\right] \geq 1 - \varepsilon_{\mathsf{HE}}.$$

The second condition is correctness of $\mathsf{HE.Eval}_{\mathsf{priv}}$. We say $\mathsf{HE}$ is correct if $\varepsilon_{\mathsf{HE}}$ is negligible.

– **IND-CPA security.** A ciphertext reveals nothing about the encrypted message, even to an adversary that knows the encryption key. Formally, for every PPT $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the advantage

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{A}}(\lambda) := \left|\Pr\left[b' = b : \begin{array}{c}(ek, dk) \leftarrow \mathsf{HE.KGen}(\mathsf{pp}),\ (\mathsf{msg}_0, \mathsf{msg}_1, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{pp}, ek), \\ b \leftarrow \{0,1\},\ \theta \leftarrow \mathcal{D},\ \mathsf{ctx} := \mathsf{HE.Enc}(\mathsf{pp}, ek, \mathsf{msg}_b; \theta), \\ b' \leftarrow \mathcal{A}_2(\mathsf{st}, \mathsf{ctx})\end{array}\right] - \frac{1}{2}\right|$$

is negligible.

– **Chosen-key circuit privacy.** Following [HMM+26], private evaluation must hide $f$ and the private input beyond the output, even from the party that chose $ek$ and knows any decryption key and all coins. Let $\mathsf{HE.Eval}_{\mathsf{priv}}(\mathsf{pp}, ek, f, (\mathsf{ctx}_i)_{i\in[n-1]}, \mathsf{msg}_n)$ be as above, and let $\mathsf{HE.Sim}(\mathsf{pp}, ek, y) := \mathsf{HE.Enc}(\mathsf{pp}, ek, y; \mathcal{D}_{\mathsf{fl}})$. An instance $(ek, f, (\mathsf{msg}_i, \mathsf{ctx}_i)_{i\in[n-1]}, \mathsf{msg}_n)$ is *admissible* if $ek$ is syntactically valid (not necessarily honestly generated), $f \in \mathcal{F}$, and each $\mathsf{ctx}_i = \mathsf{HE.Enc}(\mathsf{pp}, ek, \mathsf{msg}_i; \theta_i)$ for some $\theta_i \in \Theta_{\mathsf{adm}}$, where the plaintexts and induced flooding shifts satisfy the prescribed bounds. This guarantee holds even when the $\theta_i$ are unknown. We require that for every admissible instance

$$\Delta\Big(\mathsf{HE.Eval}_{\mathsf{priv}}\big(\mathsf{pp}, ek, f, (\mathsf{ctx}_i)_{i\in[n-1]}, \mathsf{msg}_n\big),\ \mathsf{HE.Sim}\big(\mathsf{pp}, ek, f(\mathsf{msg}_1, \ldots, \mathsf{msg}_n)\big)\Big) \leq \varepsilon_{\mathsf{cp}}(\lambda).$$

Since the bound is pointwise over the fresh flooding coins, it holds for any (even unbounded) distinguisher, given any auxiliary information fixed beforehand.

### 3.6 Multi-Signatures

A multi-signature scheme $\mathsf{MS}$ allows $n$ signers with verification keys $K = \{vk_1, \ldots, vk_n\}$ to jointly produce a single compact signature on a message $m$. In this work we focus on two-round multi-signature[5] schemes, which we formally define as follows.

– $\mathsf{Setup}(1^\lambda) \to par$: on input the security parameter, outputs public parameters $par$.
– $\mathsf{KeyGen}(par) \to (sk_i, vk_i)$: outputs a signing key $sk_i$ and a verification key $vk_i$ for signer $\mathcal{S}_i$.
– $\mathsf{Sign}_1(\,) \to (pst_i, st_i)$: Signer $i$ outputs a public state $pst_i$ and a local state $st_i$.
– $\mathsf{Sign}_2(K, \{pst_{i'}\}_{i'\in[n]}, m, sk_i, st_i) \to \sigma_i$: On input the verification key set $K := \{vk'_i\}_{i'\in[n]}$, public state from all the signers $\{pst_{i'}\}_{i'\in[n]}$, signer $i$ outputs a partial signature $\sigma_i$ on message $m$ using its secret key $sk_i$ and state $st_i$.
– $\mathsf{SignAgg}(K, m, \{\sigma_i\}_{i\in[n]}) \to \sigma$: Aggregates the partial signatures into a single signature $\sigma$.
– $\mathsf{Verify}(K, m, \sigma) \to \{0,1\}$: Outputs 1 if $\sigma$ is a valid signature on $m$ under $K$, and 0 otherwise.

**Correctness.** $\mathsf{MS}$ is correct if, for all $par \leftarrow \mathsf{Setup}(1^\lambda)$, all key sets $K = \{vk_1, \ldots, vk_n\}$ with $(sk_i, vk_i) \leftarrow \mathsf{KeyGen}(par)$, and all messages $m$, an honest execution of $\mathsf{Sign}_1$, $\mathsf{Sign}_2$ and $\mathsf{SignAgg}$ by all signers in $K$ produces a signature $\sigma$ such that

$$\Pr\left[\mathsf{Verify}(K, m, \sigma) = 1\right] = 1 - \mathsf{negl}(\lambda),$$

[5] In the plain public-key (PPK) model, a scheme additionally provides a key-aggregation algorithm $\mathsf{KeyAgg}(K) \to avk$, and verification takes only the aggregated key, i.e., $\mathsf{Verify}(avk, m, \sigma)$.

**Unforgeability in the KOSK model.** We analyze security in the *knowledge-of-secret-key* (KOSK) model. In a *rogue-key attack*, the adversary chooses its verification keys as functions of the honest signers' verification keys, without knowing the corresponding secret keys, and exploits this dependence to forge a valid multi-signature. The KOSK model prevents this by requiring the adversary to reveal a valid secret key for every verification key it contributes. In practice, this is enforced by a proof of possession at key registration.

Fig. 2 formalizes the unforgeability games for a multi-signature scheme. We consider a single honest signer(indexed by 1 w.l.o.g) with key pair $(sk_1, vk_1)$, while the adversary $\mathcal{A}$ controls all remaining signers. $\mathcal{A}$ can initiate any number of concurrent signing sessions with the honest signer through the oracles $\mathsf{SIGN}_1$ and $\mathsf{SIGN}_2$. Finally, $\mathcal{A}$ outputs a forgery consisting of a set of verification keys $K^*$ with $vk_1 \in K^*$, a message $m^*$, one or more signatures, and the secret keys $\{sk_2, \ldots, sk_n\}$ for all dishonest keys in $K^*$. As required by the KOSK model, the game accepts the forgery only if each $(sk_i, vk_i)$ is a valid key pair. The two unforgeability notions differ only in the winning condition:

- **EUF-CMA.** $\mathcal{A}$ wins if it outputs a valid signature on a fresh pair $(K^*, m^*)$, i.e., one on which it completed no signing session with the honest signer.
- **SUF-CMA.** $\mathcal{A}$ may also target a previously signed pair. Let $Q_{ctr}[(K^*, m^*)]$ be the number of sessions it completed on $(K^*, m^*)$. $\mathcal{A}$ wins if it outputs at least $l = Q_{ctr}[(K^*, m^*)] + 1$ pairwise distinct valid signatures on $(K^*, m^*)$, so at least one of them was not obtained from the honest signer.

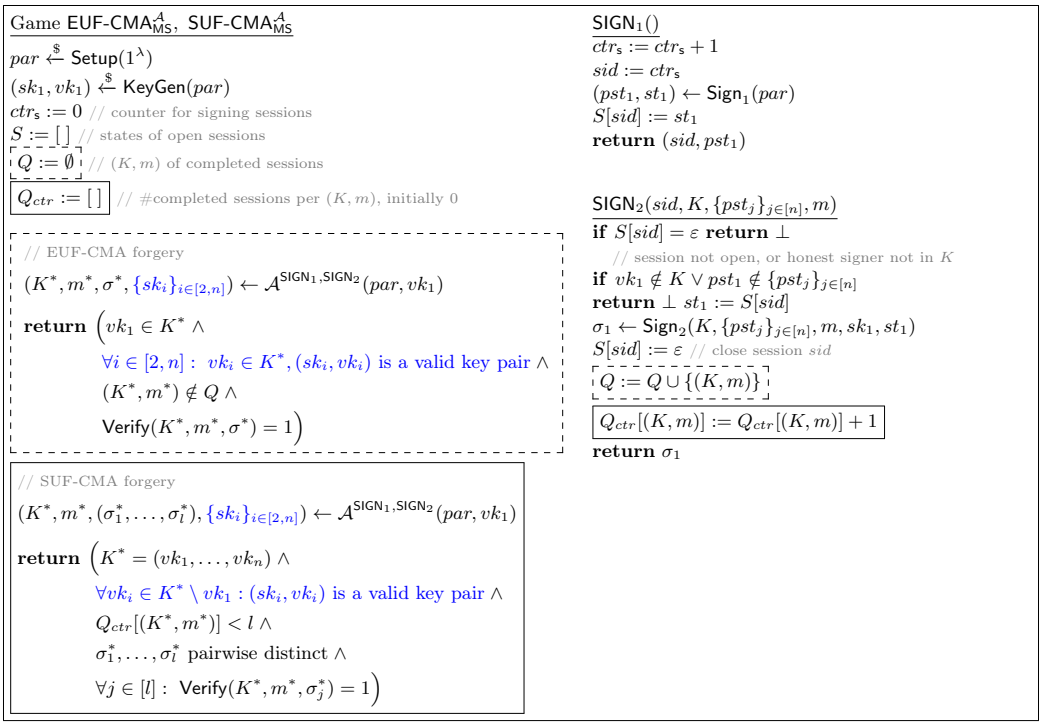



**Fig. 2.** Unforgeability games for a multi-signature scheme $\mathsf{MS}$ in the KOSK model. Blue text marks the KOSK-specific requirements. Dashed and solid boxes mark components that appear only in the EUF-CMA game and only in the SUF-CMA game, respectively.

**Definition 6.** $\mathsf{MS}$ *is* EUF-CMA *(resp.* SUF-CMA*) secure in the KOSK model if, for every PPT adversary* $\mathcal{A}$*,* $\Pr[\mathsf{EUF\text{-}CMA}^{\mathcal{A}}_{\mathsf{MS}} = 1]$ *(resp.* $\Pr[\mathsf{SUF\text{-}CMA}^{\mathcal{A}}_{\mathsf{MS}} = 1]$*) is negligible in* $\lambda$*.*

### 3.7 Blind Multi-Signatures

A blind multi-signature scheme $\mathsf{BMS}$ allows a user $\mathcal{U}$ to obtain a single compact signature on a message $m$ of its choice, from $n$ signers $\{\mathcal{S}_i\}_{i\in[n]}$ with verification keys $K = (vk_1, \ldots, vk_n)$, such that

the signers learn nothing about $m$ or the resulting signature. In this work we focus on BMS schemes with two-round (four-move) signing protocol. Formally, a BMS scheme in the KOSK model[6] consists of the following algorithms:

- $\mathsf{BMS.Setup}(1^\lambda) \to par$: On input the security parameter $\lambda$, outputs the public parameters $par$.
- $\mathsf{BMS.KeyGen}(par) \to (sk_i, vk_i)$: Outputs a signing and verification key pair $(sk_i, vk_i)$ for the signer $\mathcal{S}_i$.
- $\mathsf{BMS.Sign}\langle \mathcal{U}(K, m), \{\mathcal{S}_i(K, sk_i)\}_{i\in[n]}\rangle \to \perp/\sigma$: Let $m$ be the message and $K$ the set of verification keys of the $n$ signers. The user $\mathcal{U}$ and the signers $\{\mathcal{S}_i\}_{i\in[n]}$ execute the following interactive signing protocol on $m$.

  **First Round**
  - $\mathsf{User0}(K, m) \to (st_{\mathcal{U},0}, pst_{\mathcal{U},0})$: The user generates an internal state $st_{\mathcal{U},0}$ and outputs a public state $pst_{\mathcal{U},0}$.
  - $\mathsf{Sign1}(sk_i, K, pst_{\mathcal{U},0}) \to (st_{\mathcal{S}_i}, pst_{\mathcal{S}_i})$: Each signer $\mathcal{S}_i$ uses its signing key $sk_i$ to compute a secret state $st_{\mathcal{S}_i}$ and a public response $pst_{\mathcal{S}_i}$.

  **Second Round**
  - $\mathsf{User1}(st_{\mathcal{U},0}, \{pst_{\mathcal{S}_j}\}_{j\in[n]}) \to (st_{\mathcal{U},1}, pst_{\mathcal{U},1})$: The user updates its internal state to $st_{\mathcal{U},1}$ based on all signer responses and outputs a public response $pst_{\mathcal{U},1}$.
  - $\mathsf{Sign2}(st_{\mathcal{S}_i}, pst_{\mathcal{U},1}) \to (\sigma_i)$: Each signer $\mathcal{S}_i$ uses its state $st_{\mathcal{S}_i}$ and the user's second-round message $pst_{\mathcal{U},1}$ to compute a partial signature $\sigma_i$. It also outputs a status bit $\mathsf{b} \in \{0, 1\}$ indicating whether it successfully completed the computation ($\mathsf{b} = 1$) or not ($\mathsf{b} = 0$).
  - $\mathsf{User2}(st_{\mathcal{U},1}, \{\sigma_j\}_{j\in[n]}) \to \perp/\sigma$: The user takes its internal state $st_{\mathcal{U},1}$ and the partial signatures $\{\sigma_j\}_{j\in[n]}$ of all signers as input, and outputs either the aggregated signature $\sigma$ or $\perp$ if the protocol fails.
- $\mathsf{BMS.Ver}(K, m, \sigma) \to \{0, 1\}$: Outputs 1 if $\sigma$ is a valid signature on $m$ under the key set $K$, and 0 otherwise.

**Correctness.** BMS is correct if, for all parameters $par \leftarrow \mathsf{BMS.Setup}(1^\lambda)$, key sets $K = (vk_1, \ldots, vk_n)$ with $(sk_i, vk_i) \leftarrow \mathsf{BMS.KeyGen}(par)$, and messages $m$, whenever an honest execution of $\mathsf{BMS.Sign}\langle \mathcal{U}(K, m), \{\mathcal{S}_i(K, sk_i)\}_{i\in[n]}\rangle$ outputs $\sigma \neq \perp$,

$$\Pr\left[\mathsf{BMS.Ver}(K, m, \sigma) = 1\right] = 1 - \mathsf{negl}(\lambda),$$

**One-More Unforgeability (OMUF) in the KOSK model.** We formalize the unforgeability of a blind multi-signature scheme BMS in the KOSK model through the one-more unforgeability games $\mathsf{EOMUF}_{\mathsf{BMS}}$ and $\mathsf{OMUF}_{\mathsf{BMS}}$, shown in Figure 3. Without loss of generality, we let signer 1 be the honest signer, while the adversary $\mathcal{A}$ controls all other co-signers and acts as the user. It may initiate an arbitrary number of concurrent signing sessions with the honest signer through two signing oracles, $\mathsf{SIGN}_1$ and $\mathsf{SIGN}_2$, which use $sk_1$ to emulate the honest signer's behavior.

At the end of the game, $\mathcal{A}$ outputs a set of $f$ distinct forgeries $\{(K_l^*, m_l^*, \sigma_l^*)\}_{l\in[f]}$. Since we consider the KOSK model, $\mathcal{A}$ must also provide, for each forgery, the secret keys corresponding to all co-signers in $K_l^*$ other than the honest signer. The $\mathsf{OMUF}_{\mathsf{BMS}}$ game outputs 1 if the following conditions hold:

1. $f > \mathsf{q}_c$, where $\mathsf{q}_c$ denotes the number of successfully completed signing sessions;
2. $vk_1 \in K_l^*$ for every $l \in [f]$;
3. every key in $K_l^* \setminus vk_1$ is accompanied by a valid corresponding secret key; and
4. $\mathsf{BMS.Ver}(K_l^*, m_l^*, \sigma_l^*) = 1$ for every $\ell \in [f]$.

The $\mathsf{EOMUF}_{\mathsf{BMS}}$ game imposes the additional requirement that the pairs $(K_l^*, m_l^*)$ are pairwise distinct.

**Definition 7 (One-More Unforgeability (OMUF) [7]).** *A blind multi-signature scheme* BMS *satisfies OMUF if for all adversaries* $\mathcal{A}$

$$\mathsf{Adv}^{\mathsf{OMUF}}_{\mathsf{BMS},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{OMUF}^{\mathcal{A}}_{\mathsf{BMS}}(\lambda)\right] \leq \mathsf{negl}(\lambda),$$

*where* $\mathsf{OMUF}_{\mathsf{BMS}}$ *game is defined in Fig. 3.*

[6] In the plain public-key (PPK) model, schemes additionally provide a key-aggregation algorithm $\mathsf{BMS.KeyAgg}(K) \to avk$, and verification takes only the aggregated key, i.e., $\mathsf{BMS.Ver}(avk, m, \sigma)$.

[7] An analogous definition holds for EOMUF by replacing the OMUF game with the corresponding EOMUF game.

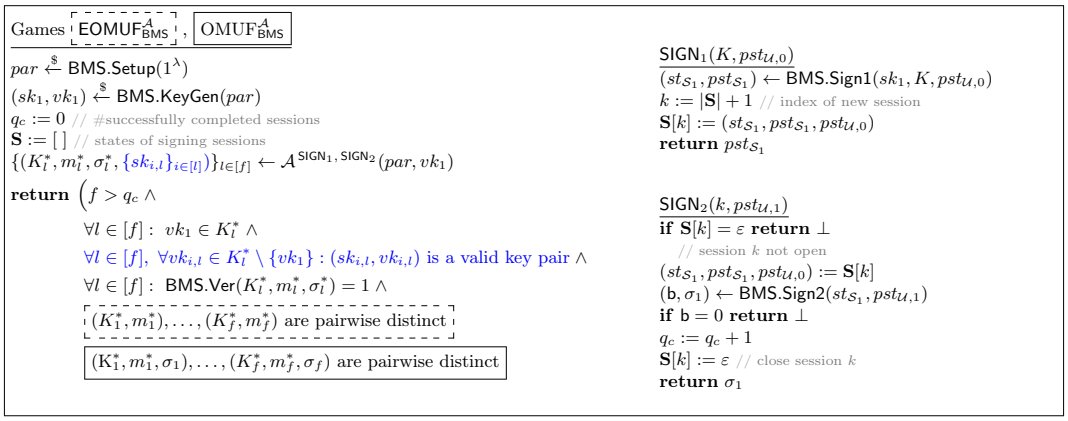



**Fig. 3.** One-more unforgeability games $\mathsf{OMUF}_{\mathsf{BMS}}$ and $\mathsf{EOMUF}_{\mathsf{BMS}}$ for a blind multi-signature scheme BMS in the KOSK model. The dashed box is included only in the EOMUF game, whereas the solid box is included only in the OMUF game.

**Blindness.** We formalize the blindness of a blind multi-signature scheme BMS through blindness game, shown in Fig. 4. In this game, the adversary $\mathcal{A}$ controls all $n$ corrupted signers and is represented by two algorithms $(\mathcal{A}_1, \mathcal{A}_2)$. In the first phase, $\mathcal{A}_1$ outputs a vector of verification keys $\hat{K}$ and two challenge messages $(m_0, m_1)$. Next, the challenger samples a bit $\mathsf{b} \in \{0,1\}$ and runs two concurrent signing sessions $k \in \{0,1\}$: session 0 signs $m_{\mathsf{b}}$ and session 1 signs $m_{1-\mathsf{b}}$. The honest user's behaviour in each session is simulated by the oracles $\mathsf{USER}_0$, $\mathsf{USER}_1$, and $\mathsf{USER}_2$. If either session fails, the game outputs $(\perp, \perp)$. Otherwise, once both sessions complete successfully, the adversary $\mathcal{A}$ receives the final signatures $\sigma_0$ and $\sigma_1$ on messages $m_0$ and $m_1$, respectively. At the end, $\mathcal{A}$ outputs a guess bit $\mathsf{b}'$ and wins the game if $\mathsf{b}' = \mathsf{b}$. Hence, any non-negligible advantage in guessing $\mathsf{b}$ means that $\mathcal{A}$ can determine the session corresponding to the signature, thereby violating blindness.

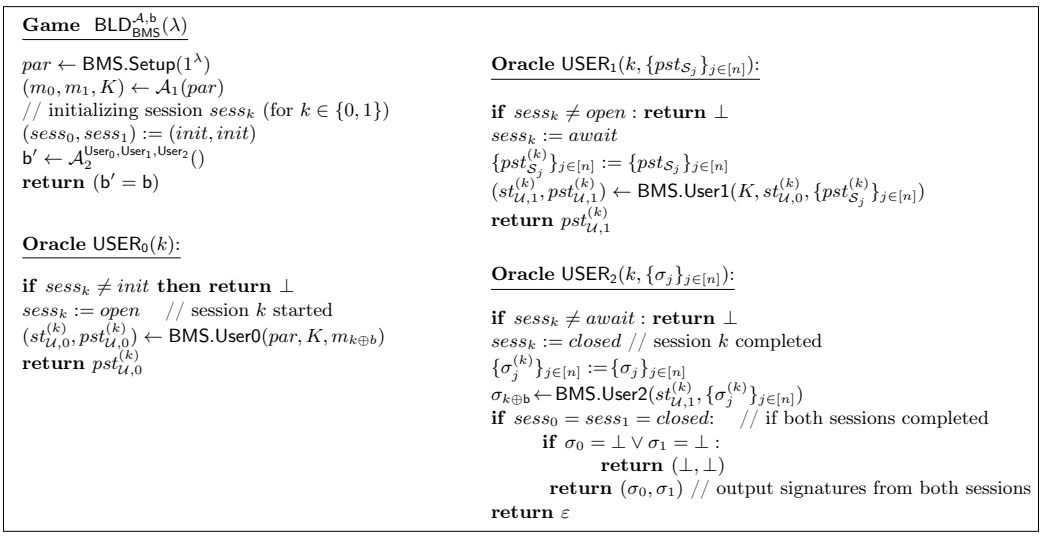



**Fig. 4.** Blindness game for BMS.

**Definition 8.** *A blind multi-signature scheme* BMS *satisfies blindness if for all adversaries* $\mathcal{A}$

$$\mathsf{Adv}^{\mathsf{BLD}}_{\mathsf{BMS},\mathcal{A}}(\lambda) := \Pr\left[\mathsf{BLD}^{\mathcal{A},1}_{\mathsf{BMS}}(\lambda)\right] - \Pr\left[\mathsf{BLD}^{\mathcal{A},0}_{\mathsf{BMS}}(\lambda)\right] \leq \mathsf{negl}(\lambda),$$

*where* $\mathsf{BLD}^{\mathcal{A},\mathsf{b}}_{\mathsf{BMS}}$ *(for* $\mathsf{b} \in \{0,1\}$*) is the game defined in Fig. 4.*

## 4 Multi-signature scheme MRaccoon

Figure 5 presents MRaccoon, our base multi-signature scheme for $n$ signers. It consists of the unboxed components only. The boxed components belong to its intermediate extension MRaccoon$^\star$, which we introduce in Section 5.

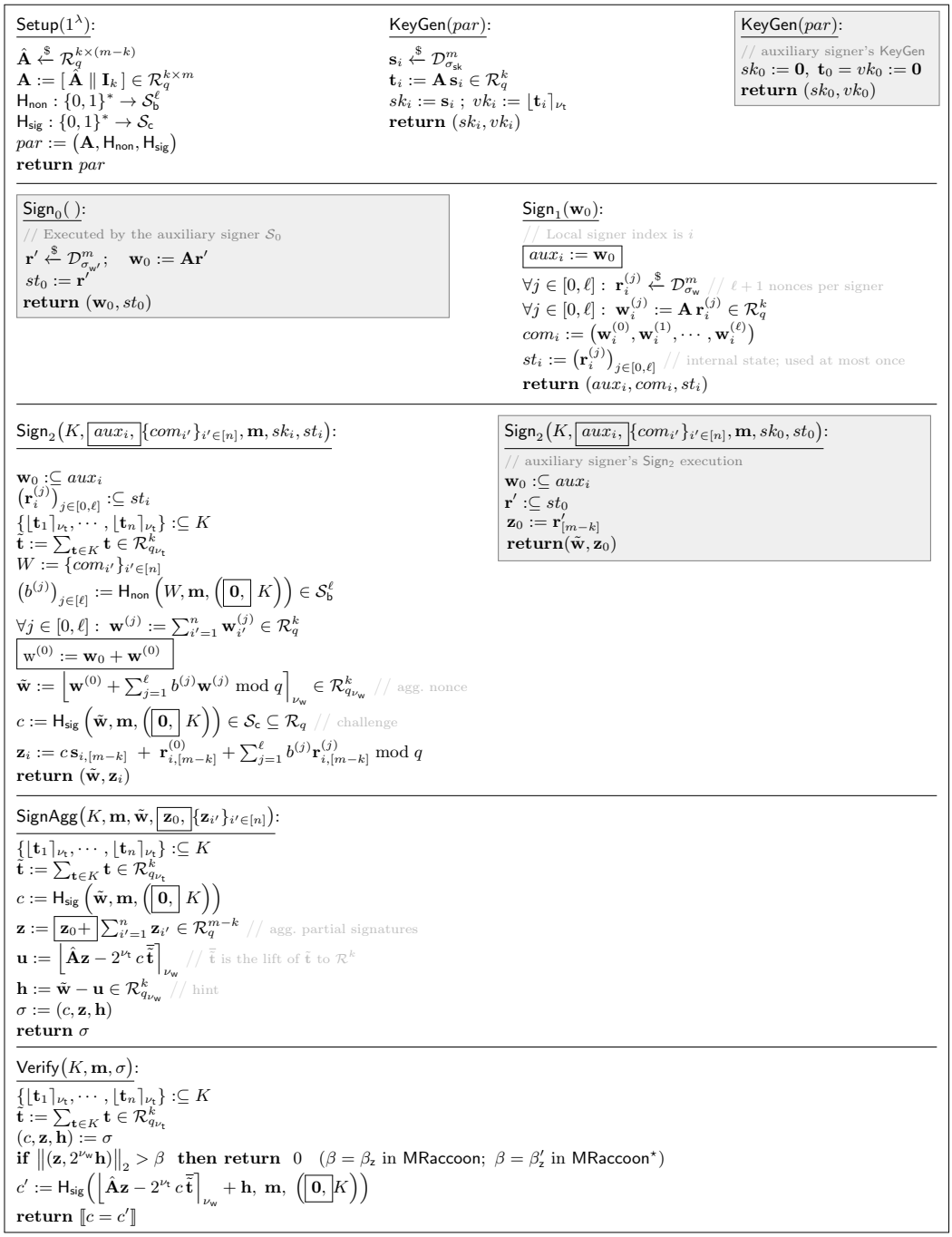



**Fig. 5.** MRaccoon and MRaccoon$^\star$: MRaccoon excludes all boxed components and MRaccoon$^\star$ includes them; grey boxes denote auxiliary signer's executions in MRaccoon$^\star$.

### 4.1 Security

**Theorem 1 (Correctness of MRaccoon).** *Let* $\beta_{\mathsf{sk}} := e^{1/4}\sigma_{\mathsf{sk}}\sqrt{Nm}$ *and* $\beta_{\mathsf{w}} := e^{1/4}\sigma_{\mathsf{w}}\sqrt{Nm}$. *Assume that* $\|c\|_1 \le \beta_{\mathsf{c}}$ *for every* $c \in \mathcal{S}_{\mathsf{c}}$ *and* $\|b\|_1 \le 1$ *for every* $b \in \mathcal{S}_{\mathsf{b}}$. *If*

$$\beta_{\mathsf{z}} \ge \sqrt{2}\, n\big(\beta_{\mathsf{c}}\beta_{\mathsf{sk}} + (1+\ell)\beta_{\mathsf{w}}\big) + \big((3n-1)\beta_{\mathsf{c}}\, 2^{\nu_{\mathsf{t}}-1} + 4\cdot 2^{\nu_{\mathsf{w}}-1}\big)\sqrt{Nk},$$

*then* MRaccoon *with* $n$ *signers is correct, and an honestly generated signature is rejected with probability at most* $n(\ell+2)\,2^{-Nm/10}$.

*Proof.* Consider an honest execution with signers holding $(sk_i, vk_i) = (\mathbf{s}_i, \lfloor \mathbf{t}_i \rceil_{\nu_{\mathsf{t}}})$ for $i \in [n]$. For $\mathbf{x} \in \mathcal{R}_q^m$, write $\mathbf{x} = (\mathbf{x}_{[m-k]}, \mathbf{x}_{(k)})$ with $\mathbf{x}_{(k)} \in \mathcal{R}_q^k$ the last $k$ components, so that $\mathbf{A}\mathbf{x} = \hat{\mathbf{A}}\mathbf{x}_{[m-k]} + \mathbf{x}_{(k)}$. Let

$$\mathbf{s}_{agg} := \sum_{i\in[n]} \mathbf{s}_i, \qquad \mathbf{r}_{agg} := \sum_{i\in[n]} \Big(\mathbf{r}_i^{(0)} + \sum_{j\in[\ell]} b^{(j)}\mathbf{r}_i^{(j)}\Big), \qquad \mathbf{t}_{agg} := \sum_{i\in[n]} \mathbf{t}_i = \mathbf{A}\mathbf{s}_{agg}.$$

All signers run $\mathsf{Sign}_2$ on the same input $(W, \mathbf{m}, K)$, so they compute the same nonce coefficients $(b^{(j)})_{j\in[\ell]}$, the same $\tilde{\mathbf{w}}$ and the same challenge $c$, which $\mathsf{SignAgg}$ recomputes. The unrounded aggregated nonce satisfies $\mathbf{w}' := \mathbf{w}^{(0)} + \sum_{j\in[\ell]} b^{(j)}\mathbf{w}^{(j)} = \mathbf{A}\mathbf{r}_{agg} \bmod q$ and $\tilde{\mathbf{w}} = \lfloor \mathbf{w}' \rceil_{\nu_{\mathsf{w}}}$, and the aggregated response is $\mathbf{z} = c\,\mathbf{s}_{agg,[m-k]} + \mathbf{r}_{agg,[m-k]} \bmod q$. Let $\mathbf{u} := \lfloor \hat{\mathbf{A}}\mathbf{z} - 2^{\nu_{\mathsf{t}}} c\,\bar{\bar{\mathbf{t}}} \rceil_{\nu_{\mathsf{w}}}$. Since $\mathsf{SignAgg}$ sets $\mathbf{h} := \tilde{\mathbf{w}} - \mathbf{u}$, the verifier recomputes $\lfloor \mathbf{y} \rceil_{\nu_{\mathsf{w}}} + \mathbf{h} = \tilde{\mathbf{w}}$ and hence $c' = \mathsf{H}_{\mathsf{sig}}(\tilde{\mathbf{w}}, \mathbf{m}, K) = c$. It therefore remains to bound $\|(\mathbf{z}, 2^{\nu_{\mathsf{w}}}\mathbf{h})\|_2$.

By Lemma 6 and a union bound over the $n$ secret keys and the $n(\ell+1)$ nonces, we have $\|\mathbf{s}_i\| \le \beta_{\mathsf{sk}} := e^{1/4}\sigma_{\mathsf{sk}}\sqrt{Nm}$ and $\|\mathbf{r}_i^{(j)}\| \le \beta_{\mathsf{w}} := e^{1/4}\sigma_{\mathsf{w}}\sqrt{Nm}$, except with probability at most $n(\ell+2)\,2^{-Nm/10}$. Since $\|b^{(j)}\mathbf{x}\| \le \|b^{(j)}\|_1\|\mathbf{x}\|$, it follows that

$$\|\mathbf{s}_{agg}\| \le n\beta_{\mathsf{sk}}, \qquad \|\mathbf{r}_{agg}\| \le n(1+\ell)\beta_{\mathsf{w}}.$$

Let $q_{\perp,t}$ and $q_{\perp,w}$ for $q_\perp$ with respect to $\nu_t$ and $\nu_w$; by assumption, $q_{\perp,t} \le 2^{\nu_t-1}-1$ and $q_{\perp,w} \le 2^{\nu_w-1}-1$. Below, we identify elements of $\mathcal{R}_{q_\nu}^k$ with their canonical lifts.

For each $i \in [n]$, since $\mathbf{vk}_i = \lfloor \mathbf{t}_i \rceil_{\nu_t}$, Lemma 3 gives $\mathbf{e}_i \in [-2^{\nu_t-1}, 2^{\nu_t-1}-1]^{Nk}$ and $\mathbf{d}_i \in \{0,1\}^{Nk}$ with $2^{\nu_t}\mathbf{vk}_i \equiv \mathbf{t}_i - \mathbf{e}_i + q_{\perp,t}\mathbf{d}_i \pmod q$. Since $\bar{\mathbf{t}} = \sum_{i\in[n]} \mathbf{vk}_i \bmod q_{\nu_t}$, there exists $\boldsymbol{\kappa} \in \{0,\dots,n-1\}^{Nk}$ with $\bar{\mathbf{t}} - \sum_i \mathbf{vk}_i = -q_{\nu_t}\boldsymbol{\kappa}$, and Corollary 1 yields $2^{\nu_t}\bar{\mathbf{t}} \equiv \sum_i 2^{\nu_t}\mathbf{vk}_i + q_{\perp,t}\boldsymbol{\kappa} \pmod q$. Hence

$$2^{\nu_t}\bar{\mathbf{t}} \equiv \mathbf{t}_{agg} + \boldsymbol{\varepsilon}_t \pmod q, \qquad \boldsymbol{\varepsilon}_t := q_{\perp,t}\Big(\boldsymbol{\kappa} + \sum_{i\in[n]} \mathbf{d}_i\Big) - \sum_{i\in[n]} \mathbf{e}_i,$$

and, as $\boldsymbol{\kappa} + \sum_i \mathbf{d}_i \in \{0,\dots,2n-1\}^{Nk}$, $\|\boldsymbol{\varepsilon}_t\|_\infty \le (2n-1)(2^{\nu_t-1}-1) + n\,2^{\nu_t-1} \le (3n-1)2^{\nu_t-1}$.

Using $\mathbf{t}_{agg} = \hat{\mathbf{A}}\mathbf{s}_{agg,[m-k]} + \mathbf{s}_{agg,[m-k+1\dots k]}$, we obtain $\mathbf{y} = \hat{\mathbf{A}}\mathbf{r}_{agg,[m-k]} - c\,\mathbf{s}_{agg,[m-k+1\dots k]} - c\,\boldsymbol{\varepsilon}_t \pmod q$, and therefore

$$\mathbf{w}' = \mathbf{y} + \boldsymbol{\delta} \pmod q, \qquad \boldsymbol{\delta} := \mathbf{r}_{agg,[m-k+1\dots k]} + c\,\mathbf{s}_{agg,[m-k+1\dots k]} + c\,\boldsymbol{\varepsilon}_t.$$

We now bound the hint. Lemma 3, applied to $\bar{\mathbf{w}} = \lfloor \mathbf{w}' \rceil_{\nu_w}$ and to $\lfloor \mathbf{y} \rceil_{\nu_w}$, gives $\mathbf{e}_{w'}, \mathbf{e}_y \in [-2^{\nu_w-1}, 2^{\nu_w-1}-1]^{Nk}$ and $\mathbf{d}_{w'}, \mathbf{d}_y \in \{0,1\}^{Nk}$ with

$$2^{\nu_w}\bar{\mathbf{w}} \equiv \mathbf{w}' - \mathbf{e}_{w'} + q_{\perp,w}\mathbf{d}_{w'}, \qquad 2^{\nu_w}\lfloor \mathbf{y} \rceil_{\nu_w} \equiv \mathbf{y} - \mathbf{e}_y + q_{\perp,w}\mathbf{d}_y \pmod q.$$

Since $\mathbf{h} = \bar{\mathbf{w}} - \lfloor \mathbf{y} \rceil_{\nu_w} \bmod q_{\nu_w}$, there exists $\mathbf{d}_h \in \{0,1\}^{Nk}$ with $\mathbf{h} - \bar{\mathbf{w}} + \lfloor \mathbf{y} \rceil_{\nu_w} = q_{\nu_w}\mathbf{d}_h$, and Corollary 1 together with $\mathbf{w}' - \mathbf{y} = \boldsymbol{\delta}$ yields

$$2^{\nu_w}\mathbf{h} \equiv \boldsymbol{\delta} - \mathbf{e}_{w'} + \mathbf{e}_y + q_{\perp,w}\big(\mathbf{d}_{w'} - \mathbf{d}_y - \mathbf{d}_h\big) \pmod q.$$

The coefficients of $\mathbf{e}_y - \mathbf{e}_{w'}$ lie in $[-(2^{\nu_w}-1), 2^{\nu_w}-1]$ and those of $\mathbf{d}_{w'} - \mathbf{d}_y - \mathbf{d}_h$ in $\{-2,\dots,1\}$, so the last three terms have $\infty$-norm at most $2^{\nu_w} - 1 + 2q_{\perp,w} \le 4\cdot 2^{\nu_w-1}$. Using $\|c\,\boldsymbol{\varepsilon}_t\| \le \sqrt{Nk}\,\|c\|_1\|\boldsymbol{\varepsilon}_t\|_\infty$, we get

$$\|2^{\nu_w}\mathbf{h} \bmod q\| \le \|\mathbf{r}_{agg,[m-k+1\dots k]}\| + \beta_c\|\mathbf{s}_{agg,[m-k+1\dots k]}\| + \big((3n-1)\beta_c 2^{\nu_t-1} + 4\cdot 2^{\nu_w-1}\big)\sqrt{Nk}.$$

Moreover, $\|\mathbf{z}\| \le \beta_c\|\mathbf{s}_{agg,[m-k]}\| + \|\mathbf{r}_{agg,[m-k]}\|$. Since $\|\mathbf{x}_{[m-k]}\| + \|\mathbf{x}_{(k)}\| \le \sqrt{2}\|\mathbf{x}\|$, we conclude that

$$\big\|(\mathbf{z}, 2^{\nu_w}\mathbf{h})\big\|_2 \le \sqrt{2}\beta_c\|\mathbf{s}_{agg}\| + \sqrt{2}\|\mathbf{r}_{agg}\| + \big((3n-1)\beta_c 2^{\nu_t-1} + 4\cdot 2^{\nu_w-1}\big)\sqrt{Nk} \le \beta_z.$$

□

**Theorem 2 (EUF-CMA security of MRaccoon).** *Let $q = q(\lambda)$, $k = k(\lambda)$, $m = m(\lambda)$ be integers. Let $\mathcal{A}$ be an EUF-CMA adversary against* MRaccoon *in the KOSK model that outputs forgeries for signer sets of size at most $n$ and makes at most $q_s$ signing queries and $q_h$ random-oracle queries, there exists an AOM-MISIS adversary $\mathcal{C}$ running in time roughly two times that of $\mathcal{A}$ such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{MRaccoon},\mathcal{A}}(\lambda) \le \sqrt{\mathsf{q}\,\mathsf{Adv}^{\mathsf{AOM\text{-}MISIS}}_{\mathsf{par}}(\mathcal{C},\lambda) + 8\mathsf{q}^3 2^{-2\lambda}}\,,$$

*where $\mathsf{q} = q_h + q_s + 1$ and $\mathsf{par} = \big(q, k, m, Q = 1+\mathsf{q}_s(1+\ell), (\sigma_i)_{i\in[Q]}, \beta_{\mathsf{s}}, \beta_{\mathsf{b}}, \beta_{\mathsf{u}}\big)$ with $\beta_{\mathsf{s}} = 4\beta_{\mathsf{z}} + \big(2^{\nu_{\mathsf{w}}+1} + (3n-1)\beta_{\mathsf{c}}\, 2^{\nu_{\mathsf{t}}}\big)\sqrt{Nk} + 2\sqrt{2}(n-1)\beta_{\mathsf{c}}\beta_{\mathsf{sk}}, \beta_{\mathsf{b}} = 2\sigma_{\mathsf{sk}}\beta_{\mathsf{c}}, \beta_{\mathsf{u}} = 2/\sigma_{\mathsf{sk}} + \beta_{\mathsf{c}}\sqrt{8N\mathsf{q}_s}/\sigma_{\mathsf{w}}$, and $\sigma_1 = \sigma_{\mathsf{sk}}$, $\sigma_{1+i} = \sigma_{\mathsf{w}}$ for $i \in [\mathsf{q}_s(\ell+1)]$.*

*Proof.* Let $\mathcal{A}$ be an EUF-CMA adversary as described in the theorem. We assume that whenever $\mathcal{A}$ returns $(K^*, \mathbf{m}^*, (c^*, \mathbf{z}^*, \mathbf{h}^*))$, it has already queried $\mathsf{H}_{\mathsf{sig}}(\tilde{\mathbf{w}}^*, \mathbf{m}^*, K^*)$, where $\tilde{\mathbf{w}}^*$ is the aggregated nonce recomputed during verification. This adds at most one random-oracle query. The game itself makes at most one query to each of $\mathsf{H}_{\mathsf{non}}$ and $\mathsf{H}_{\mathsf{sig}}$ per signing query, so each of these two oracles receives at most $\mathsf{q} = q_h + q_s + 1$ queries in total.

We now construct an algorithm $\mathcal{B}$ compatible with the following forking lemma and construct $\mathcal{C}$ from $\mathsf{Fork}^{\mathcal{B}}$.

**Lemma 9 (General Forking Lemma [ZT25]).** *Let $q \ge 1$ be an integer, $S \subseteq [q]$ be a set, and* HG *be an algorithm that outputs $h_1, \ldots, h_q$ where each $h_i$ is independently sampled. Let $\mathcal{A}$ be a randomized algorithm that on input $x, h_1, \ldots, h_q$ outputs a pair $(I, \mathsf{Out})$, where $I \in \{\perp\} \cup S$ and* Out *is a side output. Let* IG *be a randomized algorithm that generates $x$. The accepting probability of $\mathcal{A}$ is defined as*

$$\mathsf{acc}(\mathcal{A}) = \Pr_{x \xleftarrow{\$} \mathsf{IG},\, h_1,\ldots,h_q \xleftarrow{\$} \mathsf{HG}} \big[(I, \mathsf{Out}) \xleftarrow{\$} \mathcal{A}(x, h_1, \ldots, h_q) \,:\, I \neq \perp\big].$$

*Consider the algorithm $\mathsf{Fork}^{\mathcal{A}}(x)$ that proceeds as follows:*

1. *Sample random coins $\rho$ for $\mathcal{A}$ and $h_1, \ldots, h_q \xleftarrow{\$}$ HG, and run $(I, \mathsf{Out}) \leftarrow \mathcal{A}(x, h_1, \ldots, h_q; \rho)$. If $I = \perp$, return $\perp$.*
2. *Sample fresh $h'_I, \ldots, h'_q \xleftarrow{\$}$ HG, and run $(I', \mathsf{Out}') \leftarrow \mathcal{A}(x, h_1, \ldots, h_{I-1}, h'_I, \ldots, h'_q; \rho)$.*
3. *If $I' = I$, return $\alpha = \big(I, \mathsf{Out}, \mathsf{Out}'\big)$; otherwise, return $\perp$.*

*The accepting probability of $\mathsf{Fork}^{\mathcal{A}}$ is defined as*

$$\mathsf{acc}(\mathsf{Fork}^{\mathcal{A}}) = \Pr_{x \xleftarrow{\$} \mathsf{IG}} \big[\alpha \xleftarrow{\$} \mathsf{Fork}^{\mathcal{A}}(x) \,:\, \alpha \neq \perp\big].$$

*Then, $\mathsf{acc}(\mathsf{Fork}^{\mathcal{A}}) \ge \mathsf{acc}(\mathcal{A})^2/|S|$.*

**Construction of $\mathcal{B}$.** Let $\mathcal{B}$ be the algorithm that simulates the EUF-CMA game for $\mathcal{A}$ as specified in Fig. 6. Its input consists of $\mathbf{A}$, the vectors $(\mathbf{p}_i)_{i\in[1+q_s(\ell+1)]}$ and a list of hash values $h_1, \ldots, h_{2\mathsf{q}}$. Here, $\mathbf{A}$ and the $\mathbf{p}_i$ are sampled as in the AOM-MISIS game, and for each $i \in [\mathsf{q}]$, $h_{2i-1}$ is uniform in $\mathcal{S}^{\ell}_{\mathsf{b}}$ and $h_{2i}$ is uniform in $\mathcal{S}_{\mathsf{c}}$.

$\mathcal{B}$ uses the AOM-MISIS instance as the honest signer's verification key and as the nonces returned by $\mathsf{SIGN}_1$. For each query to $\mathsf{SIGN}_2$, it calls the PI oracle of the AOM-MISIS game to compute the response.

Let $\mathcal{A}$ output a valid forgery $(K^*, \mathbf{m}^*, \sigma^* := (c^*, \mathbf{z}^*, \mathbf{h}^*))$. Then, $\mathcal{B}$ computes the aggregated verification key $\tilde{\mathbf{t}}^* := \sum_{\mathbf{t}\in K^*} \mathbf{t}$ and the corresponding aggregated nonce $\tilde{\mathbf{w}}^* := \big\lfloor \hat{\mathbf{A}}\mathbf{z}^* - 2^{\nu_{\mathsf{t}}} c^* \widetilde{\tilde{\mathbf{t}}^*} \big\rceil_{\nu_{\mathsf{w}}} + \mathbf{h}^*$. Using the simulated hash transcripts, $\mathcal{B}$ identifies the index $i_{\mathsf{sig}} := T_{\mathsf{isig}}(\tilde{\mathbf{w}}^*, \mathbf{m}^*, K^*)$. Finally, $\mathcal{B}$ outputs $\big(i_{\mathsf{sig}}, K^*, \mathbf{m}^*, \sigma^*, (sk_2, \ldots, sk_n)\big)$.

**Analysis of $\mathcal{B}$.** We apply Lemma 9 to $\mathcal{B}$ with $2\mathsf{q}$ hash values and $S := \{2j\}_{j\in[\mathsf{q}]}$. The algorithm IG samples $(\mathbf{A}, (\mathbf{p}_i)_{i\in[1+q_s(\ell+1)]})$ as in the AOM-MISIS game, and HG samples $h_{2i-1}$ uniformly from $\mathcal{S}^{\ell}_{b}$ and $h_{2i}$ uniformly from $\mathcal{S}_{c}$ for each $i \in [\mathsf{q}]$. By construction, the index $i_{\mathsf{sig}}$ output by $\mathcal{B}$ always lies in $S$. Moreover, $\mathcal{B}$ perfectly simulates the EUF-CMA game, so

$$\mathsf{acc}(\mathcal{B}) \ge \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{MRaccoon},\mathcal{A}}(\lambda),$$

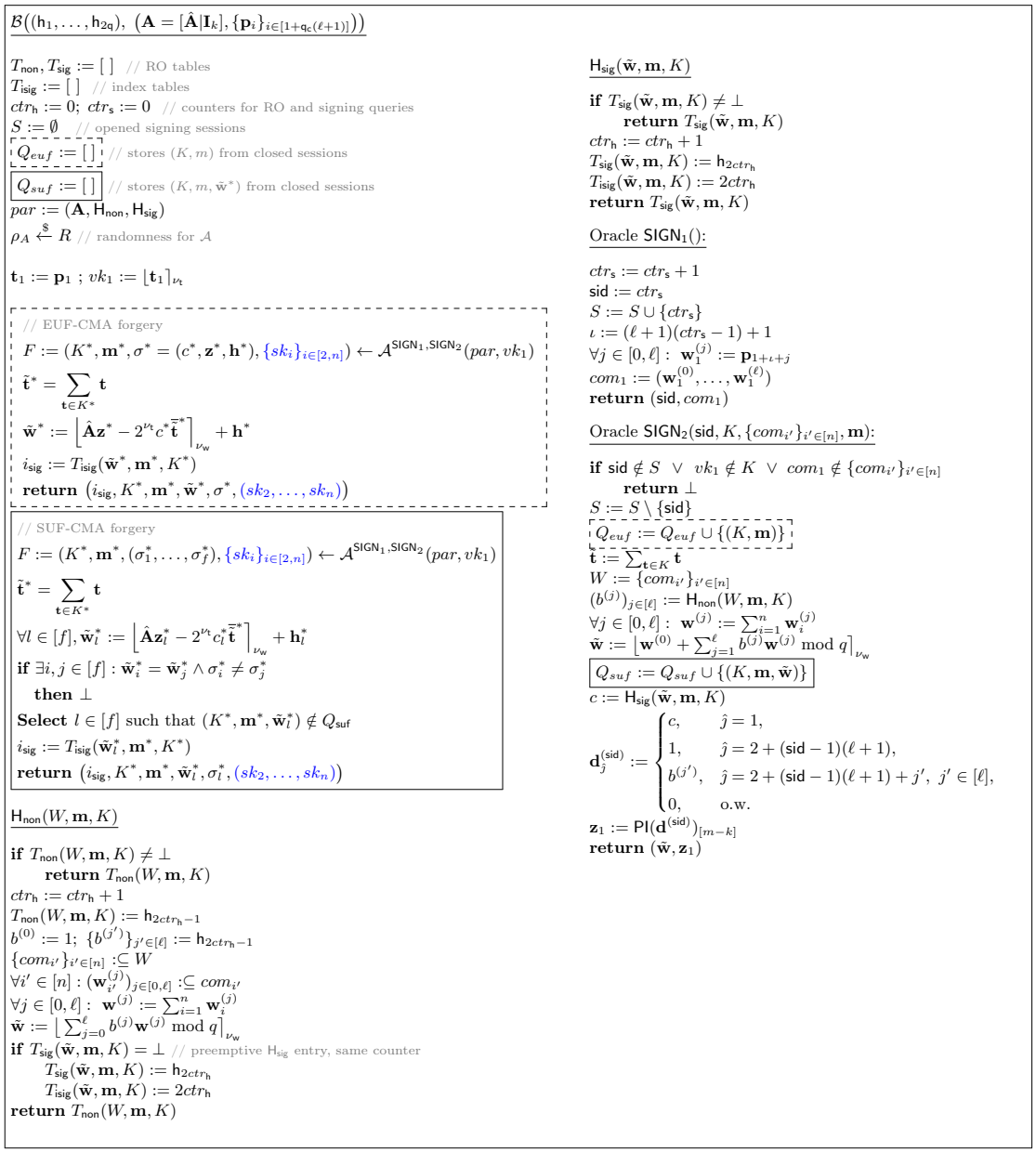



**Fig. 6.** The simulator $\mathcal{B}$

and Lemma 9 gives

$$\mathsf{acc}(\mathsf{Fork}^{\mathcal{B}}) \geq \frac{\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{MRaccoon},\mathcal{A}}(\lambda)^2}{\mathsf{q}}.$$

**Construction of $\mathcal{C}$.** Algorithm $\mathcal{C}$ runs $\mathsf{Fork}^{\mathcal{B}}$ on its AOM-MISIS instance $(\mathbf{A}, \{\mathbf{p}_i\}_{i\in[1+q_s(\ell+1)]})$ and answers the $\mathsf{PI}$ queries of both executions of $\mathcal{B}$ with its own oracle. In detail, it picks random coins $\rho_\mathcal{B}$ and runs $\mathcal{B}$ on these coins, the instance and hash values $h_1, \ldots, h_{2\mathsf{q}}$. If $\mathcal{B}$ returns $\perp$, so does $\mathcal{C}$. Otherwise, $\mathcal{B}$ returns $(i_{\mathsf{sig}}, K^*, \mathbf{m}^*, \tilde{\mathbf{w}}^*, \sigma^*, (sk_2, \ldots, sk_n))$.

$\mathcal{C}$ then reruns $\mathcal{B}$ on the same coins and instance, but with the hash values $h_1, \ldots, h_{i_{\mathsf{sig}}-1}, h'_{i_{\mathsf{sig}}}, \ldots, h'_{2\mathsf{q}}$. If this run returns $\perp$, so does $\mathcal{C}$. Otherwise, the run returns $(i'_{\mathsf{sig}}, K^{*\prime}, \mathbf{m}^{*\prime}, \tilde{\mathbf{w}}^{*\prime}, \sigma^{*\prime}, (sk'_2, \ldots, sk'_n))$. If $i_{\mathsf{sig}} \neq i'_{\mathsf{sig}}$, or if $h_{i_{\mathsf{sig}}} = h'_{i_{\mathsf{sig}}}$, then $\mathcal{C}$ returns $\perp$.

Otherwise, the two runs are identical until $\mathcal{B}$ first reads its $i_{\mathsf{sig}}$-th hash value. In particular, every slot $i \leq i_{\mathsf{sig}}$ is assigned to the same input in both runs, and the input $x_{i_{\mathsf{sig}}}$ of slot $i_{\mathsf{sig}}$ is the $\mathsf{H}_{\mathsf{sig}}$ input of both forgeries: $(\tilde{\mathbf{w}}^*, \mathbf{m}^*, K^*) = (\tilde{\mathbf{w}}^{*\prime}, \mathbf{m}^{*\prime}, K^{*\prime}) = x_{i_{\mathsf{sig}}},\ c^* = h_{i_{\mathsf{sig}}},\ c^{*\prime} = h'_{i_{\mathsf{sig}}}$.

Thus, $\mathcal{C}$ returns $\big(i_{\mathsf{sig}}, Out := (K^*, \mathbf{m}^*, \mathbf{w}^*, \sigma^* = (c^*, \mathbf{z}^*, \mathbf{h}^*), \mathbf{s}_{\mathsf{cs}}), \widetilde{Out} := (K^{*\prime}, \mathbf{m}^{*\prime}, \mathbf{w}^{*\prime}, \sigma^{*\prime} = (c^{*\prime}, \mathbf{z}^{*\prime}, \mathbf{h}^{*\prime}), \mathbf{s}'_{\mathsf{cs}})\big)$, where $\mathbf{s}_{\mathsf{cs}} = \sum_{i\in[2,n]} sk_i$ and $\mathbf{s}'_{\mathsf{cs}} = \sum_{i\in[2,n]} sk'_i$.

**The AOM-MISIS solution.** If $\mathcal{C}$ does not abort, it uses the two forgeries $(c^*, \mathbf{z}^*, \mathbf{h}^*)$ and $(c^{*\prime}, \mathbf{z}^{*\prime}, \mathbf{h}^{*\prime})$ to construct an AOM-MISIS solution $(\hat{\mathbf{s}}, \hat{\mathbf{b}}, \mathbf{u})$. Let $\Delta c := c^* - c^{*\prime}$, $\Delta\mathbf{z} := \mathbf{z}^* - \mathbf{z}^{*\prime}$, $\mathbf{s}_{\mathsf{cs}} := \sum_{i=2}^{n} sk_i \in \mathcal{R}^m$, $\mathbf{t}_i := \mathbf{A}sk_i$ for $i \geq 2$, and $\tilde{\mathbf{t}}_{ur} := \mathbf{p}_1 + \sum_{i=2}^{n} \mathbf{t}_i \in \mathcal{R}_q^k$. Then, $\mathcal{C}$ sets

$$\hat{\mathbf{s}} := \big(\Delta\mathbf{z},\ \Delta c\,\tilde{\mathbf{t}}_{ur} - \hat{\mathbf{A}}\Delta\mathbf{z}\big) - \Delta c\,\mathbf{s}_{\mathsf{cs}} \in \mathcal{R}^m, \qquad \hat{\mathbf{b}} := (\Delta c, 0, \ldots, 0) \in \mathcal{R}^Q. \tag{1}$$

Since $\mathbf{A} = [\hat{\mathbf{A}} \,\|\, \mathbf{I}_k]$, we have $\mathbf{A}\,\mathbf{s}_{\mathsf{cs}} = \sum_{i\geq 2} \mathbf{t}_i$, and hence

$$\mathbf{A}\hat{\mathbf{s}} = \hat{A}\Delta\mathbf{z} + \big(\Delta c\,\tilde{\mathbf{t}}_{ur} - \hat{A}\Delta\mathbf{z}\big) - \Delta c \sum_{i\geq 2} \mathbf{t}_i = \Delta c\,\mathbf{p}_1 = \sum_{i\in[Q]} \hat{b}_i\mathbf{p}_i \pmod q.$$

It remains to define $\mathbf{u}$ so that $\hat{\mathbf{b}}^T\mathbf{u} \neq 0 \pmod q$, and $\mathbf{d}^T\mathbf{u} = 0 \pmod q$ for every query $\mathsf{PI}(\mathbf{d})$ made in either execution of $\mathcal{B}$. We set $u_1 = 2$, so that $\hat{\mathbf{b}}^T\mathbf{u} = 2\Delta c$. Since $c^* \neq c^{*\prime}$ are both in $\mathcal{S}_c$, the polynomial $2\Delta c$ is short and non-zero, so $\hat{\mathbf{b}}^T\mathbf{u} \neq 0 \pmod q$.

We call the $\ell + 1$ nonces of the $j$-th $\mathsf{SIGN}_1$ query the *$j$-th token*. The remaining entries of $\mathbf{u}$ split into blocks of length $\ell + 1$, one per token. For $j = 1, \ldots, q_s$ in turn, $\mathcal{C}$ sets the $j$-th block $\mathbf{v} := (v_1, \ldots, v_{\ell+1}) := \mathbf{u}_{1+(j-1)(\ell+1)+[\ell+1]}$ such that

$$\sum_{i\in[1+j(\ell+1)]} u_i d_i = 0 \pmod q \quad \text{for each } \mathsf{PI} \text{ query } \mathbf{d} \text{ on one of the first } j \text{ tokens.} \tag{2}$$

When the $j$-th block is set, all earlier entries of $\mathbf{u}$ are already fixed. A $\mathsf{PI}$ query on an earlier token is 0 on the $j$-th block, so Eq. (2) for $j-1$ carries over to such queries for $j$. After the last block, Eq. (2) for $j = q_s$ gives $\mathbf{d}^T\mathbf{u} = 0 \pmod q$ for every $\mathsf{PI}$ query $\mathbf{d}$.

By the simulation, at most one $\mathsf{SIGN}_2$ query, and hence at most one $\mathsf{PI}$ query, uses the $j$-th token in each execution of $\mathcal{B}$. There are therefore three cases.

**Case 1:** No query uses the $j$-th token in either execution. Then $\mathbf{v}$ is set to $\mathbf{0}$.

**Case 2:** Exactly one query $\mathbf{d}$ uses the $j$-th token across the two executions. Then $\mathcal{C}$ sets $\mathbf{v} := \big(-\sum_{i\in[1+(j-1)(\ell+1)]} u_i d_i, 0, \ldots, 0\big) = (-2c, 0, \ldots, 0)$, where $c$ is the challenge of this query. Since $d_{1+(j-1)(\ell+1)+1} = 1$, it follows that $\sum_{i\in[1+j(\ell+1)]} u_i d_i = \sum_{i\in[1+(j-1)(\ell+1)]} u_i d_i + v_1 = 0$. As $\mathbf{d}$ is the only $\mathsf{PI}$ query on this token, Eq. (2) holds.

**Case 3:** Two queries use the $j$-th token, one in each execution. We denote them by $\mathbf{d}$ and $\tilde{\mathbf{d}}$. If $\mathbf{d} = \tilde{\mathbf{d}}$, then $\mathbf{v}$ is set as in Case 2. Otherwise, $\mathcal{C}$ chooses an index $\hat{k} \in [\ell]$ such that $b := d_{1+(j-1)(\ell+1)+1+\hat{k}}$ and $\tilde{b} := \tilde{d}_{1+(j-1)(\ell+1)+1+\hat{k}}$ satisfy $b \neq \tilde{b}$. If no such $\hat{k}$ exists, $\mathcal{C}$ aborts.

Let $c$ and $\tilde{c}$ be the challenges of the two queries. Since $\mathbf{d}$ and $\tilde{\mathbf{d}}$ are 0 on the blocks $1, \ldots, j-1$ and $u_1 = 2$, we have $\sum_{i\in[1+(j-1)(\ell+1)]} u_i d_i = 2c$ and $\sum_{i\in[1+(j-1)(\ell+1)]} u_i \tilde{d}_i = 2\tilde{c}$. Since $b, \tilde{b} \in \mathcal{S}_b$ and $b \neq \tilde{b}$, Lemma 1 gives a $\gamma \in \mathcal{R}$ with coefficients in $\{-1, 0, 1\}$ such that $\gamma(b - \tilde{b}) = 2 \pmod q$. The block $\mathbf{v}$ is then set as

$$v_{\hat{\jmath}} := \begin{cases} -2c + b\gamma(c - \tilde{c}) & \hat{\jmath} = 1, \\ -\gamma(c - \tilde{c}) & \hat{\jmath} = 1 + \hat{k}, \\ 0 & \text{otherwise.} \end{cases}$$

Then, modulo $q$

$$\sum_{i\in[1+j(\ell+1)]} u_i d_i = \sum_{i\in[1+(j-1)(\ell+1)]} u_i d_i + v_1 + b\,v_{1+\hat{k}} = 2c - 2c + (b - b)\gamma(c - \tilde{c}) = 0,$$
$$\sum_{i\in[1+j(\ell+1)]} u_i \widetilde{d}_i = \sum_{i\in[1+(j-1)(\ell+1)]} u_i \widetilde{d}_i + v_1 + \widetilde{b}\,v_{1+\hat{k}} = 2\tilde{c} - 2c + (b - \widetilde{b})\gamma(c - \tilde{c}) = 0,$$

where the last equality uses $(b - \widetilde{b})\gamma = 2 \pmod q$. Finally, since $\mathbf{d}$ and $\widetilde{\mathbf{d}}$ are the only $\mathsf{PI}$ queries with $d_{1+(j-1)(\ell+1)+[\ell+1]} \neq \mathbf{0}$ and $\widetilde{d}_{1+(j-1)(\ell+1)+[\ell+1]} \neq \mathbf{0}$, respectively, Eq. (2) holds.

**Analysis of $\mathcal{C}$.** Let $\mathsf{BadHash}$ be the event that the values $h_{2i-1}, h'_{2i-1}$ for $i \in [\mathsf{q}]$ are not pairwise distinct, or that the values $h_{2i}, h'_{2i}$ for $i \in [\mathsf{q}]$ are not pairwise distinct. We show that $\mathcal{C}$ wins the AOM-MISIS game if $\mathsf{Fork}^{\mathcal{B}}$ does not return $\perp$ and $\mathsf{BadHash}$ does not occur.

The only step at which $\mathcal{C}$ may abort is Case 3, when $\mathbf{d} \neq \tilde{\mathbf{d}}$ but $\mathbf{d}$ and $\tilde{\mathbf{d}}$ agree on all coefficients $b^{(1)}, \ldots, b^{(\ell)}$ of the $j$-th token. Let $\mathsf{inp}$ and $\mathsf{inp}'$ be the $\mathsf{H}_{\mathsf{non}}$ inputs $(W, \mathbf{m}, K)$ of the $\mathsf{SIGN}_2$ queries on the $j$-th token before and after rewinding, and let $J \in [\mathsf{q}]$ be such that $T_{\mathsf{non}}(\mathsf{inp}) = h_{2J-1}$. Since $\mathsf{BadHash}$ does not occur, the coefficients can only agree if $2J - 1 < i_{\mathsf{sig}}$ and $\mathsf{inp}' = \mathsf{inp}$, because both executions assign every slot before $i_{\mathsf{sig}}$ to the same input. Hence both executions compute the same $\tilde{\mathbf{w}}$ and the same $\mathsf{H}_{\mathsf{sig}}$ input $x := (\tilde{\mathbf{w}}, \mathbf{m}, K)$.

Let $2\hat{J}$ be the slot of $x$ before rewinding. Then $\hat{J} \leq J$, and since $i_{\mathsf{sig}}$ is even and larger than $2J-1$, we get $2\hat{J} \leq 2J \leq i_{\mathsf{sig}}$. If $2\hat{J} = i_{\mathsf{sig}}$, then $x = (\tilde{\mathbf{w}}^*, \mathbf{m}^*, K^*)$, so $(K^*, \mathbf{m}^*) \in Q_{\mathsf{euf}}$, which contradicts the assumption that $\mathcal{A}$ wins. Hence $2\hat{J} < i_{\mathsf{sig}}$, and both executions read the same challenge $h_{2\hat{J}}$. This implies $\mathbf{d} = \tilde{\mathbf{d}}$, a contradiction. Therefore $\mathcal{C}$ does not abort, and Eq. (2) holds for $j = q_s$ by the construction of $\mathbf{u}$.

**Norm bounds.** We have shown that $\mathbf{A}\hat{\mathbf{s}} = \sum_{i\in[Q]} \hat{b}_i \mathbf{p}_i \pmod q$, that $\hat{\mathbf{b}}^T \mathbf{u} \neq 0$, and that $\mathbf{d}^T \mathbf{u} = 0$ for every $\mathsf{PI}$ query $\mathbf{d}$. It remains to verify the norm conditions of the AOM-MISIS game, namely $\|\hat{\mathbf{s}}\| \leq \beta_{\mathsf{s}}$, $\left\|(\hat{b}_i \sigma_i)_{i\in[Q]}\right\|_1 \leq \beta_{\mathsf{b}}$ and $\left\|(u_i/\sigma_i)_{i\in[Q]}\right\| \leq \beta_{\mathsf{u}}$. Since both forgeries are valid, each of $\mathbf{z}^*$, $\mathbf{z}^{*\prime}$, $2^{\nu_{\mathsf{w}}}\mathbf{h}^* \bmod q$ and $2^{\nu_{\mathsf{w}}}\mathbf{h}^{*\prime} \bmod q$ has norm at most $\beta_{\mathsf{z}}$. We write $\mathbf{s}_{\mathsf{cs}} = (\mathbf{s}_{\mathsf{cs},[m-k]}, \mathbf{s}_{\mathsf{cs},(k)})$ according to $\mathbf{A} = [\hat{\mathbf{A}} \,|\, \mathbf{I}_k]$.

*Bound on* $\hat{\mathbf{s}}$. Let $\mathbf{t}_1 := \mathbf{p}_1$ and $\mathbf{t}_i := \mathbf{A}\, sk_i \bmod q$ for $i \geq 2$, and let $\mathbf{t}_{agg} := \sum_{i\in[n]} \mathbf{t}_i$. By construction of $\mathcal{B}$ and the KOSK key check, $vk_i = \lfloor \mathbf{t}_i \rceil_{\nu_{\mathsf{t}}}$ for every $i \in [n]$ and $\|sk_i\| \leq \beta_{\mathsf{sk}}$ for $i \geq 2$. In particular, $\left\|\mathbf{s}_{\mathsf{cs},[m-k]}\right\| + \left\|\mathbf{s}_{\mathsf{cs},(k)}\right\| \leq \sqrt{2}\,\|\mathbf{s}_{\mathsf{cs}}\| \leq \sqrt{2}(n-1)\beta_{\mathsf{sk}}$. Write $q_{\perp,\mathsf{t}}$ and $q_{\perp,\mathsf{w}}$ for $q_\perp$ with respect to $\nu_{\mathsf{t}}$ and $\nu_{\mathsf{w}}$, so that $q_{\perp,\mathsf{t}} \leq 2^{\nu_{\mathsf{t}}-1} - 1$ and $q_{\perp,\mathsf{w}} \leq 2^{\nu_{\mathsf{w}}-1} - 1$.

We first relate the rounded aggregated key $\tilde{\mathbf{t}}$ to $\mathbf{t}_{agg}$. For each $i \in [n]$, Lemma 3 gives $\mathbf{e}_i \in [-2^{\nu_{\mathsf{t}}-1}, 2^{\nu_{\mathsf{t}}-1} - 1]^{Nk}$ and $\mathbf{d}_i \in \{0,1\}^{Nk}$ with $2^{\nu_{\mathsf{t}}}\overline{vk_i} \equiv \mathbf{t}_i - \mathbf{e}_i + q_{\perp,\mathsf{t}}\mathbf{d}_i \pmod q$. Since $\tilde{\mathbf{t}} = \sum_i vk_i \bmod q_{\nu_{\mathsf{t}}}$, there exists $\boldsymbol{\kappa} \in \{0, \ldots, n-1\}^{Nk}$ such that $\bar{\tilde{\mathbf{t}}} - \sum_i \overline{vk_i} = -q_{\nu_{\mathsf{t}}}\boldsymbol{\kappa}$, and Corollary 1 gives

$$2^{\nu_{\mathsf{t}}}\bar{\tilde{\mathbf{t}}} \equiv \sum_{i\in[n]} 2^{\nu_{\mathsf{t}}}\overline{vk_i} + q_{\perp,\mathsf{t}}\boldsymbol{\kappa} \equiv \mathbf{t}_{agg} + \varepsilon_t \pmod q, \qquad \varepsilon_t := q_{\perp,\mathsf{t}}\Big(\boldsymbol{\kappa} + \sum_{i\in[n]} \mathbf{d}_i\Big) - \sum_{i\in[n]} \mathbf{e}_i.$$

As $\boldsymbol{\kappa} + \sum_i \mathbf{d}_i \in \{0, \ldots, 2n-1\}^{Nk}$, we get $\|\varepsilon_t\|_\infty \leq (2n-1)(2^{\nu_{\mathsf{t}}-1} - 1) + n\,2^{\nu_{\mathsf{t}}-1} \leq (3n-1)2^{\nu_{\mathsf{t}}-1}$.

By construction, $\hat{\mathbf{s}} = (\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2)$ with $\hat{\mathbf{s}}_1 = \Delta\mathbf{z} - \Delta c\,\mathbf{s}_{\mathsf{cs},[m-k]}$ and $\hat{\mathbf{s}}_2 = \Delta c\,\mathbf{t}_{agg} - \hat{A}\Delta\mathbf{z} - \Delta c\,\mathbf{s}_{\mathsf{cs},(k)}$, where $\Delta\mathbf{z} = \mathbf{z}^* - \mathbf{z}^{*\prime}$ and $\Delta c = c^* - c^{*\prime}$. Since $\|\Delta\mathbf{z}\| \leq 2\beta_{\mathsf{z}}$ and $\|\Delta c\|_1 \leq 2\beta_{\mathsf{c}}$, we have $\|\hat{\mathbf{s}}_1\| \leq 2\beta_{\mathsf{z}} + 2\beta_{\mathsf{c}}\left\|\mathbf{s}_{\mathsf{cs},[m-k]}\right\|$. For $\hat{\mathbf{s}}_2$, let

$$\mathbf{y} := \hat{A}\mathbf{z}^* - 2^{\nu_{\mathsf{t}}}c^*\bar{\tilde{\mathbf{t}}} \bmod q, \qquad \mathbf{y}' := \hat{A}\mathbf{z}^{*\prime} - 2^{\nu_{\mathsf{t}}}c^{*\prime}\bar{\tilde{\mathbf{t}}} \bmod q$$

be the values rounded during verification of the two forgeries. Then $\mathbf{y} - \mathbf{y}' = \hat{A}\Delta\mathbf{z} - \Delta c\,(\mathbf{t}_{agg} + \varepsilon_t) \pmod q$, and hence

$$\hat{\mathbf{s}}_2 = (\mathbf{y}' - \mathbf{y}) - \Delta c\,\varepsilon_t - \Delta c\,\mathbf{s}_{\mathsf{cs},(k)} \pmod q.$$

It remains to bound $\mathbf{y}' - \mathbf{y}$. Both forgeries recompute the same commitment $\tilde{\mathbf{w}}^* = \lfloor \mathbf{y} \rceil_{\nu_{\mathsf{w}}} + \mathbf{h}^* = \lfloor \mathbf{y}' \rceil_{\nu_{\mathsf{w}}} + \mathbf{h}^{*\prime} \bmod q_{\nu_{\mathsf{w}}}$, so there exists $\boldsymbol{\delta} \in \{-1, 0, 1\}^{Nk}$ with

$$\overline{\lfloor \mathbf{y}' \rceil_{\nu_{\mathsf{w}}}} - \overline{\lfloor \mathbf{y} \rceil_{\nu_{\mathsf{w}}}} - \overline{\mathbf{h}^*} + \overline{\mathbf{h}^{*\prime}} = q_{\nu_{\mathsf{w}}}\boldsymbol{\delta}.$$

Lemma 3, applied to $\mathbf{y}$ and $\mathbf{y}'$, gives $\mathbf{e}_{\mathbf{y}}, \mathbf{e}_{\mathbf{y}'} \in [-2^{\nu_{\mathsf{w}}-1}, 2^{\nu_{\mathsf{w}}-1} - 1]^{Nk}$ and $\mathbf{d}_{\mathbf{y}}, \mathbf{d}_{\mathbf{y}'} \in \{0,1\}^{Nk}$ with

$$2^{\nu_{\mathsf{w}}}\overline{\lfloor \mathbf{y} \rceil_{\nu_{\mathsf{w}}}} \equiv \mathbf{y} - \mathbf{e}_{\mathbf{y}} + q_{\perp,\mathsf{w}}\mathbf{d}_{\mathbf{y}}, \qquad 2^{\nu_{\mathsf{w}}}\overline{\lfloor \mathbf{y}' \rceil_{\nu_{\mathsf{w}}}} \equiv \mathbf{y}' - \mathbf{e}_{\mathbf{y}'} + q_{\perp,\mathsf{w}}\mathbf{d}_{\mathbf{y}'} \pmod q.$$

Combining these with Corollary 1 yields

$$\mathbf{y}' - \mathbf{y} \equiv 2^{\nu_{\mathsf{w}}}\big(\overline{\mathbf{h}^*} - \overline{\mathbf{h}^{*\prime}}\big) + \mathbf{e}_{\mathbf{y}'} - \mathbf{e}_{\mathbf{y}} - q_{\perp,\mathsf{w}}\big(\boldsymbol{\delta} + \mathbf{d}_{\mathbf{y}'} - \mathbf{d}_{\mathbf{y}}\big) \pmod q.$$

The coefficients of $\mathbf{e}_{\mathbf{y}'} - \mathbf{e}_{\mathbf{y}}$ lie in $[-(2^{\nu_{\mathsf{w}}} - 1), 2^{\nu_{\mathsf{w}}} - 1]$ and those of $\boldsymbol{\delta} + \mathbf{d}_{\mathbf{y}'} - \mathbf{d}_{\mathbf{y}}$ in $\{-2, \ldots, 2\}$, so the last two terms have $\infty$-norm at most $2^{\nu_{\mathsf{w}}} - 1 + 2q_{\perp,\mathsf{w}} \leq 2^{\nu_{\mathsf{w}}+1}$. In verification, the scaled hint

$2^{\nu_w}\overline{\mathbf{h}}$ is reduced modulo $q$ before its norm is taken. Since both forgeries pass the norm check, we have $\left\|2^{\nu_w}\overline{\mathbf{h}^*} \bmod q\right\|, \left\|2^{\nu_w}\overline{\mathbf{h}^{*\prime}} \bmod q\right\| \le \beta_z$, and therefore

$$\|\mathbf{y}' - \mathbf{y}\| \le 2\beta_z + 2^{\nu_w+1}\sqrt{Nk}.$$

Finally, $\|\Delta c\, \varepsilon_t\| \le \sqrt{Nk}\, \|\Delta c\|_1 \|\varepsilon_t\|_\infty \le (3n-1)\beta_c 2^{\nu_t}\sqrt{Nk}$ and $\left\|\Delta c\, \mathbf{s}_{cs,(k)}\right\| \le 2\beta_c \left\|\mathbf{s}_{cs,(k)}\right\|$. Combining all bounds,

$$\|\hat{\mathbf{s}}\| \le \|\hat{\mathbf{s}}_1\| + \|\hat{\mathbf{s}}_2\| \le 4\beta_z + \left(2^{\nu_w+1} + (3n-1)\beta_c 2^{\nu_t}\right)\sqrt{Nk} + 2\sqrt{2}(n-1)\beta_c\beta_{sk} = \beta_s.$$

*Bound on* $\hat{\mathbf{b}}$. Since $\hat{\mathbf{b}} = (\Delta c, 0, \ldots, 0)$ and $\sigma_1 = \sigma_{sk}$, we get $\left\|(\hat{b}_i\sigma_i)_{i\in[Q]}\right\|_1 = \sigma_{sk}\|\Delta c\|_1 \le 2\beta_c\sigma_{sk} = \beta_b$.

*Bound on* $\mathbf{u}$. We have $u_1 = 2$, and we bound the block $\mathbf{v}^{(j)}$ of the $j$-th token in each case. In Case 1, $\mathbf{v}^{(j)} = \mathbf{0}$. In Case 2, its only non-zero entry is $-2c$, so $\left\|\mathbf{v}^{(j)}\right\| = 2\sqrt{\beta_c}$. In Case 3, its only non-zero entries are $v_1 = \gamma(\tilde{b}c - b\tilde{c})$, using $\gamma(b - \tilde{b}) = 2$, and $v_{1+\hat{k}} = -\gamma(c - \tilde{c})$. Since $\gamma$ has coefficients in $\{-1, 0, 1\}$, we have $\|\gamma\| \le \sqrt{N}$, and both entries have norm at most $2\beta_c\sqrt{N}$. Hence $\left\|\mathbf{v}^{(j)}\right\|^2 \le 8N\beta_c^2$ in every case. Since $\sigma_i = \sigma_w$ for all nonce entries,

$$\left\|(u_i/\sigma_i)_{i\in[Q]}\right\| \le \frac{2}{\sigma_{sk}} + \frac{1}{\sigma_w}\Big(\sum_{j\in[q_s]}\left\|\mathbf{v}^{(j)}\right\|^2\Big)^{1/2} \le \frac{2}{\sigma_{sk}} + \frac{\beta_c\sqrt{8Nq_s}}{\sigma_w} = \beta_u.$$

Finally, since each of $h_1, h_3, \ldots, h_{2q-1}$ and $h_2, h_4, \ldots, h_{2q}$ are sampled uniformly from $\mathcal{S}_b^\ell$ and $\mathcal{S}_c$, respectively,

$$\Pr[\mathsf{BadHash}] \le \frac{(2q)^2}{|\mathcal{S}_b|^\ell} + \frac{(2q)^2}{|\mathcal{S}_c|} \le 8q^2 2^{-2\lambda}.$$

Therefore,

$$\mathsf{Adv}^{\mathsf{AOM\text{-}MISIS}}_{\mathsf{par}}(\mathcal{C}, \lambda) \ge \mathsf{acc}(\mathsf{Fork}^{\mathcal{B}}) - \Pr[\mathsf{BadHash}] \ge \frac{\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{MRaccoon},\mathcal{A}}(\lambda)^2}{q} - 8q^2 2^{-2\lambda},$$

which concludes the theorem. □

**Theorem 3 (SUF-CMA security of MRaccoon).** *Let $\mathcal{A}$ be an adversary against the SUF-CMA security of* MRaccoon *in the KOSK model that makes at most $q_s$ signing queries and at most $q_h$ queries to RO. Then there exist an algorithm $\mathcal{C}$ against AOM-MISIS and an algorithm $\mathcal{B}_{\mathsf{msis}}$ against* $\mathsf{MSIS}_{\mathsf{par}}$ *with par $:= (q, k, m, \beta_{\mathsf{msis}})$ and $\beta_{\mathsf{msis}} := 4\beta_z + 2^{\nu_w+2}\sqrt{kN}$, where $2\beta_z < q/2$, such that the bound of Theorem 2 holds with $\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{MRaccoon},\mathcal{A}}(\lambda)$ replaced by*

$$\mathsf{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{MRaccoon},\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathsf{MSIS}}_{\mathsf{par}}(\mathcal{B}_{\mathsf{msis}}, \lambda).$$

*Proof.* The proof follows that of Theorem 2. The algorithms $\mathcal{B}$ and $\mathcal{C}$, the forking step and the construction of the AOM-MISIS solution are reused, so we only describe the differences, which concern the forgery processed by $\mathcal{C}$.

$\mathcal{A}$ now returns $(K^*, \mathbf{m}^*, (\sigma_1^*, \ldots, \sigma_f^*), (sk_2, \ldots, sk_n))$ with pairwise distinct $\sigma_l^* = (c_l^*, \mathbf{z}_l^*, \mathbf{h}_l^*)$ and $f > Q_{\mathsf{ctr}}[(K^*, \mathbf{m}^*)]$. $\mathcal{B}$ computes $\tilde{\mathbf{w}}_l^* := \lfloor \hat{\mathbf{A}}\mathbf{z}_l^* - 2^{\nu_t} c_l^* \overline{\tilde{\mathbf{t}}^*} \rceil_{\nu_w} + \mathbf{h}_l^*$ for every $l \in [f]$, without any random-oracle query. It returns $\perp$ if two distinct signatures share the same value $\tilde{\mathbf{w}}^*$. Otherwise, it picks the smallest index $l$ with $(K^*, \mathbf{m}^*, \tilde{\mathbf{w}}_l^*) \notin Q_{\mathsf{suf}}$, sets $\sigma^* := \sigma_l^*$ and continues as in the EUF-CMA case, with $i_{\mathsf{sig}} := T_{\mathsf{isig}}(\tilde{\mathbf{w}}_l^*, \mathbf{m}^*, K^*)$. It verifies only $\sigma_l^*$, using one query to $\mathsf{H}_{\mathsf{sig}}$. Since the $\mathsf{H}_{\mathsf{sig}}$ queries of the signing oracle never receive a new index, the bound $q = q_h + q_s + 1$ still holds.

In the EUF-CMA proof, freshness of the forged message guarantees that the query at index $i_{\mathsf{sig}}$ was not programmed by the signing oracle. Here, the condition $(K^*, \mathbf{m}^*, \tilde{\mathbf{w}}_l^*) \notin Q_{\mathsf{suf}}$ provides the same guarantee, even though $\mathbf{m}^*$ may have been signed before. The remainder of the proof applies verbatim to $\sigma_l^*$.

**Construction of $\mathcal{B}$.** $\mathcal{B}$ simulates the SUF-CMA game exactly as before, with one addition: whenever a session is closed, it appends $(K, \mathbf{m}, \tilde{\mathbf{w}})$ to the list $Q_{\mathsf{suf}}$, where $\tilde{\mathbf{w}}$ is the aggregated nonce of that session. Upon receiving $\big(K^*, \mathbf{m}^*, (\sigma_1^*, \ldots, \sigma_f^*), (sk_2, \ldots, sk_n)\big)$ with $\sigma_j^* = (c_j^*, \mathbf{z}_j^*, \mathbf{h}_j^*)$, $\mathcal{B}$ computes

$\tilde{\mathbf{w}}_j^* := \lfloor \hat{\mathbf{A}}\mathbf{z}_j^* - 2^{\nu_t} c_j^* \overline{\tilde{\mathbf{t}}^*} \rceil_{\nu_w} + \mathbf{h}_j^*$ for all $j \in [f]$. It returns $\perp$ if two distinct signatures share the same $\tilde{\mathbf{w}}^*$. Otherwise, it lets $j$ be the smallest index with $(K^*, \mathbf{m}^*, \tilde{\mathbf{w}}_j^*) \notin Q_{\mathsf{suf}}$ and continues with $\sigma^* := \sigma_j^*$ exactly as in the EUF-CMA case. Taking the smallest index keeps the output of $\mathcal{B}$ a deterministic function of its coins and hash values, as required by Lemma 9.

**Analysis of $\mathcal{B}$.** Suppose that $\mathcal{A}$ wins. Let $\mathsf{Reuse}$ be the event that two distinct output signatures $\sigma_a^* \neq \sigma_b^*$ satisfy $\tilde{\mathbf{w}}_a^* = \tilde{\mathbf{w}}_b^*$. Their $\mathsf{H}_{\mathsf{sig}}$ inputs then coincide, so $c_a^* = c_b^* =: c$. If also $\mathbf{z}_a^* = \mathbf{z}_b^*$, then $\mathbf{h}_a^* = \mathbf{h}_b^*$ and the two signatures would be equal. Hence $\Delta\mathbf{z} := \mathbf{z}_a^* - \mathbf{z}_b^* \neq \mathbf{0}$.

For $\iota \in \{a, b\}$, let $\mathbf{y}_\iota := \hat{\mathbf{A}}\mathbf{z}_\iota^* - 2^{\nu_t} c \overline{\tilde{\mathbf{t}}^*}$, so that $\Delta\mathbf{y} := \mathbf{y}_a - \mathbf{y}_b = \hat{\mathbf{A}}\Delta\mathbf{z}$. Since both signatures reconstruct the same $\tilde{\mathbf{w}}^*$, taking canonical lifts gives a $\boldsymbol{\delta} \in \{-1, 0, 1\}^{Nk}$ with $\overline{\lfloor \mathbf{y}_a \rceil_{\nu_w}} - \overline{\lfloor \mathbf{y}_b \rceil_{\nu_w}} - \overline{\mathbf{h}_b^*} + \overline{\mathbf{h}_a^*} = q_{\nu_w}\boldsymbol{\delta}$. Lemma 3 gives $\mathbf{e}_\iota \in [-2^{\nu_w - 1}, 2^{\nu_w - 1} - 1]^{Nk}$ and $\mathbf{d}_\iota \in \{0, 1\}^{Nk}$ with $2^{\nu_w} \overline{\lfloor \mathbf{y}_\iota \rceil_{\nu_w}} \equiv \mathbf{y}_\iota - \mathbf{e}_\iota + q_{\perp,\mathsf{w}}\mathbf{d}_\iota \pmod q$, and Corollary 1 then yields

$$\hat{\mathbf{A}}\Delta\mathbf{z} = \Delta\mathbf{y} \equiv 2^{\nu_w}\big(\overline{\mathbf{h}_b^*} - \overline{\mathbf{h}_a^*}\big) + \mathbf{e}_a - \mathbf{e}_b - q_{\perp,\mathsf{w}}\big(\boldsymbol{\delta} + \mathbf{d}_a - \mathbf{d}_b\big) \pmod q.$$

Hence $(\Delta\mathbf{z}, -\Delta\mathbf{y})$ is a non-zero vector in the kernel of $[\hat{\mathbf{A}} \mid \mathbf{I}_k]$. The verification bounds on $\mathbf{z}_\iota^*$ and on $2^{\nu_w}\overline{\mathbf{h}_\iota^*} \bmod q$ give $\|\Delta\mathbf{z}\| \leq 2\beta_{\mathsf{z}}$ and $\|2^{\nu_w}(\overline{\mathbf{h}_b^*} - \overline{\mathbf{h}_a^*}) \bmod q\| \leq 2\beta_{\mathsf{z}}$. The coefficients of $\mathbf{e}_a - \mathbf{e}_b$ lie in $[-(2^{\nu_w} - 1), 2^{\nu_w} - 1]$ and those of $\boldsymbol{\delta} + \mathbf{d}_a - \mathbf{d}_b$ in $\{-2, \ldots, 2\}$, so the remaining error has $\infty$-norm at most $2^{\nu_w} - 1 + 2q_{\perp,\mathsf{w}} \leq 2^{\nu_w + 1}$, hence $\ell_2$-norm at most $2^{\nu_w + 1}\sqrt{Nk}$. Altogether,

$$\|(\Delta\mathbf{z}, -\Delta\mathbf{y})\| \leq 4\beta_{\mathsf{z}} + 2^{\nu_w + 1}\sqrt{Nk} = \beta_{\mathsf{msis}},$$

and the vector is non-zero modulo $q$ since $2\beta_{\mathsf{z}} < q/2$.

The event $\mathsf{Reuse}$ depends only on the view of $\mathcal{A}$, which $\mathcal{B}$ simulates perfectly, so we may bound it in the real game. The algorithm $\mathcal{B}_{\mathsf{msis}}$ embeds its instance as $\hat{\mathbf{A}}$, runs the SUF-CMA game for $\mathcal{A}$ honestly with a freshly generated key for the honest signer, and outputs $(\Delta\mathbf{z}, -\Delta\mathbf{y})$. Thus $\Pr[\mathsf{Reuse}] \leq \mathsf{Adv}_{par}^{\mathsf{MSIS}}(\mathcal{B}_{\mathsf{msis}}, \lambda)$.

If $\mathsf{Reuse}$ does not occur, the values $\tilde{\mathbf{w}}_1^*, \ldots, \tilde{\mathbf{w}}_f^*$ are pairwise distinct. Each closed session on $(K^*, \mathbf{m}^*)$ appends exactly one entry to $Q_{\mathsf{suf}}$, and fewer than $f$ such sessions were closed. So the index $l$ exists, and $\mathcal{B}$ does not return $\perp$. Consequently,

$$\mathsf{acc}(\mathcal{B}) \geq \mathsf{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{MRaccoon},\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathsf{MSIS}}_{par}(\mathcal{B}_{\mathsf{msis}}, \lambda),$$

and Lemma 9 gives $\mathsf{acc}(\mathsf{Fork}^{\mathcal{B}}) \geq \mathsf{acc}(\mathcal{B})^2/\mathsf{q}$.

**Construction of $\mathcal{C}$ and the AOM-MISIS solution.** These are unchanged. The extraction uses only three facts: both executions output valid signatures on the same $\mathsf{H}_{\mathsf{sig}}$ input $x_{i_{\mathsf{sig}}} = (\tilde{\mathbf{w}}_l^*, \mathbf{m}^*, K^*)$, the challenges satisfy $c^* \neq c^{*\prime}$, and the corrupted secret keys are known. All three hold for the selected signature.

**Analysis of $\mathcal{C}$.** The EUF-CMA winning condition is used in exactly one place: to show that, in Case 3, the slot $2\hat{J}$ of the token's $\mathsf{H}_{\mathsf{sig}}$ input is not $i_{\mathsf{sig}}$. We replace that step as follows. Suppose that $2\hat{J} = i_{\mathsf{sig}}$. Then $x_{i_{\mathsf{sig}}} = (\tilde{\mathbf{w}}, \mathbf{m}, K)$ is the $\mathsf{H}_{\mathsf{sig}}$ input of the session of the $j$-th token. That session was closed in the first execution, so it appended $(K, \mathbf{m}, \tilde{\mathbf{w}}) = (K^*, \mathbf{m}^*, \tilde{\mathbf{w}}_l^*)$ to $Q_{\mathsf{suf}}$, which contradicts the choice of $l$. Hence $2\hat{J} < i_{\mathsf{sig}}$, and the rest of the analysis applies unchanged.

Combining these steps with the remainder of the proof of Theorem 2 yields the claim. □

## 5 Intermediate scheme MRaccoon⋆

We introduce the intermediate scheme $\mathsf{MRaccoon}^\star$ to capture the shift that blindness adds to our blind multi-signature construction. It extends $\mathsf{MRaccoon}$ (Section 4) with an auxiliary signer $\mathcal{S}_0$ whose secret and verification keys are both fixed to $\mathbf{0}$. Figure 5 presents both schemes, and its boxed components are the additions of $\mathsf{MRaccoon}^\star$.

The auxiliary signer works as follows.

- $\mathcal{S}_0$ opens each signing session with an additional round, $\mathsf{Sign}_0$. It samples $\mathbf{r}'$ from the wider Gaussian $\mathcal{D}^m_{\sigma_{w'}}$ and broadcasts the nonce commitment $\mathbf{w}_0 = \mathbf{A}\mathbf{r}'$.

– Each honest signer fixes $\mathbf{w}_0$ before it samples its own nonce commitments. The commitment $\mathbf{w}_0$ is added to the aggregated nonce, but it is not part of the hash input that determines $\mathbf{b}$, because hashing it would break blindness in BMRaccoon. The round order binds it to the session instead.
– At aggregation, $\mathcal{S}_0$ contributes the partial signature $\mathbf{z}_0 = \mathbf{r}'_{[m-k]}$.

The final signature therefore carries the additional shift $\mathbf{r}'$, which we accommodate with a larger verification bound $\beta'_{\mathsf{z}}$. Let $\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]$ denote MRaccoon with verification bound $\beta'_{\mathsf{z}}$. Then

$$\mathsf{MRaccoon}^{\star}.\mathsf{Verify}(K, \mathbf{m}, \sigma) = \mathsf{MRaccoon}[\beta'_{\mathsf{z}}].\mathsf{Verify}\big((\mathbf{0}, K), \mathbf{m}, \sigma\big), \tag{3}$$

that is, $\mathsf{MRaccoon}^{\star}$ accepts a signature for $K$ exactly when $\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]$ accepts it for the extended key set $\hat{K} := (\mathbf{0}, K)$.

### 5.1 Security of MRaccoon⋆

**Theorem 4 (Correctness of $\mathsf{MRaccoon}^{\star}$).** *Let $\beta_{\mathsf{sk}}$ and $\beta_{\mathsf{w}}$ be as in Theorem 1, and let $\beta_{\mathsf{w}'} := e^{1/4}\sigma_{\mathsf{w}'}\sqrt{Nm}$. If*

$$\beta'_{\mathsf{z}} \geq \sqrt{2}\left(n\beta_{\mathsf{c}}\beta_{\mathsf{sk}} + n(1+\ell)\beta_{\mathsf{w}} + \beta_{\mathsf{w}'}\right) + \left((3n-1)\beta_{\mathsf{c}}\, 2^{\nu_{\mathsf{t}}-1} + 4\cdot 2^{\nu_{\mathsf{w}}-1}\right)\sqrt{Nk},$$

*then $\mathsf{MRaccoon}^{\star}$ with $n$ signers and the auxiliary signer $S_0$ is correct, and an honestly generated signature is rejected with probability at most $(n(\ell+2)+1)\, 2^{-Nm/10}$.*

*Proof.* The proof follows that of Theorem 1 with $\mathbf{r}_{agg}$ replaced by $\mathbf{r}^{\star} := \mathbf{r}_{agg} + \mathbf{r}'$, where $\mathbf{r}' \leftarrow D^{m}_{\sigma_{\mathsf{w}'}}$ is the nonce of the auxiliary signer $S_0$. Since every signer adds $\mathbf{w}_0 = \mathbf{A}\mathbf{r}'$ to $\mathbf{w}^{(0)}$ and $\mathsf{SignAgg}$ adds $\mathbf{z}_0 = \mathbf{r}'_{[m-k]}$, the aggregated nonce is $\mathbf{w}' = \mathbf{A}\mathbf{r}^{\star} \bmod q$ and the response is $\mathbf{z} = c\,\mathbf{s}_{agg,[m-k]} + \mathbf{r}^{\star}_{[m-k]} \bmod q$. All parties hash with the same $(0, K)$ and therefore agree on $(b^{(j)})_j$, $\tilde{\mathbf{w}}$ and $c$. As $(sk_0, vk_0) = (\mathbf{0}, \mathbf{0})$, $S_0$ contributes nothing to $\mathbf{s}_{agg}$ or $\tilde{\mathbf{t}}$, and its rounding error is $\mathbf{0}$, so $\varepsilon_t$ and its bound $(3n-1)2^{\nu_{\mathsf{t}}-1}$ are unchanged. The previous proof uses $\mathbf{r}_{agg}$ only through $\mathbf{w}'$ and $\mathbf{z}$, so it carries over verbatim. The verifier recovers $\tilde{\mathbf{w}}$, hence $c' = c$, and $\|(\mathbf{z}, 2^{\nu_{\mathsf{w}}}\mathbf{h})\| \leq \sqrt{2}\beta_{\mathsf{c}}\|\mathbf{s}_{agg}\| + \sqrt{2}\|\mathbf{r}^{\star}\| + \big((3n-1)\beta_{\mathsf{c}}2^{\nu_{\mathsf{t}}-1} + 4\cdot 2^{\nu_{\mathsf{w}}-1}\big)\sqrt{Nk}$. A union bound over the $n$ keys, the $n(\ell+1)$ nonces and $\mathbf{r}'$ gives $\|\mathbf{r}^{\star}\| \leq n(1+\ell)\beta_{\mathsf{w}} + \beta_{\mathsf{w}'}$ with $\beta_{\mathsf{w}'} := e^{1/4}\sigma_{\mathsf{w}'}\sqrt{Nm}$, except with probability $(n(\ell+2)+1)2^{-Nm/10}$, so the norm is at most $\beta'_{\mathsf{z}}$. □

**Theorem 5 (SUF of $\mathsf{MRaccoon}^{\star}$).** *Let $\beta'_{\mathsf{z}}$ be the verification bound of $\mathsf{MRaccoon}^{\star}$, and let $\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]$ denote MRaccoon with verification bound $\beta'_{\mathsf{z}}$. Let $\mathcal{A}$ be an adversary against the SUF-CMA security of $\mathsf{MRaccoon}^{\star}$ in the KOSK model and the random oracle model. Suppose $\mathcal{A}$ opens at most $q_s$ signing sessions and makes at most $q_h$ queries to $\mathsf{H}_{\mathsf{non}}$ and $\mathsf{H}_{\mathsf{sig}}$. Then there is an adversary $\mathcal{R}'$ against the SUF-CMA security of $\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]$, with running time about that of $\mathcal{A}$. $\mathcal{R}'$ opens at most $q_s$ sessions, makes at most $q_h$ queries to the random oracles, and registers one additional key. Moreover,*

$$\mathsf{Adv}^{\mathsf{SUF}}_{\mathsf{MRaccoon}^{\star},\mathcal{A}}(\lambda) \;\leq\; \mathsf{Adv}^{\mathsf{SUF}}_{\mathsf{MRaccoon}[\beta'_{\mathsf{z}}],\mathcal{R}'}(\lambda) + \frac{q_s\,(q_h+q_s)+1}{2^{\gamma}},$$

*where $\gamma$ is a lower bound on the min-entropy of an honest commitment $com_{1,\mathsf{sid}}$ (Lemma 10).*

*Proof.* **The reduction $\mathcal{R}'$.** $\mathcal{R}'$ plays the SUF-CMA game of $\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]$ with honest signer $vk_1$, and runs $\mathcal{A}$ on $(par, vk_1)$. Besides the signers of $\mathcal{A}$, it treats the auxiliary signer as one more corrupted co-signer, with key pair $(sk_0, vk_0) = (\mathbf{0}, \mathbf{0})$. This is a valid key pair in the KOSK model, since $\lfloor \mathbf{A0} \rceil_{\nu_{\mathsf{t}}} = \mathbf{0}$.

We say that a hash input $(W, \mathbf{m}, \hat{K})$ *belongs to session sid* if $\hat{K} = (\mathbf{0}, K)$ and the entry of $W$ at the first position of $vk_1$ in $K$ is $com_{1,sid}$. $\mathcal{R}'$ answers the queries that $\mathcal{A}$ makes in the game of $\mathsf{MRaccoon}^{\star}$ as follows.

– *Challenge hash.* On a query $\mathsf{H}_{\mathsf{sig}}(x)$, $\mathcal{R}'$ queries $\mathsf{MRaccoon}.\mathsf{H}_{\mathsf{sig}}(x)$ and returns the answer unchanged. Both schemes evaluate the challenge hash on the same inputs, so no translation is needed.

– *Nonce hash.* On a query $\mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \hat{K})$, $\mathcal{R}'$ returns the stored answer if the query was made before. Otherwise, it computes

$$\phi(W, \mathbf{m}, \hat{K}) := \begin{cases} \big((com_{0,sid}, W), \mathbf{m}, \hat{K}\big) & \text{if the input belongs to session } sid, \\ \big((\mathbf{0}^{\ell+1}, W), \mathbf{m}, \hat{K}\big) & \text{if it belongs to no session,} \end{cases}$$

where $com_{0,sid} = (\mathbf{w}_{0,sid}, \mathbf{0}, \ldots, \mathbf{0})$ is the nonce list of the auxiliary signer in session $sid$. It then queries $\mathsf{MRaccoon.H}_{\mathsf{non}}$ on the result, stores the answer and returns it.

– *Oracle* $\mathsf{SIGN}_0$. This algorithm belongs to the auxiliary signer, whose key pair is public. $\mathcal{A}$ therefore runs it itself, and $\mathcal{R}'$ does not simulate it. $\mathcal{A}$ sends a shift nonce $\mathbf{w}_0$ to the honest signer, and this nonce may be chosen maliciously. The shift is not an input to $\mathsf{H}_{\mathsf{non}}$ and enters only the aggregated nonce $\tilde{\mathbf{w}}$. This is sound because $\mathbf{w}_0$ is fixed in $\mathsf{SIGN}_1$, before the honest nonces of the session are sampled.

– *Oracle* $\mathsf{SIGN}_1(\mathbf{w}_0)$. $\mathcal{R}'$ queries $\mathsf{MRaccoon.SIGN}_1$ and obtains $(sid, com_{1,sid})$. It stores $aux_i := \mathbf{w}_{0,sid}$ as part of the state of session $sid$, and only then returns $(sid, com_{1,sid})$ to $\mathcal{A}$.

– *Oracle* $\mathsf{SIGN}_2(sid, K, W, \mathbf{m})$. $\mathcal{R}'$ retrieves $\mathbf{w}_{0,sid}$ from $aux_i$ and sets $\hat{K} := (\mathbf{0}, K)$ and

$$com_{0,sid} := (\mathbf{w}_{0,sid}, \mathbf{0}, \ldots, \mathbf{0}), \qquad \hat{W} := (com_{0,sid}, W).$$

It queries $\mathsf{MRaccoon.SIGN}_2(sid, \hat{K}, \hat{W}, \mathbf{m})$ and returns the answer $(\tilde{\mathbf{w}}, \mathbf{z}_1)$. The auxiliary signer thus appears in the $\mathsf{MRaccoon}$ session as an ordinary co-signer with nonce list $com_{0,sid}$.

– *Output.* When $\mathcal{A}$ outputs signatures $\{\sigma_j^*\}_{j\in[l]}$ on $(K^*, \mathbf{m}^*)$ with secret keys $sk_2^*, \ldots sk_n^*$, $\mathcal{R}'$ outputs the same signatures on $(\hat{K}^*, \mathbf{m}^*)$ with secret keys $(\mathbf{0}, sk_2^*, \ldots sk_n^*)$, where $\hat{K}^* := (\mathbf{0}, K^*)$.

We now show that $\mathcal{R}'$ perfectly simulates the view of $\mathcal{A}$, unless a bad event occurs, and that every forgery of $\mathcal{A}$ gives a valid forgery for $\mathcal{R}'$ using the following lemma.

**Lemma 10 (Min-entropy of honest commitments [DPKM+24, Lemma 3.8]).** *Let $q, k, m$ be positive integers with $k < m$ and let $\sigma > 0$. For $\hat{\mathbf{A}} \in \mathcal{R}_q^{k\times(m-k)}$, let $\mathcal{W}_\sigma(\hat{\mathbf{A}})$ be the distribution of $[\hat{\mathbf{A}} \mid \mathbf{I}_k]\,\mathbf{s} \bmod q$ for $\mathbf{s} \leftarrow \mathcal{D}_\sigma^m$, and let*

$$H_\infty\big(\mathcal{W}_\sigma(\hat{\mathbf{A}})\big) := -\log_2 \max_{\mathbf{x}'\in\mathcal{R}_q^k} \Pr_{\mathbf{x}\leftarrow\mathcal{W}_\sigma(\hat{\mathbf{A}})}[\mathbf{x} = \mathbf{x}']$$

*denote its min-entropy. If $\sigma > \sqrt{2\pi}\cdot 2N \cdot q^{\frac{1}{m}+\frac{2}{N(m-k)}}$, then*

$$\Pr_{\hat{\mathbf{A}}\xleftarrow{\$}\mathcal{R}_q^{k\times(m-k)}}\Big[H_\infty\big(\mathcal{W}_\sigma(\hat{\mathbf{A}})\big) \geq N-1\Big] \geq 1 - 2^{-N+1}.$$

*In particular, if $\sigma_{\mathsf{w}}$ satisfies this condition, then except with probability $2^{-N+1}$ over the choice of $\hat{\mathbf{A}}$, an honest nonce commitment $\mathbf{w}_1^{(0)} = \mathbf{A}\mathbf{r}_1^{(0)}$ with $\mathbf{r}_1^{(0)} \leftarrow \mathcal{D}_{\sigma_{\mathsf{w}}}^m$, and hence $com_{1,sid}$, has min-entropy at least $\gamma := N - 1$.*

*Simulation of* $\mathsf{H}_{\mathsf{non}}$. Let $\mathsf{BadA}$ be the event that $H_\infty(\mathcal{W}_{\sigma_{\mathsf{w}}}(\hat{\mathbf{A}})) < \gamma$. By Lemma 10, $\Pr[\mathsf{BadA}] \leq 2^{-N+1}$. In the rest of the proof we condition on $\neg\mathsf{BadA}$. Let $\mathsf{Coll}$ be the event that two sessions output the same commitment $com_{1,sid}$. Let $\mathsf{Guess}$ be the event that $\mathcal{A}$ queries $\mathsf{H}_{\mathsf{non}}$ on an input that belongs to a session before $\mathsf{SIGN}_1$ returns the commitment of that session. By the min-entropy bound $\gamma$ on honest commitments,

$$\Pr[\mathsf{Coll} \vee \mathsf{Guess}] \leq \frac{q_s(q_h + q_s)}{2^\gamma}.$$

If neither event occurs, each query belongs to at most one session, and the corresponding shift $\mathbf{w}_{0,sid}$ is already stored when the query is made. So $\phi$ is well defined at query time. Moreover, $\phi$ is injective, since its output preserves $W$, $\mathbf{m}$ and $\hat{K}$, from which the input can be recovered. Let $\mathsf{H} := \mathsf{MRaccoon.H}_{\mathsf{non}}$. The oracle that $\mathcal{A}$ sees is $\mathsf{H} \circ \phi$. Since $\mathsf{H}$ is a random oracle, $\phi$ is injective, and $\mathcal{R}'$ queries $\mathsf{H}$ only through $\phi$, this composition is a uniformly random function, exactly as in the real game.

The simulation is also consistent with the honest signer. In session $sid$, the oracle $\mathsf{MRaccoon.SIGN}_2$ derives its coefficients from $\mathsf{H}(\hat{W}, \mathbf{m}, \hat{K})$. Since $\hat{W} = (com_{0,sid}, W)$, this is $\mathsf{H}(\phi(W, \mathbf{m}, \hat{K}))$, the value that $\mathcal{A}$ receives on the query $\mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \hat{K})$.

*Honest responses.* Consider a session $sid$ with honest nonces $(\mathbf{r}_1^{(j)})_{j\in[0,\ell]}$ and stored shift $\mathbf{w}_{0,sid}$. Write $com_{0,sid} = (\mathbf{w}_{0,sid}^{(0)}, \ldots, \mathbf{w}_{0,sid}^{(\ell)})$, where $\mathbf{w}_{0,sid}^{(0)} = \mathbf{w}_{0,sid}$ and $\mathbf{w}_{0,sid}^{(j)} = \mathbf{0}$ for $j \in [\ell]$. On input $(sid, \hat{K}, \hat{W}, \mathbf{m})$, the honest signer of $\mathsf{MRaccoon}$ computes

$$\big(b^{(j)}\big)_{j\in[\ell]} = \mathsf{H}(\hat{W}, \mathbf{m}, \hat{K}) = \mathsf{H}\big(\phi(W, \mathbf{m}, \hat{K})\big), \quad \mathbf{w}^{(j)} = \mathbf{w}_{0,sid}^{(j)} + \sum_{i=1}^{n} \mathbf{w}_i^{(j)} \quad (j \in [0, \ell]).$$

The first identity holds because $(W, \mathbf{m}, \hat{K})$ belongs to session $sid$, the entry of $W$ at the position of $vk_1$ is $com_{1,sid}$. The coefficients are therefore exactly those that $\mathcal{A}$ obtains from $\mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \hat{K})$. The second identity holds because $com_{0,sid}$ is the nonce list of the auxiliary signer. Both values coincide with those computed by the boxed $\mathsf{Sign}_2$ of $\mathsf{MRaccoon}^\star$. Consequently, $\tilde{\mathbf{w}}$ coincides, and so does

$$c = \mathsf{MRaccoon}.\mathsf{H}_{\mathsf{sig}}(\tilde{\mathbf{w}}, \mathbf{m}, \hat{K}) = \mathsf{H}_{\mathsf{sig}}(\tilde{\mathbf{w}}, \mathbf{m}, \hat{K}).$$

Finally, $\mathbf{z}_1 = c\,\mathbf{s}_{1,[m-k]} + \mathbf{r}_{1,[m-k]}^{(0)} + \sum_{j=1}^{\ell} b^{(j)} \mathbf{r}_{1,[m-k]}^{(j)}$ also coincides, since $\mathbf{s}_1 = sk_1$ and the nonces are those of session $sid$.

The $\mathsf{MRaccoon}$ oracle accepts the query as it places no condition on co-signer nonces, and it checks the position of the honest key and of the honest nonce exactly as $\mathsf{MRaccoon}^\star$ does. The commitments returned by $\mathsf{SIGN}_1$ are identically distributed in both schemes. The auxiliary signer holds no secret and is run by $\mathcal{A}$, so it need not be simulated. Therefore, every answer that $\mathcal{R}'$ gives to $\mathcal{A}$ is distributed exactly as in the real game.

*Forgeries.* By Eq. (3), the two verification algorithms agree: they compute the same aggregated key, since the extra entry $vk_0 = \mathbf{0}$ contributes $\mathbf{0}$ to the sum in $\mathsf{MRaccoon}$, and they use the same $\mathsf{H}_{\mathsf{sig}}$ and the same norm bound $\beta'_{\mathsf{z}}$. Hence

$$\mathsf{MRaccoon}^\star.\mathsf{Verify}(K^*, \mathbf{m}^*, \sigma) = 1 \iff \mathsf{MRaccoon}[\beta'_{\mathsf{z}}].\mathsf{Verify}(\hat{K}^*, \mathbf{m}^*, \sigma) = 1.$$

The output of $\mathcal{R}'$ passes the KOSK check: the keys of $\mathcal{A}$ come with valid secret keys, and $(\mathbf{0}, \mathbf{0})$ is a valid key pair. Each completed call to $\mathsf{MRaccoon}^\star.\mathsf{SIGN}_2$ on $(K, \mathbf{m})$ causes exactly one completed call to $\mathsf{MRaccoon}.\mathsf{SIGN}_2$ on $(\hat{K}, \mathbf{m})$, so $Q_{\mathsf{ctr}}[(K^*, \mathbf{m}^*)]$ in the game of $\mathcal{A}$ equals $Q_{\mathsf{ctr}}[(\hat{K}^*, \mathbf{m}^*)]$ in the game of $\mathcal{R}'$. The $l$ signatures output by $\mathcal{A}$ are pairwise distinct in both games. It follows that $\mathcal{R}'$ wins whenever $\mathcal{A}$ wins and neither $\mathsf{Coll}$ nor $\mathsf{Guess}$ occurs, so

$$\mathsf{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{MRaccoon}^\star,\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{SUF\text{-}CMA}}_{\mathsf{MRaccoon}[\beta'_{\mathsf{z}}],\mathcal{R}'}(\lambda) + \frac{q_s(q_h + q_s) + 1}{2^\gamma}.$$

Theorem 3, instantiated with the bound $\beta'_{\mathsf{z}}$, bounds the first term on the right-hand side. □

*Remark 1.* The reduction targets $\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]$ rather than $\mathsf{MRaccoon}[\beta_{\mathsf{z}}]$. A valid $\mathsf{MRaccoon}^\star$ forgery may satisfy $\beta_{\mathsf{z}} < \|(\mathbf{z}, 2^{\nu_{\mathsf{w}}}\mathbf{h})\|_2 \le \beta'_{\mathsf{z}}$, and $\mathsf{MRaccoon}[\beta_{\mathsf{z}}]$ would reject it. The larger bound preserves correctness, since honest $\mathsf{MRaccoon}$ signatures already satisfy $\beta_{\mathsf{z}}$. The security of $\mathsf{MRaccoon}[\beta'_{\mathsf{z}}]$ follows from the proof of Theorem 3, with $\beta_{\mathsf{z}}$ replaced by $\beta'_{\mathsf{z}}$ in the extraction step. This gives reductions to AOM-MISIS and MSIS with correspondingly larger norm bounds. The simulation of honest partial signatures is unchanged, because they are distributed exactly as in $\mathsf{MRaccoon}$.

## 6 Blind multi-signature scheme BMRaccoon

We now present our blind multi-signature scheme $\mathsf{BMRaccoon}$ , which builds on $\mathsf{MRaccoon}^\star$ (Fig. 5). The user $\mathcal{U}$ plays the role of the auxiliary signer $\mathcal{S}_0$ and also performs the aggregation. The auxiliary nonce randomness $\mathbf{r}'$ therefore becomes the user's blinding vector, which masks both the aggregated nonce and the final response.

Signing is a two-round (four-move) protocol between the user and the signers $\mathcal{S}_1, \ldots, \mathcal{S}_n$. We outline it below and give the full description in Fig. 7.

<u>*Round 1.*</u> The user samples the blinding vector $\mathbf{r}'$ and generates a homomorphic encryption key pair $(ek, dk)$. It sends $ek$ and a commitment $C$ to $(\mathbf{r}', \mathbf{m})$ to the signers. Each signer $\mathcal{S}_i$ replies with its nonce list $com_i$.

*Round 2.* The user computes the nonce coefficients $\mathbf{b}$ from the signers' nonces and uses them to form the aggregated nonce. It masks this nonce by adding $\mathbf{A}\mathbf{r}'$, rounds the result to obtain $\tilde{\mathbf{w}}$, and computes the challenge $c$. It then sends the nonce lists $W$, the encryptions $ctx_{\mathbf{b}}$ and $ctx_c$ of $\mathbf{b}$ and $c$, and a proof $\pi_1$ that these values were computed correctly from the committed $(\mathbf{r}', \mathbf{m})$.

Each signer $\mathcal{S}_i$ verifies $\pi_1$ and checks that $com_i$ is in $W$. It then evaluates its partial signature homomorphically on $ctx_c$ and $ctx_{\mathbf{b}}$, which gives an encryption $ctx_{\mathbf{z}_i}$ of its partial signature $\mathbf{z}_i$. It sends $ctx_{\mathbf{z}_i}$ to the user, together with a proof $\pi_{i,2}$ that the evaluation is correct.

*Output.* The user verifies every proof $\pi_{i,2}$ and decrypts each $ctx_{\mathbf{z}_i}$ to recover $\mathbf{z}_i$. It adds the partial signatures and the mask $\mathbf{r}'_{[m-k]}$ to obtain $\mathbf{z}$. Finally, it computes the hint $\mathbf{h}$ and outputs the signature $\sigma := (c, \mathbf{z}, \mathbf{h})$.

*Language description.* The protocol uses NIZK arguments for two languages, with $\hat{K} := (\mathbf{0}, K)$. The user's proof $\pi_1$ shows membership in $\mathsf{L}_1$, and the proof $\pi_{i,2}$ of signer $\mathcal{S}_i$ shows membership in $\mathsf{L}_{i,2}$. As stated in Sec. 3.5, the witnesses implicitly contain all encryption coins and the opening of $C$.

$$\mathsf{L}_1 := \left\{ \mathsf{stmt}_1 := \big(C,\ ek,\ \hat{K},\ W,\ ctx_{\mathbf{b}},\ ctx_c\big) \;\middle|\; \begin{array}{l} \exists\, \big(dk, \mathbf{r}', \mathbf{m}, (b^{(j)})_{j\in[\ell]}, \tilde{\mathbf{w}}, \mathbf{e}_{\mathsf{w}}, c\big) \text{ s.t.,} \\ (ek, dk) := \mathsf{HE.KGen}(pp_{he}) \\ \mathsf{CS.Ver}\big((\mathbf{r}', \mathbf{m}), C\big) = 1, \quad \|\mathbf{r}'\|_2 \le \beta_{\mathsf{w}'}, \\ \big(b^{(j)}\big)_{j\in[\ell]} = \mathsf{H}_{\mathsf{non}}\big(W, \mathbf{m}, \hat{K}\big), \\ \forall j \in [0,\ell] : \mathbf{w}^{(j)} = \sum_{i'=1}^{n} \mathbf{w}_{i'}^{(j)} \\ \mathbf{w}^{(0)} + \sum_{j=1}^{\ell} b^{(j)} \mathbf{w}^{(j)} + \mathbf{A}\mathbf{r}' = 2^{\nu_{\mathsf{w}}} \tilde{\mathbf{w}} + \mathbf{e}_{\mathsf{w}} \pmod q, \\ \|\mathbf{e}_{\mathsf{w}}\|_\infty < 2^{\nu_{\mathsf{w}}-1}, \\ ctx_{b^{(j)}} = \mathsf{HE.Enc}\big(ek, b^{(j)}\big) \ \ \forall j \in [\ell], \\ c = \mathsf{H}_{\mathsf{sig}}\big(\tilde{\mathbf{w}}, \mathbf{m}, \hat{K}\big), \\ ctx_c = \mathsf{HE.Enc}(ek, c) \end{array} \right\}.$$

The language $\mathsf{L}_1$ shows that the user computed the nonce coefficients, the masked nonce, the challenge and their encryptions honestly. It guarantees that $(b^{(j)})_{j\in[\ell]}$ and $c$ are derived from the committed message $\mathbf{m}$ and blinding vector $\mathbf{r}'$, that $\tilde{\mathbf{w}}$ is the correct rounding of the aggregated nonce including the shift $\mathbf{A}\mathbf{r}'$, and that $ctx_{b^{(j)}}$ and $ctx_c$ encrypt exactly these values. It also shows that $ek$ is well formed. Since $C$ binds $(\mathbf{r}', \mathbf{m})$, the user cannot adapt the shift, the message or the challenge to the signers' responses.

$$\mathsf{L}_{i,2} := \left\{ \mathsf{stmt}_{i,2} := \big(vk_i,\ com_i,\ ctx_{\mathbf{z}_i}, ctx_c,\ ctx_{\mathbf{b}},\ ek\big) \;\middle|\; \begin{array}{l} \exists\, \big(\mathbf{s}_i, \mathbf{e}_{\mathsf{t}_i}, \mathbf{r}_i^{(0)}, \ldots, \mathbf{r}_i^{(\ell)}, ctx_{\mathbf{r}_i}\big) \text{ s.t.,} \\ \mathbf{t}_i = \mathbf{A}\,\mathbf{s}_i \pmod q, \quad \mathbf{t}_i = 2^{\nu_{\mathsf{t}}} vk_i + \mathbf{e}_{\mathsf{t}_i} \pmod q, \quad \|\mathbf{e}_{\mathsf{t}_i}\|_\infty < 2^{\nu_{\mathsf{t}}-1}, \\ \mathbf{w}_i^{(j)} = \mathbf{A}\,\mathbf{r}_i^{(j)} \pmod q \ \ \forall j \in [0,\ell], \\ \|\mathbf{s}_i\| \le \beta_{\mathsf{sk}}, \quad \|\mathbf{r}_i^{(j)}\| \le \beta_{\mathsf{w}} \ \ \forall j \in [0,\ell], \\ ctx_{\mathbf{r}_i} = \mathsf{HE.Enc}\big(ek, \mathbf{r}_{i,[m-k]}^{(0)}\big), \\ \boldsymbol{\alpha}_i = \big(\mathbf{s}_{i,[m-k]}, \mathbf{r}_{i,[m-k]}^{(1)}, \ldots, \mathbf{r}_{i,[m-k]}^{(\ell)}\big), \\ ctx_{\mathbf{z}_i} = \mathsf{HE.Eval}\big(ek, f_{\boldsymbol{\alpha}_i}, ctx_c, ctx_{b^{(1)}}, \ldots, ctx_{b^{(\ell)}}, ctx_{\mathbf{r}_i}\big) \end{array} \right\}.$$

The language $\mathsf{L}_{i,2}$ shows that signer $\mathcal{S}_i$ computed its encrypted response honestly. It guarantees that $vk_i$ is well formed and that the secret key and the nonces in $com_i$ are short. It also guarantees that $ctx_{\mathbf{z}_i}$ is the evaluation of $f_{\boldsymbol{\alpha}_i}$ (Sec. 3.5) on the encrypted challenge, the encrypted coefficients and an encryption of $\mathbf{r}_{i,[m-k]}^{(0)}$, using the signer's secrets $\boldsymbol{\alpha}_i$. A malicious signer therefore cannot deviate from this computation, which rules out selective-failure attacks on blindness.

### 6.1 Security

**Theorem 6 (Correctness of BMRaccoon).** *Let $\beta'_{\mathsf{z}}$ be as in Theorem 4 and let $\beta_{\mathsf{bms}} = \beta'_{\mathsf{z}}$. Assume that the norm bounds in $\mathsf{L}_1$ and $\mathsf{L}_{i,2}$ are $\beta_{\mathsf{w}'}$, $\beta_{\mathsf{sk}}$ and $\beta_{\mathsf{w}}$. Assume further that CS is correct, that HE is correct with error $\varepsilon_{\mathsf{HE}}$ for every $f_\alpha \in \mathcal{F}$ and every input (including flooded evaluation), and that $\mathsf{NIZK}_{\mathcal{L}_1}$ and $\mathsf{NIZK}_{\mathcal{L}_{i,2}}$ are complete with errors $\varepsilon_1$ and $\varepsilon_2$. Then for all honestly generated keys and every message $\mathbf{m}$, an honest execution of BMRaccoon.Sign outputs $\sigma \neq \perp$ with $\mathsf{BMRaccoon.Verify}(K, \mathbf{m}, \sigma) = 1$, except with probability at most*

$$(n(\ell+2)+1)\, 2^{-Nm/10} + \varepsilon_1 + n\,\varepsilon_2 + n\,\varepsilon_{\mathsf{HE}}.$$

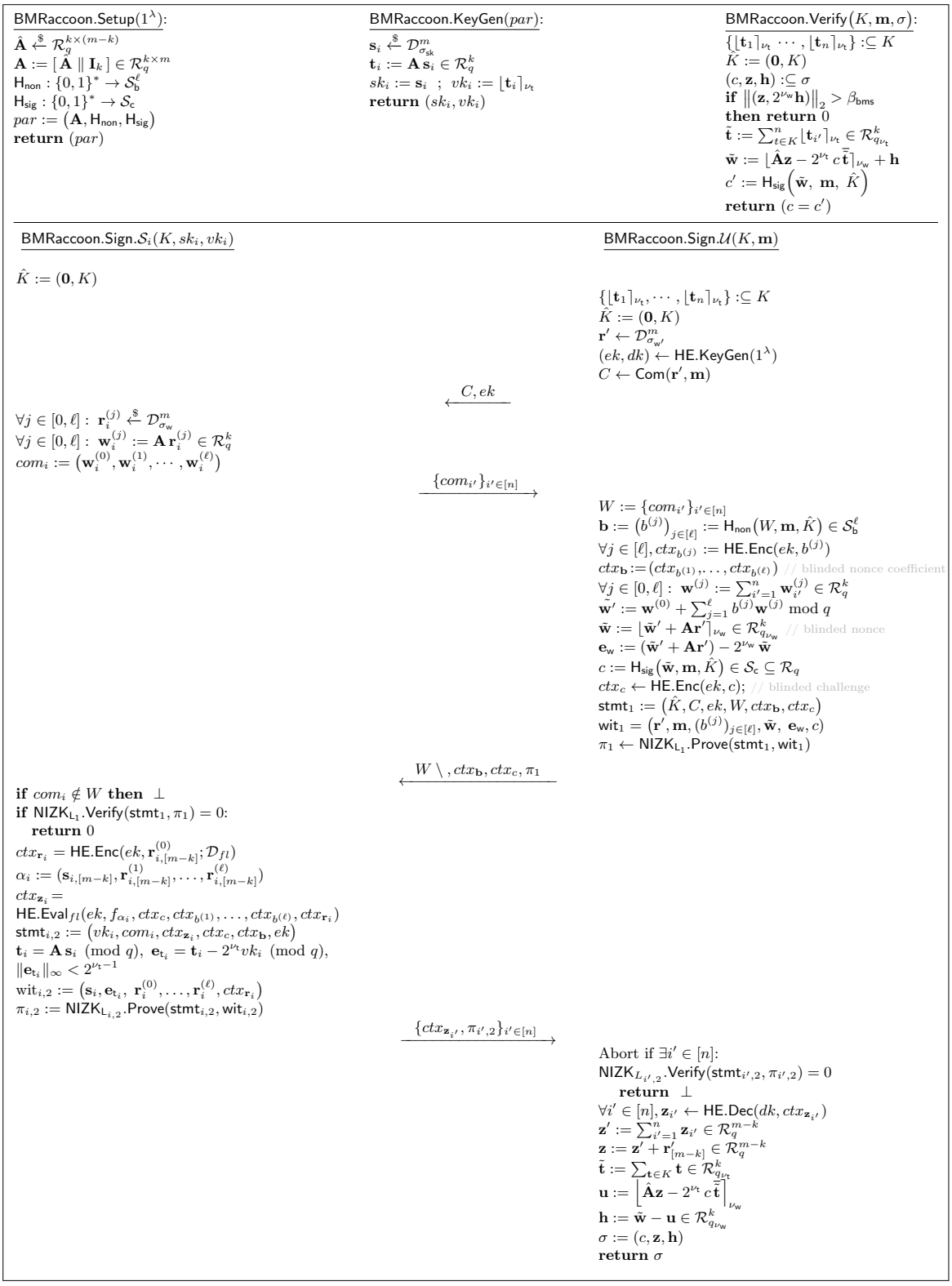



**Fig. 7.** BMRaccoon: Blind Multi-Raccoon

*Proof.* An honest execution of BMRaccoon corresponds to an honest execution of MRaccoon$^\star$ on $\widehat{K} = (0, K)$, in which the user acts as the auxiliary signer $S_0$ with shift $\mathbf{r}'$ and performs the aggregation. In particular, the user computes $\mathbf{b} = \mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \widehat{K})$, $\widetilde{\mathbf{w}} = \lfloor \mathbf{A}(\mathbf{r}_{\mathsf{agg}} + \mathbf{r}') \rceil_{\nu_{\mathsf{w}}}$, and $c = \mathsf{H}_{\mathsf{sig}}(\widetilde{\mathbf{w}}, \mathbf{m}, \widehat{K})$, exactly as in MRaccoon$^\star$.

The two executions differ in the following respects. First, the signers receive $c$ and $\mathbf{b}$ only in encrypted form. By correctness of $\mathsf{HE}$, the decryption of $ctx_{z_i}$ equals

$$f_{\alpha_i}(c, \mathbf{b}, \mathbf{r}^{(0)}_{i,[m-k]}) = c\,\mathbf{s}_{i,[m-k]} + \mathbf{r}^{(0)}_{i,[m-k]} + \sum_j b^{(j)} \mathbf{r}^{(j)}_{i,[m-k]},$$

which coincides with the partial signature $\mathbf{z}_i$ of $\mathsf{MRaccoon}^\star$, except with probability at most $n\varepsilon_{\mathsf{HE}}$.

Second, the user computes $\mathbf{z} = \sum_i \mathbf{z}_i + \mathbf{r}'_{[m-k]}$ and $\mathbf{h}$ as in $\mathsf{MRaccoon}^\star.\mathsf{SignAgg}$. Thus, the resulting signature $\sigma$ is an honestly generated $\mathsf{MRaccoon}^\star$ signature. Since $\mathsf{BMRaccoon.Verify}$ coincides with $\mathsf{MRaccoon}^\star.\mathsf{Verify}$, except that its bound is $\beta_{\mathsf{bms}} = \beta'_{\mathsf{z}}$, Theorem 4 implies that verification accepts, except with probability $(n(\ell+2)+1)2^{-Nm/10}$.

Third, an honest execution of $\mathsf{BMRaccoon}$ may abort during proof verification. Conditioned on $\|\mathbf{s}_i\| \le \beta_{\mathsf{sk}}$, $\|\mathbf{r}^{(j)}_i\| \le \beta_{\mathsf{w}}$, and $\|\mathbf{r}'\| \le \beta_{\mathsf{w}'}$, the honest witnesses belong to $\mathcal{L}_1$ and $\mathcal{L}_{i,2}$. These norm bounds are already accounted for in Theorem 4. Moreover, $\mathsf{CS.Ver} = 1$ by correctness of $\mathsf{CS}$, and $\|\mathbf{e}_{\mathsf{w}}\|_\infty < 2^{\nu_{\mathsf{w}}-1}$ and $\|\mathbf{e}_{t_i}\|_\infty < 2^{\nu_{\mathsf{t}}-1}$ by Lemma 2; all other clauses hold by construction. By completeness, $\pi_1$ is accepted except with probability $\varepsilon_1$, and each $\pi_{i,2}$ is accepted except with probability $\varepsilon_2$. Since verification is deterministic, all signers agree on the acceptance of $\pi_1$. Hence, by a union bound, all proofs are accepted except with probability $\varepsilon_1 + n\varepsilon_2$.

These are the only points at which an honest execution can output $\perp$. The proof follows by combining the above bounds. □

**Theorem 7 (One-more unforgeability (OMUF)).** *Let $\mathcal{A}$ be any PPT adversary against OMUF of* $\mathsf{BMRaccoon}$ *(Fig. 8) that opens at most $q_s$ sessions, then there exist PPT adversaries $\mathcal{B}_1, \mathcal{B}'_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4$, one for each computational hop, and an adversary $\mathcal{R}$ against SUF-CMA of* $\mathsf{MRaccoon}^\star$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{OMUF}}_{\mathsf{BMRaccoon},\mathcal{A}}(\lambda) \le\ & \mathsf{Adv}^{\mathsf{SetupInd}}_{\mathsf{CS},\mathcal{B}_1}(\lambda) + q_s \mathsf{Adv}^{\mathsf{Ext}}_{\mathsf{CS},\mathcal{B}'_1}(\lambda) + q_s \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{NIZK}_2,\mathcal{B}_2}(\lambda) \\ & + q_s \mathsf{Adv}^{\mathsf{SND}}_{\mathsf{NIZK}_1,\mathcal{B}_3}(\lambda) + q_s \varepsilon_{\mathsf{cp}} + \mathsf{Adv}^{\mathsf{SUF}}_{\mathsf{MRaccoon}^\star,\mathcal{R}}(\lambda).\end{aligned}$$

*Proof.* We prove the one-more unforgeability of $\mathsf{BMRaccoon}$ via a sequence of hybrid games depicted in Fig. 8, where $\mathsf{G}_i$ denotes the $i$-th hybrid. We describe each game below.

Game $\mathsf{G}_0$: This is the original game $\mathsf{OMUF}_{\mathsf{BMRaccoon}}$ in Fig. 8, with the colored boxes omitted. The challenger answers all oracle queries by honestly executing the protocol. Suppose $\mathcal{A}$ opens $q_s$ signing sessions and completes $q_c$ of them. At the end of the game, $\mathcal{A}$ outputs $f$ forgeries $\{(K^*_l, m^*_l, \sigma^*_l := (c^*_l, \mathbf{z}^*_l, \mathbf{h}^*_l))\}_{l\in[f]}$, together with, for each $l \in [f]$, the secret keys $\{sk_{i,l}\}_{i\in[2,n]}$ of all keys in $K^*_l$ other than the honest signer's. $\mathcal{A}$ wins if $f > q_c$, the pairs $(K^*_l, m^*_l)$ are pairwise distinct, and every $\sigma^*_l$ is valid. We denote the We denote the adversary's winning probability in this game as

$$\Pr[\mathsf{G}_0 = 1] = \mathsf{Adv}^{\mathsf{OMUF}}_{\mathsf{BMRaccoon},\mathcal{A}}(\lambda).$$

Game $\mathsf{G}_1$ (commitment extractability): We move from $\mathsf{G}_0$ to $\mathsf{G}_1$ through an intermediate game $\mathsf{G}'_1$.

$\mathsf{G}_0 \to \mathsf{G}'_1$. In $\mathsf{G}'_1$, the challenger generates the commitment key $ck$ along with a trapdoor $\tau$ using the extractable setup $(ck, \tau) \leftarrow \mathsf{CS.SetupExt}(1^\lambda)$, instead of $ck \leftarrow \mathsf{CS.Setup}(1^\lambda)$. An adversary $\mathcal{B}_1$ that receives a challenge key $ck$ and simulates the rest of the game itself (it knows $sk_1$ and does not need $\tau$) shows that

$$\left|\Pr[\mathsf{G}_0] - \Pr[\mathsf{G}'_1]\right| \le \mathsf{Adv}^{\mathsf{SetupInd}}_{\mathsf{CS},\mathcal{B}'_1}(\lambda).$$

$\mathsf{G}'_1 \to \mathsf{G}_1$. In $\mathsf{G}_1$, upon receiving a user commitment $C$ in session $\mathsf{sid}$, the challenger additionally computes $(\mathbf{r}'_{\mathsf{sid}}, \mathbf{m}_{\mathsf{sid}}) := \mathsf{CS.Ext}(\tau, C)$, stores the values in $L_{\mathbf{r}'}$ and $L_{\mathbf{m}}$, and aborts if extraction fails (event $F_1$). The extracted values are not used yet, so $\mathsf{G}'_1$ and $\mathsf{G}_1$ are identical unless $F_1$ occurs. By extractability of $\mathsf{CS}$ and a union bound over at most $q_s$ sessions, there is a PPT adversary $\mathcal{B}_1$ such that

$$\left|\Pr[\mathsf{G}'_1] - \Pr[\mathsf{G}_1]\right| \le q_s \cdot \mathsf{Adv}^{\mathsf{Ext}}_{\mathsf{CS},\mathcal{B}'_1}(\lambda).$$

Combining the two hops,

$$\left|\Pr[\mathsf{G}_0] - \Pr[\mathsf{G}_1]\right| \le \mathsf{Adv}^{\mathsf{SetupInd}}_{\mathsf{CS},\mathcal{B}_1}(\lambda) + q_s \cdot \mathsf{Adv}^{\mathsf{Ext}}_{\mathsf{CS},\mathcal{B}'_1}(\lambda).$$

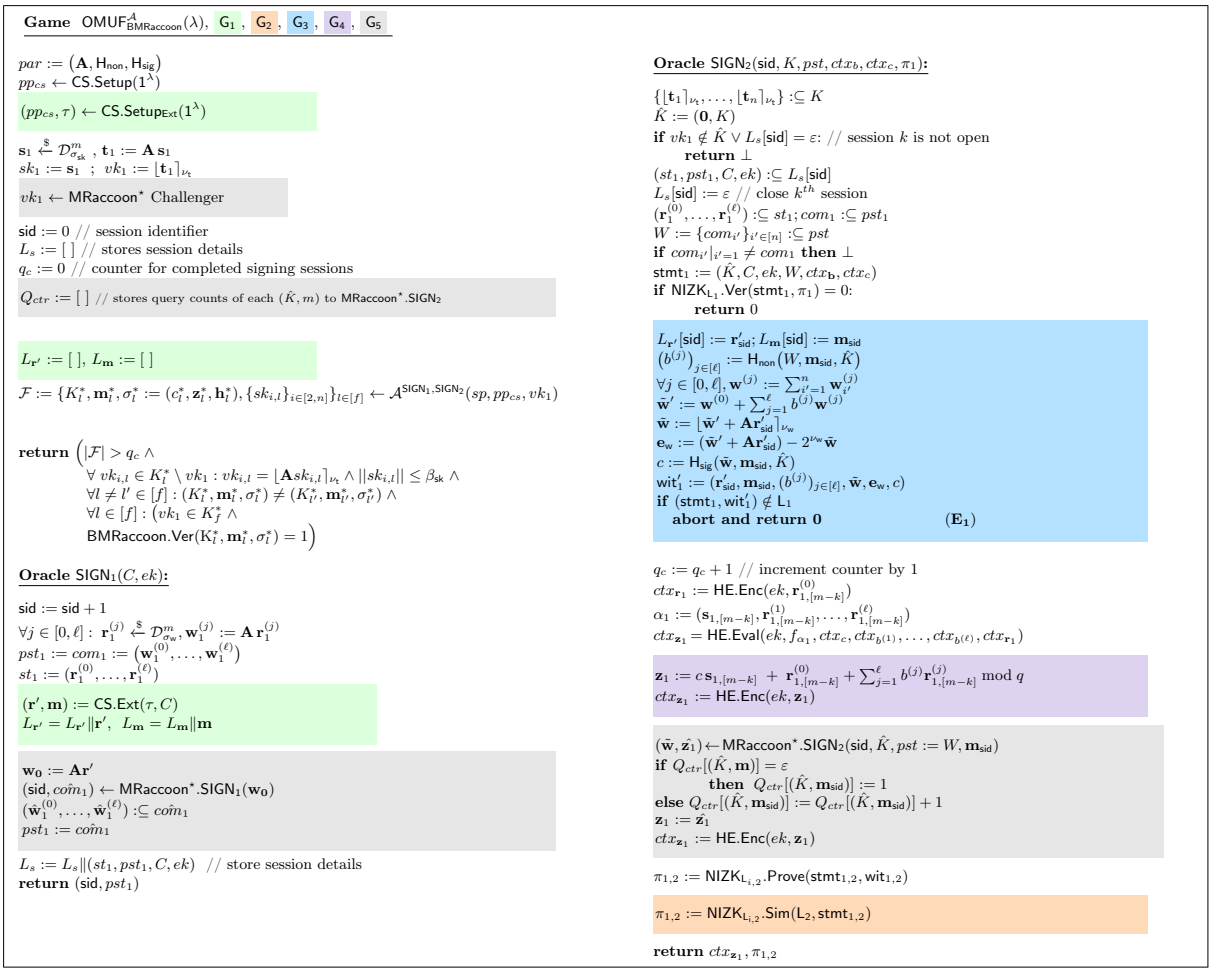



**Fig. 8.** The OMUF game for BMRaccoon and the hybrids used in the proof of Theorem 7. $\mathsf{G}_0$ is the original game, without the coloured boxes. The games $\mathsf{G}_1, \ldots, \mathsf{G}_5$ successively add the green, orange, blue, violet and grey boxes.

Game $\mathsf{G}_2$ (zero-knowledge): In this game, the challenger replaces the honest signer's proof $\pi_{1,2}$ with a simulated proof $\pi_{1,2} \leftarrow \mathsf{NIZK}_{\mathsf{L}_{1,2}}.\mathsf{Sim}(\mathsf{stmt}_{1,2})$. From the zero-knowledge property of NIZK $\pi_{1,2}$ and a union bound over at most $q_s$ sessions, there exists a PPT adversary $\mathcal{B}_2$ such that,

$$|\Pr[\mathsf{G}_1] - \Pr[\mathsf{G}_2]| \le q_s \cdot \mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{NIZK}_{\mathsf{L}_{1,2}},\mathcal{B}_2}(\lambda).$$

Game $\mathsf{G}_3$ (NIZK soundness): In each signing session, the challenger verifies $\pi_1$ as specified by the protocol and aborts the session if verification fails. Otherwise, it uses the tuple $(\mathbf{m}_{\mathsf{sid}}, \mathbf{r}'_{\mathsf{sid}})$ extracted from $C$ in $\mathsf{G}_1$ to recompute the values certified by $\pi_1$, and aborts if $\pi_1$ is accepted but the corresponding statement is not in $L_1$. Any such session yields a soundness adversary $\mathcal{B}_3$ which runs the game and outputs the false statement together with its accepting proof. By the soundness of $\pi_1$ and a union bound over the at most $q_s$ sessions,

$$\big|\Pr[\mathsf{G}_2] - \Pr[\mathsf{G}_3]\big| \le q_s \cdot \mathsf{Adv}^{\mathsf{SND}}_{\mathsf{NIZK}_{\mathsf{L}_1},\mathcal{B}_3}(\lambda).$$

Game $\mathsf{G}_4$ (circuit privacy): The challenger no longer evaluates the homomorphic evaluation on the user's ciphertexts. In every completed session it computes the honest signer's response in the clear,

$$\mathbf{z}_1 := c\,\mathbf{s}_{1,[m-k]} + \mathbf{r}^{(0)}_{1,[m-k]} + \textstyle\sum_{j\in[\ell]} b^{(j)} \mathbf{r}^{(j)}_{1,[m-k]} \bmod q,$$

where $c$ and $(b^{(j)})_j$ are the values recomputed in $\mathsf{G}_3$ from the extracted $(\mathbf{r}'_{\mathsf{sid}}, \mathbf{m}_{\mathsf{sid}})$. It then returns a fresh flooded encryption $\mathsf{ctx}_{z_1} \leftarrow \mathsf{Sim}_{\mathsf{HE}}(ek, \mathbf{z}_1) = \mathsf{HE.Enc}(ek, \mathbf{z}_1)$.

In $\mathsf{G}_3$, $\mathsf{ctx}_{z_1} = \mathsf{HE.Eval}_{\mathsf{priv}}(ek, f_{\alpha_1,\mathbf{x}_1}, \mathsf{ctx}_c, \mathsf{ctx}_b)$ with $\alpha_1 = (\mathbf{s}_{1,[m-k]}, \mathbf{r}^{(1)}_{1,[m-k]}, \ldots, \mathbf{r}^{(\ell)}_{1,[m-k]})$, $\mathbf{x}_1 = \mathbf{r}^{(0)}_{1,[m-k]}$ and output plaintext $f_{\alpha_1,\mathbf{x}_1}(c, (b^{(j)})_j) = \mathbf{z}_1$. This instance is admissible (Sec 3.5). Hence each replacement changes $\mathcal{A}$'s view by at most $\varepsilon_{\mathsf{cp}}$. A hybrid over the at most $q_s$ sessions gives

$$|\Pr[\mathsf{G}_3] - \Pr[\mathsf{G}_4]| \le q_s \cdot \mathsf{Adv}^{\mathsf{CP}}_{\mathsf{HE},\mathcal{B}_4}(\lambda) \le q_s \cdot \varepsilon_{\mathsf{cp}}.$$

Game $\mathsf{G}_5$ (Reduction to SUF-CMA of $\mathsf{MRaccoon}^\star$):

In this game, the challenger instead answers the signing queries using the signing oracles of SUF-CMA game of $\mathsf{MRaccoon}^\star$.

(i) $vk_1$ is the honest key of the $\mathsf{MRaccoon}^\star$ challenger;
(ii) $\mathsf{H}_{\mathsf{non}}$ and $\mathsf{H}_{\mathsf{sig}}$ queries are forwarded to the corresponding oracles.
(iii) In $\mathsf{SIGN}_1$, $com_1$ is obtained from $\mathsf{MRaccoon}^\star.\mathsf{SIGN}_1(\mathbf{w}_0)$, where $\mathbf{w}_0 := \mathbf{A}\mathbf{r}'$.
(iv) In each completed session, $\mathbf{z}_1$ is obtained from $\mathsf{MRaccoon}^\star.\mathsf{SIGN}_2(\mathsf{sid}, \hat{K}, W, \mathbf{m}_{\mathsf{sid}})$, where $com_0 = (\mathbf{A}\mathbf{r}'_{\mathsf{sid}}, 0, \ldots, 0)$, $\hat{K} := (\mathbf{0}, K)$, and $\mathbf{m}_{\mathsf{sid}}$ is the extracted message.

The games $\mathsf{G}_4$ and $\mathsf{G}_5$ are identical. $\mathsf{MRaccoon}^\star.\mathsf{SIGN}_2$ makes the same query $\mathsf{H}_{\mathsf{non}}(W, \mathbf{m}_{\mathsf{sid}}, \hat{K})$ as $\mathsf{G}_4$, and hence derives the same $(b^{(j)})_j$, $\tilde{\mathbf{w}}$ and $c$, and returns the same $\mathbf{z}_1$. It never outputs $\perp$, since $vk_1 \in \hat{K}$ and $com_1 \in W$ are checked beforehand and each $\mathsf{sid}$ is queried at most once. Hence $\Pr[\mathsf{G}_4] = \Pr[\mathsf{G}_5]$.

Note that $\mathsf{G}_5$ no longer uses $\mathsf{sk}_1$ or the nonces $\mathbf{r}^{(j)}_1$ and each completed session makes exactly one $\mathsf{MRaccoon}^\star.\mathsf{SIGN}_2$ query, on $(\hat{K}, \mathbf{m}_{\mathsf{sid}})$, and increments $q_c$ once, so $q_c = \sum_{(\hat{K},m)} Q_{\mathsf{ctr}}[(\hat{K}, \mathbf{m})]$.

Reduction $\mathcal{R}$: $\mathcal{R}$ plays the SUF-CMA game of $\mathsf{MRaccoon}^\star$ and runs $\mathcal{A}$ exactly as in $\mathsf{G}_5$, using its own oracles for the honest signer. It generates all the other parameters itself.

*Extracting a forgery.* Suppose $\mathcal{A}$ wins, i.e. it outputs $f > q_c$ pairwise distinct triples $(K^*_l, \mathbf{m}^*_l, \sigma^*_l)$ with valid keys. Group the forgeries by their pair $((0, K^*_l), \mathbf{m}^*_l)$. Since $f > q_c$, there exist a tuple $(K^*, \mathbf{m}^*)$ on which the number of forgeries $N_f$ exceeds $Q_{\mathsf{ctr}}[((0, K^*), \mathbf{m}^*)]$. $\mathcal{R}$ outputs these $N_f$ signatures on $((0, K^*), \mathbf{m}^*)$, together with $\mathcal{A}$'s secret keys and $\mathsf{sk}_0 := 0$ for the auxiliary key. This is a valid SUF-CMA forgery:

(i) the signatures are pairwise distinct, because the triples are distinct but share the same pair.
(ii) each one verifies for $\mathsf{MRaccoon}^\star$, because $\mathsf{BMRaccoon}.\mathsf{Ver}(K, m, \sigma) = \mathsf{MRaccoon}^\star.\mathsf{Verify}(K, \mathbf{m}, \sigma)$ when $\beta_{\mathsf{bms}} = \beta'_{\mathsf{z}}$.
(iii) $N_f > Q_{\mathsf{ctr}}[(0 \| K^*, \mathbf{m}^*)]$
(iv) the keys pass the KOSK check: $\mathcal{A}$'s keys satisfy the same. validity predicate, and $(0, 0)$ is a valid key pair since $\lfloor \mathbf{A} \cdot \mathbf{0} \rceil_{\nu_t} = 0$.

Hence $\Pr[\mathsf{G}_5] \le \mathsf{Adv}^{\mathsf{SUF}}_{\mathsf{MRaccoonStar},\mathcal{R}}(\lambda)$, and summing the bounds of all hops proves the Theorem 7. □

**Theorem 8 (Blindness).** *Let $\mathcal{A}$ be any PPT adversary against the blindness game of* $\mathsf{BMRaccoon}$ *(Fig. 9), then there exist PPT adversaries $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4$ one for each computational hop such that*

$$\mathsf{Adv}^{\mathsf{BLD}}_{\mathsf{BMRaccoon},\mathcal{A}}(\lambda) \le 2\mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{NIZK}_{\mathsf{L}_1},\mathcal{B}_1}(\lambda) + 2\mathsf{Adv}^{\mathsf{HID}}_{\mathsf{CS},\mathcal{B}_2}(\lambda) + 2n\mathsf{Adv}^{\mathsf{KS}}_{\mathsf{NIZK}_{\mathsf{L}_{i',2}},\mathcal{B}_3}(\lambda) + 2(\ell+1)\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{B}_4}(\lambda) + \epsilon_{\mathsf{stat}},$$

*where $\beta_r := n\beta_{\mathsf{w}}(1+\ell)$ and $\epsilon_{\mathsf{stat}} := 4\sqrt{\pi}\,\beta_r/\sigma_{\mathsf{w}'} + 2\sqrt{6\epsilon}$.*

*Proof.* We prove the blindness of $\mathsf{BMRaccoon}$ via a sequence of hybrid games depicted in Fig. 9, where $\mathsf{G}_i$ denotes the $i$-th hybrid. We describe each game below.

$\mathsf{G}_0$. This is the original blindness game $\mathsf{BLD}_{\mathsf{BMRaccoon}}$ from Fig. 9 (without colored boxes). Hence

$$\Pr[\mathsf{G}_0] = \mathsf{Adv}^{\mathsf{BLD}}_{\mathsf{BMRaccoon},\mathcal{A}}(\lambda).$$

$\mathsf{G}_1$ (zero-knowledge). In $\mathsf{USER}_1$, the challenger replaces $\pi_{1,\mathsf{k}}$ by simulated proof using $\mathsf{NIZK}_{\mathsf{L}_1}.\mathsf{Sim}(\mathsf{stmt}_{1,\mathsf{k}})$. In this and the subsequent hops, the modification is applied separately to each of the two sessions in the blindness experiment, accounting for the factor of 2 in each bound.




**Fig. 9.** The blindness game for BMRaccoon and the hybrids used in the proof of Theorem 8. $\mathsf{G}_0$ is the original game, without the coloured boxes. The games $\mathsf{G}_1, \ldots, \mathsf{G}_5$ successively add the orange, blue, violet, grey and green boxes.

By the zero-knowledge property of $\pi_1$, there exists a PPT adversary $\mathcal{B}_1$ such that

$$|\Pr[\mathsf{G}_0] - \Pr[\mathsf{G}_1]| \leq 2\mathsf{Adv}^{\mathsf{ZK}}_{\mathsf{NIZK}_{\mathsf{L}_1},\mathcal{B}_1}(\lambda).$$

$\mathsf{G}_2$ (commitment hiding). In $\mathsf{USER}_0$, the challenger replaces the commitment with a commitment to $\mathbf{0}$ vectors. By the commitment hiding property of $C$, there exists a PPT adversary $\mathcal{B}_2$ such that

$$|\Pr[\mathsf{G}_1] - \Pr[\mathsf{G}_2]| \leq 2\mathsf{Adv}^{\mathsf{HID}}_{\mathsf{CS},\mathcal{B}_2}(\lambda).$$

$\mathsf{G}_3$ (knowledge-soundness). In $\mathsf{USER}_2$ the challenger extracts the witnesses from all the $n$ proofs $\pi_{i',2,\mathsf{k}}$. If any of the extraction fails, the subroutine aborts; otherwise it proceeds as in $\mathsf{G}_2$. By the knowledge-soundness property of $\pi_{i',2}$ there exists a PPT adversary $\mathcal{B}_3$ such that

$$|\Pr[\mathsf{G}_2] - \Pr[\mathsf{G}_3]| \le 2n\mathsf{Adv}^{\mathsf{KS}}_{\mathsf{NIZK}_{\mathsf{L}_{i',2}},\mathcal{B}_3}(\lambda).$$

$\underline{\mathsf{G}_4}$. In $\mathsf{USER}_2$ the challenger directly computes partial signatures using the extracted witnesses. By the definition of $\mathsf{L}_2$, each signer's ciphertext encrypts the evaluation of the extracted values on the user's honestly generated ciphertexts. By decryption correctness, this yields the same plaintext as the original computation. Hence, $\mathsf{G}_3$ and $\mathsf{G}_4$ are identically distributed i.e.

$$\Pr[\mathsf{G}_3] = \Pr[\mathsf{G}_4]$$

$\underline{\mathsf{G}_5 \text{ (IND-CPA)}}$. In $\mathsf{USER}_1$, the challenger replaces $ctx_{\mathbf{b},\mathsf{k}}$ and $ctx_{c,\mathsf{k}}$ by encryptions of $\mathbf{0}$. By the IND-CPA security $ctx_{\mathbf{b}}$ and $ctx_c$ there exists a PPT adversary $\mathcal{B}_4$ such that

$$|\Pr[\mathsf{G}_4] - \Pr[\mathsf{G}_5]| \le 2(\ell+1)\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{HE},\mathcal{B}_4}(\lambda).$$

*Statistical argument in* $\mathsf{G}_5$. In $\mathsf{G}_5$, all user-generated components are independent of the bit $\mathsf{b}$. Now it remains to bound the statistical dependence of the output signatures on the signers' per-session randomness.

For a session $\mathsf{k}$ and message $\mathbf{m}$, define the aggregated nonce randomness as

$$\mathbf{r}_{\mathsf{agg}}(\mathsf{k},\mathbf{m}) := \sum_{i'=1}^{n}\left(\mathbf{r}^{(0)}_{i',\mathsf{k}} + \sum_{j=1}^{\ell} b^{(j)}\mathbf{r}^{(j)}_{i',\mathsf{k}}\right) \in \mathcal{R}_q^m,$$

where $(b^{(j)})_{j\in[\ell]} := \mathsf{H}_{\mathsf{non}}(W_{\mathsf{k}},\mathbf{m},\hat{K})$, and let $\mathbf{r}'$ denote the user's blinding vector for that session.

In $\mathsf{G}_5$, the signature generated in session $\mathsf{k}$ is determined by the public values, secret keys $\{\mathbf{s}_{i'}\}_{i'\in[n]}$, and the aggregated nonce randomness $\mathbf{r}_{\mathsf{agg}}(\mathsf{k},\mathbf{m}) + \mathbf{r}'$. In particular,

$$\mathbf{z} = c\sum_{i'\in[n]} \mathbf{s}_{i',[m-k]} + \mathbf{r}_{\mathsf{agg},[m-k]}(\mathsf{k},\mathbf{m}) + \mathbf{r}'_{[m-k]}, \quad \tilde{\mathbf{w}} = \left\lfloor \mathbf{A}\big(\mathbf{r}_{\mathsf{agg}}(\mathsf{k},\mathbf{m}) + \mathbf{r}'\big)\right\rceil_{\nu_{\mathsf{w}}},$$
$$c = \mathsf{H}_{\mathsf{sig}}(\tilde{\mathbf{w}},\mathbf{m},\hat{K}),$$

where the hint $\mathbf{h}$ is determined by $(\mathbf{z}, c, \tilde{\mathbf{w}})$. Thus, switching the challenge bit $\mathsf{b}$ swaps which session's aggregated nonce randomness is combined with the blinding vector for each message. By the data-processing inequality, the statistical distance between the views for $\mathsf{b}=0$ and $\mathsf{b}=1$ is at most

$$\sum_{\mathbf{m}\in\{\mathbf{m}_0,\mathbf{m}_1\}} \Delta\big(\mathbf{r}_{\mathsf{agg}}(0,\mathbf{m}) + \mathcal{D}^m_{\sigma_{\mathsf{w}'}},\ \mathbf{r}_{\mathsf{agg}}(1,\mathbf{m}) + \mathcal{D}^m_{\sigma_{\mathsf{w}'}}\big) = \sum_{\mathbf{m}} \Delta\big(\mathcal{D}^m_{\sigma_{\mathsf{w}'}},\ \delta_{\mathbf{m}} + \mathcal{D}^m_{\sigma_{\mathsf{w}'}}\big),$$

with $\delta_{\mathbf{m}} := \mathbf{r}_{\mathsf{agg}}(1,\mathbf{m}) - \mathbf{r}_{\mathsf{agg}}(0,\mathbf{m})$. The extracted nonces satisfy $\|\mathbf{r}^{(j)}_{i',\mathsf{k}}\| \le \beta_{\mathsf{w}}$, and $\|b\,\mathbf{r}\| \le \|b\|_1\|\mathbf{r}\|$. Hence $\|\mathbf{r}_{\mathsf{agg}}(\mathsf{k},\mathbf{m})\| \le \beta_r = n\,\beta_{\mathsf{w}}\,(1+\ell)$ and $\|\delta_{\mathbf{m}}\| \le 2\beta_r$. By the Corollary 2 we have

$$\Delta(\mathcal{D}^m_{\sigma_{\mathsf{w}'}}, \delta + \mathcal{D}^m_{\sigma_{\mathsf{w}'}}) \le \sqrt{\pi}\|\delta\|/\sigma_{\mathsf{w}'} + \sqrt{6\epsilon}.$$

So the total distance is at most $\epsilon_{\mathsf{stat}}$, and $\Pr[\mathsf{G}_5 - \frac{1}{2}] \le 2\sqrt{\pi}\|\delta\|/\sigma_{\mathsf{w}'} + 2\sqrt{6\epsilon} = \epsilon_{\mathsf{stat}}$. Combining the bounds across all hybrids establishes the theorem. □

## 7 Concrete instantiations and parameter settings

We instantiate BMRaccoon using the same building blocks as BRaccoon, following the instantiations of the linearly homomorphic encryption scheme, extractable commitment scheme, hash functions, and NIZK arguments given in Appendix [HMM+26]. In this section, we describe these instantiations and the additions that our multi-signer setting requires. We then give concrete parameters[8] and communication estimates for 128-bit computational and 40-bit statistical security, with at most $q_s = 2^{32}$ signing sessions. Table 1 summarizes the notation and parameters used in our BMRaccoon instantiation.

[8] https://github.com/palkeshJ02/BMRaccoon

**Table 1.** Parameters for BMRaccoon

| Parameters | Description |
|---|---|
| $\lambda$ | Security parameter |
| $n$ | Number of signers |
| $N, q$ | Ring dimension and prime modulus, with $N$ a power of two |
| $(k, m)$ | Dimension of public matrix $\mathbf{A} = [\hat{\mathbf{A}} \,\|\, \mathbf{I}_k] \in \mathcal{R}_q^{k \times m}$ |
| $\mathsf{H}_{\mathsf{non}}$ | Nonce hash, $\mathsf{H}_{\mathsf{non}} : \{0,1\}^* \to \mathcal{S}_b^{\ell}$ |
| $\mathsf{H}_{\mathsf{sig}}$ | Signature hash, $\mathsf{H}_{\mathsf{sig}} : \{0,1\}^* \to \mathcal{S}_c$ |
| $\ell + 1$ | $\ell = 2\lambda/(\log N)$, number of nonces for each signer |
| $\mathcal{S}_b$ | $\mathcal{S}_b = \{\pm 1, \pm X, \pm X^2, \ldots, \pm X^{N-1}\}$ |
| $(\mathcal{S}_c, \beta_{\mathsf{c}})$ | Challenge set, $\mathcal{S}_c = \{c \in \mathcal{R}_q \mid \|c\|_\infty = 1 \wedge \|c\|_1 = \beta_{\mathsf{c}}\}$ s.t. $\lvert\mathcal{S}_c\rvert \geq 2^{2\lambda}$ |
| $\sigma_{\mathsf{sk}}$ | Gaussian width of the secret key, $\sigma_{\mathsf{sk}} \geq \eta_{1/2}(\mathbb{Z}^{Nm})$ |
| $\sigma_{\mathsf{w}}$ | Gaussian width of the nonces $\mathbf{r}_i^{(j)}$, $\sigma_{\mathsf{w}} \geq \eta_{1/2}(\mathbb{Z}^{Nm})$ |
| $\sigma_{\mathsf{w}'}$ | Gaussian width of the user's blinding vector $\mathbf{r}'$, $\sigma_{w'} \geq \max\{4\sqrt{\pi}\,\beta_r/\varepsilon_{stat},\ \eta_{1/2}(\mathbb{Z}^{Nm})\}$ |
| $\nu_{\mathsf{t}}$ | Number of bits dropped from the verification key |
| $\nu_{\mathsf{w}}$ | Number of bits dropped from the (aggregated) nonce |
| $(q_{\nu_{\mathsf{t}}}, q_{\nu_{\mathsf{w}}})$ | Rounded moduli, $q_\nu := \lfloor q/2^\nu \rfloor = \lfloor q/2^\nu \rceil$ for $\nu \in \{\nu_{\mathsf{t}}, \nu_{\mathsf{w}}\}$ |
| $\beta_{\mathsf{sk}}$ | $\ell_2$-norm bound of a secret key, $\beta_{\mathsf{sk}} := e^{1/4}\sigma_{\mathsf{sk}}\sqrt{Nm}$ |
| $\beta_{\mathsf{w}}$ | $\ell_2$-norm bound of a nonce, $\beta_{\mathsf{w}} := e^{1/4}\sigma_{\mathsf{w}}\sqrt{Nm}$ |
| $\beta_{\mathsf{w}'}$ | $\ell_2$-norm bound of $\mathbf{r}'$, $\beta_{\mathsf{w}'} := e^{1/4}\sigma_{\mathsf{w}'}\sqrt{Nm}$ |
| $\beta_{\mathsf{r}}$ | $\ell_2$-norm bound of the aggregated signer nonce, $\beta_{\mathsf{r}} := n(1+\ell)\beta_{\mathsf{w}}$ |
| $\beta'_{\mathsf{z}}$ | $\ell_2$-norm bound of a valid $\mathsf{MRaccoon}^\star$ signature, $\beta'_{\mathsf{z}} := \sqrt{2}(n\beta_{\mathsf{c}}\beta_{\mathsf{sk}} + \beta_{\mathsf{r}} + \beta_{\mathsf{w}'}) + ((3n-1)\beta_{\mathsf{c}}2^{\nu_{\mathsf{t}}-1} + 4\cdot 2^{\nu_{\mathsf{w}}-1})\sqrt{Nk}$ |
| $\beta_{\mathsf{bms}}$ | $\ell_2$-norm bound of a valid BMRaccoon signature, $\beta_{\mathsf{bms}} := \beta'_{\mathsf{z}}$ |
| $q_s, q_h$ | Bounds on signing sessions and random-oracle queries |
| $Q$ | Number of AOM-MISIS samples, $Q := 1 + q_s(\ell+1)$ |
| $(\sigma_i)_{i\in[Q]}$ | AOM-MISIS widths, $\sigma_1 := \sigma_{\mathsf{sk}}$ and $\sigma_{1+i} := \sigma_{\mathsf{w}}$ for $i \in [q_s(\ell+1)]$ |
| $\beta_{\mathsf{s}}$ | AOM-MISIS $\ell_2$-norm bound on $\hat{\mathbf{s}}$, $\beta_{\mathsf{s}} := 4\beta'_{\mathsf{z}} + (2^{\nu_{\mathsf{w}}+1} + (3n-1)\beta_{\mathsf{c}}2^{\nu_{\mathsf{t}}})\sqrt{Nk} + 2\sqrt{2}(n-1)\beta_{\mathsf{c}}\beta_{\mathsf{sk}}$ |
| $\beta_{\mathsf{b}}$ | AOM-MISIS $\ell_1$-norm bound on $(\hat{b}_i\sigma_i)_{i\in[Q]}$, $\beta_{\mathsf{b}} := 2\sigma_{\mathsf{sk}}\beta_{\mathsf{c}}$ |
| $\beta_{\mathsf{u}}$ | AOM-MISIS $\ell_2$-norm bound on $(u_i/\sigma_i)_{i\in[Q]}$, $\beta_{\mathsf{u}} := 2/\sigma_{\mathsf{sk}} + \beta_{\mathsf{c}}\sqrt{8Nq_s}/\sigma_{\mathsf{w}}$ |

### 7.1 Building blocks

We instantiate the commitment scheme with BDLOP and the homomorphic encryption scheme with BGV, as in [HMM+26]. BGV supports the family of evaluation functions defined in Section 3.5.

The NIZK instantiation faces the same obstacle as in [HMM+26]. The module-lattice relations can be expressed as linear constraints over $\mathcal{R}_q$ and handled by lattice-based proof systems, but the statement $\mathsf{L}_1$ also contains hash computations. Whereas [HMM+26] only needs the challenge hash $\mathsf{H}_{\mathsf{sig}}$, our construction also needs the nonce hash $\mathsf{H}_{\mathsf{non}}$. Both hashes take the secret message $\mathbf{m}$ as input, so the proof must show that they were evaluated correctly. They are defined as follows:

- The nonce coefficients $(b^{(j)})_{j\in[\ell]} = \mathsf{H}_{\mathsf{non}}(W, \mathbf{m}, \hat{K})$ are obtained by hashing and mapping the output to signed monomials with SampleMon.
- The challenge $c = \mathsf{H}_{\mathsf{sig}}(\tilde{\mathbf{w}}, \mathbf{m}, \hat{K})$ is obtained by hashing and mapping the output to a sparse polynomial $c \in \mathcal{S}_{\mathsf{c}}$ with SampleInBall.

We therefore encode the statements as mixed R1CS instances, following [BS23], and prove them with the zero-knowledge LaBRADOR toolkit of Biasioli et al. [BBL+26]. Appendix A describes the concrete hash functions.

**Instantiating $\mathsf{L}_1$ and $\mathsf{L}_{i,2}$.** The user proof $\pi_1$ shows that, after receiving the signers' nonce commitments $W$, the user correctly derives all subsequent values from the message and the shift committed in the first round. Since $W$ and $\hat{K}$ are long public lists, every party prehashes them outside the circuit into the 384-bit digests $d_W := \mathsf{H}(\mathsf{Ser}(W))$ and $d_K := \mathsf{H}(\hat{K})$, and only these digests enter $\mathsf{L}_1$ (see Appendix A.2). Specifically, $\mathsf{L}_1$ establishes the following:

- $C$ is a valid extractable BDLOP commitment to $(\mathbf{r}', \mathbf{m})$, with $\|\mathbf{r}'\|_2 \leq \beta_{\mathsf{w}'}$.

– The nonce coefficients are correctly derived as

$$(b^{(j)})_{j\in[\ell]} = \mathsf{SampleMon}_\ell(\mathsf{SHAKE256}(d_W \,\|\, \mu)), \quad \text{where } \mu = \mathsf{H}(d_K \,\|\, \mathbf{m}).$$

– The blinded nonce $\tilde{\mathbf{w}}$ is the correct rounding of $\mathbf{w}^{(0)} + \sum_j b^{(j)}\mathbf{w}^{(j)} + \mathbf{A}\mathbf{r}'$. This is expressed as the linear relation

$$\mathbf{w}^{(0)} + \sum_j b^{(j)}\mathbf{w}^{(j)} + \mathbf{A}\mathbf{r}' = 2^{\nu_\mathsf{w}}\tilde{\mathbf{w}} + \mathbf{e}_w \pmod q,$$

together with a coefficient-wise bound on $\mathbf{e}_w$.
– The challenge is correctly derived as $c = \mathsf{SampleInBall}_{\beta_\mathsf{c}}(\rho)$, where $\rho = \mathsf{H}(\mu \,\|\, \mathsf{Ser}(\tilde{\mathbf{w}}))$.
– The ciphertexts $ctx_{b^{(j)}}$ and $ctx_c$ correctly encrypt $b^{(j)}$ and $c$, respectively, under encryption coins from the admissible set.

The signer proof $\pi_{i,2}$ shows that the encrypted partial signature was computed correctly. Specifically, $\mathsf{L}_{i,2}$ establishes the following:

– The key rounding relation $\mathbf{A}\mathbf{s}_i = 2^{\nu_\mathsf{t}}vk_i + \mathbf{e}_{t_i} \pmod q$ holds, with a coefficient-wise bound on $\mathbf{e}_{t_i}$.
– Each nonce commitment is correctly computed as $\mathbf{w}_i^{(j)} = \mathbf{A}\mathbf{r}_i^{(j)}$, and $\mathbf{s}_i$ and $\mathbf{r}_i^{(j)}$ satisfy their respective norm bounds.
– The ciphertext $ctx_{\mathbf{z}_i}$ correctly encrypts $\mathbf{z}_i = c\,\mathbf{s}_{i,[m-k]} + \mathbf{r}^{(0)}_{i,[m-k]} + \sum_j b^{(j)}\mathbf{r}^{(j)}_{i,[m-k]}$. It is obtained by evaluating $f_{\boldsymbol{\alpha}_i}$ on the user's ciphertexts and the additional encryption $ctx_{\mathbf{r}_i} = \mathsf{HE.Enc}(\mathsf{ek}, \mathbf{r}^{(0)}_{i,[m-k]}; \mathcal{D}_\mathsf{fl})$, where the flooding distribution $\mathcal{D}_\mathsf{fl}$ hides the evaluation noise while preserving decryption correctness.

### 7.2 Concrete parameters and communication cost

Table 2 lists our parameters for $n \in \{32, 64\}$ signers and $q_s = 2^{32}$ signing sessions, targeting 128-bit computational and 40-bit statistical security. The parameters are selected to ensure that the underlying key-recovery and forgery security assumptions, namely MSIS and AOM-MISIS, correspond to concretely hard instances at the target security level. In particular, AOM-MISIS admits reductions to MSIS and MLWE. Table 3 reports the corresponding estimated communication costs per signing session for these parameter choices.

**Table 2.** Concrete BMRaccoon parameters for 128-bit computational and 40-bit statistical security, with $n$ signers and $q_s = 2^{32}$ signing sessions.

| **Parameter** | $n = 32$ | $n = 64$ |
|---|---|---|
| Ring degree $N$ | 256 | 256 |
| Signature modulus $\lceil \log_2 q \rceil$ | 114 | 115 |
| BGV modulus $\lceil \log_2 Q_\mathsf{he} \rceil$ | 150 | 150 |
| Commitment modulus $\lceil \log_2 Q_\mathsf{ext} \rceil$ | 122 | 123 |
| Public dimension $k$ | 21 | 21 |
| Module width $m$ | 41 | 41 |
| Number of nonces $\ell$ | 32 | 32 |
| Challenge weight $\beta_\mathsf{c}$ | 61 | 61 |
| $\sigma_\mathsf{sk}$ | $2^{16.52}$ | $2^{16.52}$ |
| $\sigma_\mathsf{w}$ | $2^{42.96}$ | $2^{42.96}$ |
| $\sigma_{\mathsf{w}'}$ | $2^{102.87}$ | $2^{103.87}$ |
| $\nu_\mathsf{t}$ | 91 | 91 |
| $\nu_\mathsf{w}$ | 102 | 103 |
| Verification key (KB) | 15.46 | 16.13 |
| Signature (KB) | 67.92 | 68.56 |

**Table 3.** Estimated communication cost of BMRaccoon signing session (KB) for $q_s = 2^{32}$.

| **Component** | $n = 32$ | $n = 64$ |
|---|---|---|
| $\mathsf{NIZK}_{\mathsf{L}_1}$ | $\leq 125$ | $\leq 125$ |
| $\mathsf{NIZK}_{\mathsf{L}_{i,2}}$ | $\leq 125$ | $\leq 125$ |
| Ciphertexts | 7,377.60 | 7,377.60 |
| count × size | 53 × 139.200 | 53 × 139.200 |
| Extractable commitment $C$ | 257.66 | 259.78 |
| HE key $\mathsf{ek}$ | 134.43 | 134.43 |
| Nonce commitments | 83,426.11 | 165,765.60 |
| count × size | 33 × 2,528.064 | 65 × 2,550.240 |
| **Per signer** | **91,445.81** | **173,787.41** |

*Asymptotic parameter selection.* Let $\beta_{\mathsf{lwe}}$ denote the norm of the noise in the underlying MLWE assumption. We first select $N, m, k, \beta_{\mathsf{lwe}}$ such that $N$ is a power of two with $N \geq 2\lambda$, $m, k = \mathsf{poly}(\lambda)$, and $\beta_{\mathsf{lwe}} \geq m \log N$, as in [ZT25]. (For the concrete instantiation we enumerate plausible tuples $(N, m, k, \beta_{\mathsf{lwe}})$ and pick the most efficient one.) Given the maximum number of signers $n$ and the bound $q_s$ on signing sessions, we then set the remaining parameters as follows.

- Set $\ell := \lceil 2\lambda / \log_2 N \rceil$, so that $|\mathcal{S}_b|^\ell = (2N)^\ell \geq 2^{2\lambda}$.
- Set $\beta_{\mathsf{c}}$ as the smallest integer such that $2^{\beta_{\mathsf{c}}}\binom{N}{\beta_{\mathsf{c}}} \geq 2^{2\lambda}$ and $\Pr[\mathsf{BadHash}] \leq 2^{-\lambda}$.
- $\sigma_{\mathsf{sk}} = \max\{4\beta_{\mathsf{lwe}}\sqrt{mN},\ \sqrt{\log(6mN)/\pi}\}$. The first term is usually the leading term.
- $\sigma_{\mathsf{w}} = \max\{\frac{\sigma_{\mathsf{sk}}\beta_{\mathsf{c}}\sqrt{8Nq_s}}{2}, 2\sqrt{2\pi}Nq^{\frac{1}{m}+\frac{2}{N(m-k)}},\ \sqrt{(\log(2Nm)+\lambda)/\pi}\}$. The first term is usually the leading term.
- $\sigma_{\mathsf{w}'} = \max\{4\sqrt{\pi}\,\beta_{\mathsf{r}}/\varepsilon_{\mathsf{stat}},\ \sqrt{(\log(2Nm)+\lambda)/\pi}\}$, where $\beta_{\mathsf{r}} = n(1+\ell)\beta_{\mathsf{w}}$. We target 40-bit statistical security by setting $\varepsilon_{\mathsf{stat}} = 2^{-40}$. The first term is usually the leading term.
- $\nu_{\mathsf{t}} = \lfloor \log_2(\sigma_{\mathsf{w}'}/(n\beta_{\mathsf{c}})) \rfloor$ and $\nu_{\mathsf{w}} = \lfloor \log_2 \sigma_{\mathsf{w}'} \rfloor$. Since $\sigma_{\mathsf{w}'}$ is the dominant width of the aggregated nonce, the rounding terms of $\beta'_{\mathsf{z}}$ are then at most $\frac{3}{2}\sigma_{\mathsf{w}'}\sqrt{Nk}$ and $\frac{5}{2}\sigma_{\mathsf{w}'}\sqrt{Nk}$, i.e. within a constant factor of $\beta_{\mathsf{w}'}$.
- $\beta_{\mathsf{z}}$ is set as in Theorem 1, and $\beta_{\mathsf{bms}} = \beta'_{\mathsf{z}}$ as in Theorem 4.
- $\beta_{\mathsf{s}}, \beta_{\mathsf{b}}, \beta_{\mathsf{u}}$ are set as in Theorem 2, with $\beta_{\mathsf{z}}$ replaced by $\beta'_{\mathsf{z}}$ (Remark 1). Denote $\beta_{\mathsf{sis}} = \beta_{\mathsf{s}} + \beta_{\mathsf{b}}\sqrt{mN}$. Since $\beta_{\mathsf{msis}} = 4\beta'_{\mathsf{z}} + 2^{\nu_{\mathsf{w}}+2}\sqrt{Nk} \leq \beta_{\mathsf{s}}$, a single MSIS instance covers both Theorem 2 and Theorem 3.
- Select a prime $q$ with $q > \beta_{sis}$ and $q > 4\beta_{z'}$ that satisfies the hypothesis of Lemma 2 for both $\nu_t$ and $\nu_w$, such that $\mathrm{MSIS}_{q,k,m,\beta_{sis}}$ and $\mathrm{MLWE}_{q,N,k,m-1,\beta_{lwe}}$ are exponentially hard in $\lambda$.

The building blocks are instantiated on top of these choices.

- For $\mathsf{CS}$, let $n_r$ be the number of ring elements in the commitment randomness $\boldsymbol{\gamma} \in \{0,1\}^{n_r N}$, and let $p_{\mathsf{ext}}$ be the extraction modulus. The parameters satisfy $p_{\mathsf{ext}} > 2B_\infty(\sigma_{\mathsf{w}'})$ and $B_\infty(\sigma_{\mathsf{w}'}) + p_{\mathsf{ext}} N n_r < Q_{\mathsf{ext}}/2$, where $Q_{\mathsf{ext}}$ is the modulus of the commitment scheme. These conditions guarantee correct extraction. The dimensions are chosen so that binding, hiding and setup indistinguishability reduce to hard MSIS and MLWE instances.
- For $\mathsf{HE}$, let $B_{\mathsf{plain}}$ bound the coefficients of the decrypted partial signature $\mathbf{z}_i$, let $B_{\mathsf{shift}}$ bound the shift that homomorphic evaluation induces on the encryption coins, and let $B_{\mathsf{final}}$ bound the total decryption noise after evaluation and flooding. The plaintext modulus satisfies $p_{\mathsf{he}} > 2B_{\mathsf{plain}}$, the flooding width is $\sigma_{\mathsf{fl}} = B_{\mathsf{shift}}\sqrt{2\pi q_s(m-k)}$, and the ciphertext modulus $Q_{\mathsf{he}}$ satisfies $B_{\mathsf{plain}} + p_{\mathsf{he}}B_{\mathsf{final}} < Q_{\mathsf{he}}/2$. The rank $k_{\mathsf{he}}$ is chosen so that IND-CPA security reduces to hard MLWE instances.
- For the NIZKs, we use ZK-LaBRADOR, embedding the lifted lattice equations into the arithmetic R1CS over modulus $M = 2^{256} + 1$. Biasioli et al. [BBL+26] report that proof sizes remain essentially constant across different applications and input sizes, as they are dominated by the final round of LaBRADOR, with zero-knowledge proofs occupying approximately 125 KB. We use this estimate for $\pi_1$ and each $\pi_{i,2}$.

*How the parameters scale with $n$.* We analyze BMRaccoon targeting 128-bit computational and 40-bit statistical security, considering two parameter sets with $n \in \{32, 64\}$ signers and $q_s = 2^{32}$ signing

queries. Across all sets, $N = 256$, $\ell = 32$, and $\beta_c = 61$. Increasing $n$ affects the blinding width through $\beta_{\mathsf{r}} = n(1+\ell)\beta_{\mathbf{w}}$, adding approximately one bit to $\sigma_{\mathbf{w}'}$, $\nu_{\mathbf{w}}$, and $q$, while leaving $(k, m)$ unchanged. Doubling $n$ increases signature size by only about 1%, but multiplies per-signer and total session communication by approximately 1.9 and 3.8, respectively.

*Communication cost.* Table 3 reports the per-signer communication of BMRaccoon and the total over all $n$ signers. It has three components. First, each signer receives the list $W = (\mathsf{com}_1, \ldots, \mathsf{com}_n)$ of $n$ nonce commitments, where each $\mathsf{com}_i = (\mathbf{w}_i^{(0)}, \ldots, \mathbf{w}_i^{(\ell)})$ consists of $\ell + 1 = 33$ unrounded nonces in $R_q^k$. Second, there are $\ell + 1 = 33$ ciphertexts encrypting $c, b^{(1)}, \ldots, b^{(\ell)}$ from the user and $m - k = 20$ ciphertexts encrypting $\mathbf{z}_i$ from the signer. Third, there are the two proofs $\pi_1$ and $\pi_{i,2}$, each of about 125 KB.

Nonce commitments dominate the cost, accounting for 91–95% of the per-signer communication, while ciphertexts contribute 4–8% and proofs less than 0.3%. Since every signer receives all $n$ commitment vectors, commitment traffic grows linearly in $n$ per signer and quadratically over a session. Nonce commitments are therefore the main bottleneck, followed by the coefficient-wise BGV encryption. The proofs, although large in absolute terms, are negligible in comparison.

## References


AAB+24. M. A. Aardal, D. F. Aranha, K. Boudgoust, S. Kolby, and A. Takahashi. Aggregating falcon signatures with labrador. In *Annual International Cryptology Conference*, pages 71–106. Springer, 2024.

AAEBB20. N. Alkeilani Alkadri, R. El Bansarkhani, and J. Buchmann. Blaze: practical lattice-based blind signatures for privacy-preserving applications. In *International Conference on Financial Cryptography and Data Security*, pages 484–502. Springer, 2020.

AAHJ21. N. Alkeilani Alkadri, P. Harasser, and C. Janson. Blindor: an efficient lattice-based blind signature scheme from or-proofs. In *International Conference on Cryptology and Network Security*, pages 95–115. Springer, 2021.

AKSY22. S. Agrawal, E. Kirshanova, D. Stehlé, and A. Yadav. Practical, round-optimal lattice-based blind signatures. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS 2022)*. ACM, 2022.

BBL+26. B. Biasioli, M. Bolboceanu, V. Lyubashevsky, A. M. Gallardo, M. Osadnik, G. Seiler, and P. Steuer. A toolkit for succinct lattice-based zero knowledge proofs. In *ACM Annual Conference on Computer and Communications Security*, 2026.

BCK+14. F. Benhamouda, J. Camenisch, S. Krenn, V. Lyubashevsky, and G. Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 551–572. Springer, 2014.

BFM+26. C. Bouillaguet, T. Feneuil, J. Maire, M. Rivain, J. Sauvage, and D. Vergnaud. Blinding post-quantum hash-and-sign signatures. In *2026 IEEE Symposium on Security and Privacy (SP)*, pages 1–18. IEEE, 2026.

BKL+24. C. Boschini, D. Kaviani, R. W. F. Lai, G. Malavolta, A. Takahashi, and M. Tibouchi. Ringtail: Practical Two-Round Threshold Signatures from Learning with Errors, 2024. Published: Cryptology ePrint Archive, Paper 2024/1113.

BLL+22. F. Benhamouda, T. Lepoint, J. Loss, M. Orrù, and M. Raykova. On the (in) security of ros. *Journal of Cryptology*, 35(4):25, 2022.

BLNS23. W. Beullens, V. Lyubashevsky, N. K. Nguyen, and G. Seiler. Lattice-based blind signatures: Short, efficient, and round-optimal. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS 2023)*. ACM, 2023.

BLRL+18. S. Bai, T. Lepoint, A. Roux-Langlois, A. Sakzad, D. Stehlé, and R. Steinfeld. Improved security proofs in lattice-based cryptography: using the rényi divergence rather than the statistical distance. *Journal of Cryptology*, 31(2):610–640, 2018.

BN06. M. Bellare and G. Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS 2006)*. ACM, 2006.

Bol02. A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In *International Workshop on Public Key Cryptography*, pages 31–46. Springer, 2002.

BS23. W. Beullens and G. Seiler. Labrador: Compact proofs for r1cs from module-sis. In *Annual International Cryptology Conference*, pages 518–548. Springer, 2023.

BTT22. C. Boschini, A. Takahashi, and M. Tibouchi. Musig-l: Lattice-based multi-signature with single-round online phase. In *Annual international cryptology conference*, pages 276–305. Springer, 2022.

BVS+26. K. Bisht, K. A. Varshini, S. Sethi, M. Francis, and R. Kabaleeshwaran. Bmusig2: Schnorr-compatible blind multi-signatures. In *Proceedings of the ACM Asia Conference on Computer and Communications Security*, pages 326–342, 2026.

CDBT26. A. Centelles, Q. Dao, M. Badakhshan, and J. Thaler. Aerie: Efficient aggregation of falcon signatures. *Cryptology ePrint Archive*, 2026.

Cha83. D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology: Proceedings of Crypto 82*, pages 199–203. Springer, 1983.

Che23. Y. Chen. Dual-ms: Efficient lattice-based two-round multi-signature with trapdoor-free simulation. In *Annual International Cryptology Conference*, pages 716–747. Springer, 2023.

CP92. D. Chaum and T. P. Pedersen. Wallet databases with observers. In *Annual International Cryptology Conference*, pages 89–105. Springer, 1992.

DEF+19. M. Drijvers, K. Edalatnejad, B. Ford, E. Kiltz, J. Loss, G. Neven, and I. Stepanovs. On the security of two-round multi-signatures. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1084–1101. IEEE, 2019.

DOTT21. I. Damgård, C. Orlandi, A. Takahashi, and M. Tibouchi. Two-round $n$-out-of-$n$ and multi-signatures and trapdoor commitment from lattices. In *Public-Key Cryptography – PKC 2021*, Lecture Notes in Computer Science. Springer, 2021.

dPK22. R. del Pino and S. Katsumata. A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In *Annual International Cryptology Conference CRYPTO 2022*, pages 306–336. Springer, 2022.

DPKM+24. R. Del Pino, S. Katsumata, M. Maller, F. Mouhartem, T. Prest, and M.-J. Saarinen. Threshold raccoon: practical threshold signatures from standard lattice assumptions. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 219–248. Springer, 2024.

DPKN+25. R. Del Pino, S. Katsumata, G. Niot, M. Reichle, and K. Takemure. Unmasking TRaccoon: A lattice-based threshold signature with an efficient identifiable abort protocol. In *Annual International Cryptology Conference*, pages 423–456. Springer, 2025.

dPKPR24. R. del Pino, S. Katsumata, T. Prest, and M. Rossi. Raccoon: A masking-friendly signature proven in the probing model. In *Advances in Cryptology – CRYPTO 2024*, Lecture Notes in Computer Science. Springer, 2024. Full version: Cryptology ePrint Archive, Paper 2024/1291.

EKT24. T. Espitau, S. Katsumata, and K. Takemure. Two-round threshold signature from algebraic one-more learning with errors. In *Advances in Cryptology – CRYPTO 2024*, Lecture Notes in Computer Science. Springer, 2024.

Fis06. M. Fischlin. Round-optimal composable blind signatures in the common reference string model. In *Annual International Cryptology Conference*, pages 60–77. Springer, 2006.

FW24. G. Fuchsbauer and M. Wolf. Concurrently secure blind schnorr signatures. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 124–160. Springer, 2024.

GH26. M. Gorski and L. Hanzlik. Failure is not silent: Attacks on blind signatures. *Cryptology ePrint Archive*, 2026.

HKLN20. E. Hauck, E. Kiltz, J. Loss, and N. K. Nguyen. Lattice-based blind signatures, revisited. In *Advances in Cryptology – CRYPTO 2020*, Lecture Notes in Computer Science. Springer, 2020.

HMM+26. L. Hanzlik, M. Manulis, M. Mula, A. Pulval-Dady, T. Silde, and D. Slamanig. Braccoon: Concurrently secure blind lattice signatures from raccoon. *Cryptology ePrint Archive*, 2026.

JS25. C. Jeudy and O. Sanders. Improved lattice blind signatures from recycled entropy. In *Annual International Cryptology Conference*, pages 477–513. Springer, 2025.

KLR24. S. Katsumata, Y.-F. Lai, and M. Reichle. Breaking parallel ros: implication for isogeny and lattice-based blind signatures. In *IACR International Conference on Public-Key Cryptography*, pages 319–351. Springer, 2024.

KRB+24. I. Karantaidou, O. Renawi, F. Baldimtsi, N. Kamarinakis, J. Katz, and J. Loss. Blind multisignatures for anonymous tokens with decentralized issuance. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1508–1522, 2024.

LCL26. Q. Lai, C. Chen, and F.-H. Liu. DualMS 2.0: Practical lattice-based two-round Fiat-Shamir multi-signature with better efficiency. Cryptology ePrint Archive, Paper 2026/1941, 2026.

LNP22. V. Lyubashevsky, N. K. Nguyen, and M. Plancon. Efficient lattice-based blind signatures via gaussian one-time signatures. In *IACR International Conference on Public-Key Cryptography*, pages 498–527. Springer, 2022.

LOS+06. S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters. Sequential aggregate signatures and multisignatures without random oracles. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 465–485. Springer, 2006.

LS15. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.
MR07. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM journal on computing*, 37(1):267–302, 2007.
NRS21. J. Nick, T. Ruffing, and Y. Seurin. Musig2: Simple two-round schnorr multi-signatures. In *Annual International Cryptology Conference*, pages 189–221. Springer, 2021.
PWFHW23. G.-V. Policharla, B. Westerbaan, A. Faz-Hernández, and C. A. Wood. Post-quantum privacy pass via post-quantum anonymous credentials. *Cryptology ePrint Archive*, 2023.
Rüc10. M. Rückert. Lattice-based blind signatures. In *Advances in Cryptology – ASIACRYPT 2010*, Lecture Notes in Computer Science. Springer, 2010.
RY07. T. Ristenpart and S. Yilek. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 228–245. Springer, 2007.
Sch01. C. P. Schnorr. Security of blind discrete log signatures against interactive attacks. In *International Conference on Information and Communications Security*, pages 1–12. Springer, 2001.
SD3. N. Sha and N. DRAFT. standard: Permutation-based hash and extendable-output functions. *FIPS PUB*, 202:2015, 3.
Wag02. D. Wagner. A generalized birthday problem. In *Annual International Cryptology Conference*, pages 288–304. Springer, 2002.
ZT25. C. Zhu and S. Tessaro. The algebraic one-more misis problem and applications to threshold signatures. In *Annual International Cryptology Conference*, pages 548–581. Springer, 2025.

# A BMRaccoon Relations via Mixed ZK-LaBRADOR

This section describes the mixed-R1CS encoding of the complete user relation $\mathsf{L}_1$. It combines the lattice equations, SHA3-384, SHAKE256, the nonce-coefficient sampler, and the challenge sampler in one succinct lattice-based zero-knowledge proof.

The construction combines two types of constraints. SHA3-384, SHAKE256, SampleMon, and SampleInBall are expressed as a binary R1CS over $\mathbb{F}_2$, whereas the BMRaccoon, BGV, and extractable-commitment equations are lifted to bounded integer equations and embedded in an arithmetic R1CS. The two systems share a common binary encoding of their witnesses and are reduced to one dot-product relation using the mixed-R1CS reduction of LaBRADOR [BS23]. We instantiate the resulting relation with the zero-knowledge LaBRADOR toolkit of Biasioli et al. [BBL+26].

## A.1 Integer lifting

The relations $\mathsf{L}_1$ and $\mathsf{L}_{i,2}$ involve three moduli, $q$, $Q_{\mathsf{he}}$ and $Q_{\mathsf{ext}}$. Each coefficient equation $F = 0 \pmod{Q}$ is lifted to the integers and proved in the arithmetic R1CS as $\mathsf{clift}_Q(F) = Q\kappa$, where $\mathsf{clift}_Q(F)$ is the centred integer lift and the signed quotient $\kappa$ has a bounded binary decomposition of $\ell_\kappa = \lceil \log_2(2B_\kappa + 1) \rceil$ bits, with $B_\kappa = \lceil B_F/Q \rceil$ for $|\mathsf{clift}_Q(F)| \le B_F$. All lifted equations share the arithmetic modulus $M = 2^{256} + 1$, and the bounds satisfy $B_F + QB_\kappa < \frac{M}{2}$, so an equation accepted modulo $M$ holds over the integers, and hence modulo $Q$.

## A.2 Hash functions and prehashing

We instantiate the random oracles with two Keccak-based functions from FIPS 202: SHA3-384, written $\mathsf{H}$, which outputs fixed 384-bit digests, and the extendable-output function SHAKE256, which produces outputs of any requested length. $\mathsf{H}$ computes all digests: the prehashes $d_K$ and $d_W$ below, the message digest $\mu$, and the challenge seed $\rho$. SHAKE256 expands a digest into the random bits consumed by the two samplers. In $\mathsf{H}_{\mathsf{non}}$ it produces the $\ell(1 + \log_2 N)$ bits from which SampleMon builds the nonce coefficients, and in $\mathsf{H}_{\mathsf{sig}}$ the bits from which SampleInBall builds the challenge.

For a vector $\mathbf{v}$ over $R_{q'}$, $\mathsf{Ser}(\mathbf{v})$ writes each coefficient, taken in $[0, q'-1]$, as a $\lceil \log_2 q' \rceil$-bit string and concatenates these strings in a fixed order. For a list, $\mathsf{Ser}$ concatenates the serializations of its entries in order: $\mathsf{Ser}(W)$ is $\mathsf{Ser}(\mathsf{com}_1) \| \cdots \| \mathsf{Ser}(\mathsf{com}_n)$ with $q' = q$, and $\mathsf{Ser}(\hat{K})$ is $\mathsf{Ser}(\mathbf{0}) \| \mathsf{Ser}(\mathsf{vk}_1) \| \cdots \| \mathsf{Ser}(\mathsf{vk}_n)$ with $q' = q_{\nu_t}$, both in the order of the signers in $K$.

The inputs of the two oracles contain long public lists: $\mathsf{H}_{\mathsf{non}}$ takes the commitment list $W$ and the extended key list $\hat{K} = (0, K)$, and $\mathsf{H}_{\mathsf{sig}}$ takes $\hat{K}$. $\mathsf{Ser}(W)$ has $n(\ell+1)kN \lceil \log_2 q \rceil$ bits, between $6.5 \cdot 10^8$ and $1.3 \cdot 10^9$ for our parameter sets, and $\mathsf{Ser}(\hat{K})$ has $(n+1)kN \lceil \log_2 q_{\nu_t} \rceil$ bits, between $4.1 \cdot 10^6$ and

$8.4 \cdot 10^6$. Hashing them inside $\mathsf{L}_1$ would cost about $2^{34.8}$–$2^{35.8}$ quadratic constraints, far more than the rest of the circuit. We therefore prehash both lists: every party computes the public digests

$$d_K := \mathsf{H}(\mathsf{Ser}(\hat{K})), \qquad d_W := \mathsf{H}(\mathsf{Ser}(W))$$

outside the circuit, so they cost no constraints, and only these 384-bit digests enter $\mathsf{L}_1$.

### A.3 Constraints of SHA3-384 and SHAKE256

We instantiate the digests with SHA3-384 and the sampler streams with SHAKE256, both as standardized in FIPS 202 [SD3]. We choose SHA3-384 over SHA3-256 because of the digest collision term. With $Q = q_h + q_s + 1$ hash queries, the probability that any of the four digests $d_K$, $d_W$, $\mu$ and $\rho$ collides is at most $4Q^2/2^{384}$. For $q_s = 2^{32}$ this is $2^{-254}$. With 256-bit digests the same term would be $4Q^2/2^{256} = 2^{-126}$, which misses the $2^{-128}$ target.

Both functions use the permutation Keccak-$f[1600]$, whose state consists of 1600 bits and whose permutation has 24 rounds. Each round applies the five steps $\theta, \rho, \pi, \chi$, and $\iota$. The first, second, third, and fifth steps are affine over $\mathbb{F}_2$. The only nonlinear step is $\chi$.

For state bits $a, b, c \in \mathbb{F}_2$, one output bit of $\chi$ has the form

$$y = a \oplus ((\neg b) \wedge c).$$

Over $\mathbb{F}_2$, this is equivalently written as

$$b\,c = y + a + c.$$

Thus one binary rank-1 constraint suffices for each output bit of $\chi$. The affine steps before and after $\chi$ are absorbed into the linear forms of the R1CS matrices. It follows that one Keccak-$f[1600]$ permutation requires

$$N_{\mathsf{Keccak}} = 24 \cdot 1600 = 38\,400 \tag{4}$$

quadratic constraints, apart from linear input/output consistency equations.

SHA3-384 has rate 832 bits and capacity 768 bits. Let $P_{\mathsf{SHA3}}(L)$ denote the number of padded 832-bit blocks required for an $L$-bit input under the SHA3 domain suffix and the pad10*1 rule. The 384-bit digest is shorter than the rate, so no further permutation is needed for the output. SHAKE256 has rate 1088 bits and capacity 512 bits. Let $P_{\mathsf{SHAKE}}(L, o)$ denote the number of padded 1088-bit blocks required for an $L$-bit input under the SHAKE domain suffix and the pad10*1 rule, plus one permutation for every further 1088 bits of the $o$-bit output. The numbers of nonlinear constraints are therefore

$$N_{\mathsf{SHA3}}(L) = 38\,400\,P_{\mathsf{SHA3}}(L), \qquad N_{\mathsf{SHAKE}}(L, o) = 38\,400\,P_{\mathsf{SHAKE}}(L, o). \tag{5}$$

Absorption, padding, the domain suffixes, and output extraction are affine over $\mathbb{F}_2$ once the input and output lengths are fixed.

### A.4 Nonce-coefficient sampler

$\mathsf{H}_{\mathsf{non}}$ outputs the $\ell = 32$ nonce coefficients $b^{(1)}, \ldots, b^{(\ell)} \in S_b = \{\pm X^e : e \in [0, N-1]\}$. It feeds the 288-bit SHAKE256 stream into $\mathsf{SampleMon}_\ell$ (Algorithm 1), which reads $\ell$ blocks of $1 + \log_2 N = 9$ bits. Each block is parsed as a sign $\varsigma_j \in \{0, 1\}$ and an exponent $e_j \in [0, N-1]$, and the output is $b^{(j)} = (-1)^{\varsigma_j} X^{e_j}$. Since $N$ is a power of two, every block is a valid pair, so $\mathsf{SampleMon}_\ell$ never rejects and its output is uniform on $S_b^\ell$, with $|S_b| = 2N$.

The circuit takes $\varsigma_j$ and the $\log_2 N = 8$ bits of $e_j$ directly from the SHAKE256 output. For each $j \in [\ell]$, the witness contains selector bits $\delta_{j,0}, \ldots, \delta_{j,N-1}$ satisfying

$$\sum_{u=0}^{N-1} \delta_{j,u} = 1, \qquad e_j = \sum_{u=0}^{N-1} u\,\delta_{j,u},$$

and bits $\beta_{j,u}$ with

$$\beta_{j,u} = \varsigma_j\,\delta_{j,u} \qquad (u \in [0, N-1]).$$

The coefficients of $b^{(j)} = (-1)^{\varsigma_j} X^{e_j}$ are then the linear forms $b_u^{(j)} = \delta_{j,u} - 2\beta_{j,u}$. The two selector equations are linear, and the products $\beta_{j,u}$ require

$$N_{\mathsf{SM}}^{\mathsf{quad}} = \ell N = 32 \cdot 256 = 8\,192$$

quadratic constraints. Since $N$ is a power of two, $\mathsf{SampleMon}$ has no rejection condition.

**Algorithm 1** SampleMon

**function** $\textsc{SampleMon}_\ell(y \in \{0,1\}^{\ell(1+\log_2 N)})$
  **for** $j \leftarrow 1$ **to** $\ell$ **do**
    $(\varsigma_j, e_j) \leftarrow \mathrm{Parse}\big(y[(j-1)(1+\log_2 N)+1\,..\,j(1+\log_2 N)]\big)$
    $b^{(j)} \leftarrow (-1)^{\varsigma_j} X^{e_j}$
  **end for**
  **return** $(b^{(1)}, \ldots, b^{(\ell)})$
**end function**

### A.5 Challenge sampler

$\mathsf{H}_{\mathsf{sig}}$ maps $\rho$ to a challenge $c \in S_c$, a polynomial with exactly $\omega := \beta_c$ coefficients in $\{\pm 1\}$, using the sampler of [PWFHW23] as in BRaccoon (Algorithm 2). It expands $\rho$ with SHAKE256 into $\omega(\kappa_{\mathsf{idx}}+1)$ bits with $\kappa_{\mathsf{idx}} = 160$. The first $\omega\kappa_{\mathsf{idx}}$ bits are parsed as $\omega$ integers $x_1, \ldots, x_\omega \in [0, 2^{\kappa_{\mathsf{idx}}})$, and the remaining $\omega$ bits are the signs $sg_1, \ldots, sg_\omega$. At each iteration, rejection sampling makes $j$ uniform in $\{0, \ldots, i\}$, so the procedure is a truncated inside-out Fisher–Yates shuffle with uniform signs. By a union bound, its abort probability is below

$$\frac{\sum_{m=N-\omega+1}^{N} m}{2^{160}} = 13\,786/2^{160} < 2^{-146}$$

**Algorithm 2** SampleInBall using SHAKE256

**function** $\textsc{SampleInBall}(\rho)$
  $C \leftarrow 0 \in \{-1, 0, 1\}^N$
  $(x_1, \ldots, x_\omega, sg_1, \ldots, sg_\omega) \leftarrow \mathrm{Parse}(\mathrm{SHAKE256}(\rho),\ \omega\kappa_{\mathsf{idx}} + \omega)$
  **for** $a \leftarrow 1$ **to** $\omega$ **do**
    $i \leftarrow N - \omega + a - 1$
    $m \leftarrow i + 1$
    $T \leftarrow 2^{\kappa_{\mathsf{idx}}} - (2^{\kappa_{\mathsf{idx}}} \bmod m)$
    **if** $x_a \geq T$ **then**
      **abort**
    **end if**
    $j \leftarrow x_a \bmod m$
    $C_i \leftarrow C_j$
    $C_j \leftarrow (-1)^{sg_a}$
  **end for**
  **return** $C$
**end function**

For $a \in \{1, \ldots, \omega\}$, set $i_a = N - \omega + a - 1$ and $m_a = i_a + 1$. The witness contains an integer quotient $q_a$ and an index $j_a$ satisfying

$$x_a = m_a q_a + j_a, \qquad 0 \leq j_a < m_a.$$

The range of $j_a$ is encoded by selector bits $\delta_{a,0}, \ldots, \delta_{a,m_a-1}$ satisfying

$$\sum_{u=0}^{m_a-1} \delta_{a,u} = 1, \qquad j_a = \sum_{u=0}^{m_a-1} u\,\delta_{a,u}.$$

The quotient equation, the sum-to-one constraint, and the reconstruction of $j_a$ are linear in the arithmetic R1CS once the shared bits are interpreted as bounded integers. We bound

$$0 \leq q_a \leq \left\lfloor \frac{2^{\kappa_{\mathsf{idx}}} - 1}{m_a} \right\rfloor,$$

so the absolute value of the lifted quotient expression is less than $2^{\kappa_{\mathsf{idx}}+1} + m_a$. The common arithmetic modulus $M$ satisfies

$$2^{\kappa_{\mathsf{idx}}+1} + m_a < \frac{M}{2}$$

for every $a$, in addition to the modular no-wrap conditions.

The circuit then verifies the truncated Fisher–Yates update. The selector bits identify the entry $j_a$, and the circuit performs the simultaneous assignments $C_{i_a} \leftarrow C_{j_a}$ and $C_{j_a} \leftarrow (-1)^{sg_a}$. If $j_a = i_a$, the second assignment determines the final value of $C_{i_a}$. After $\omega$ iterations, the resulting polynomial has exactly $\omega$ nonzero coefficients in $\{-1, 1\}$. Using a conservative selector-and-swap accounting, the Fisher–Yates updates and sign assignments require at most

$$N_{\mathsf{FY}}^{\mathsf{quad}} = \omega(3N + 4) = 47\,092$$

quadratic constraints. Some selector-booleanity constraints are native in the binary R1CS and may disappear in a concrete implementation, so this is an upper bound.

The circuit does not enforce the rejection conditions used to remove the negligible bias of the modular reductions. If the prescribed public $\mathsf{SampleInBall}$ algorithm aborts, the resulting signature does not verify; omitting these comparisons therefore does not help produce a valid forgery.

### A.6 Total constraint count

Excluding the lifted linear commitment, rounding and BGV equations, the aggregation of the signers' nonces, quotient reconstructions, and other linear consistency constraints, the complete binary computation uses at most

$$N_{\mathsf{L}_1}^{\mathsf{quad}} = N_{\mathsf{hash}}^{\mathsf{quad}} + N_{\mathsf{SHAKE}}^{\mathsf{quad}} + N_{\mathsf{FY}}^{\mathsf{quad}} + N_{\mathsf{SM}}^{\mathsf{quad}}$$

quadratic constraints, which Table 4 evaluates. $\mathsf{L}_{i,2}$ has no binary constraints.

Here $N_{\mathrm{hash}}^{quad}$ counts the hash calls on the user's inputs: the two SHA3-384 digests $\mu := \mathsf{H}(d_K \| \mathbf{m})$, which takes 1 permutation, and $\rho := \mathsf{H}(\mu \| \mathsf{Ser}(\tilde{\mathbf{w}}))$, which takes 79 permutations, together with the SHAKE256 call that expands $d_W \| \mu$ into the bits of SampleMon, which takes 1 permutation. Hence $N_{\mathrm{hash}}^{quad} = 81 \cdot 38\,400 = 3\,110\,400$.

$N_{\mathrm{SHAKE}}^{quad}$ counts the SHAKE256 call that expands $\rho$ into the bits of SampleInBall (10 permutations), so $N_{\mathrm{SHAKE}}^{quad} = 10 \cdot 38\,400 = 384\,000$.

**Table 4.** Nonlinear constraints of $\mathsf{L}_1$ for $q_s = 2^{32}$ and $n \in \{32, 64\}$.

| Component | Quadratic constraints |
|---|---|
| $N_{\mathsf{hash}}^{\mathsf{quad}}$ | 3,110,400 |
| $N_{\mathsf{SHAKE}}^{\mathsf{quad}}$ | 384,000 |
| $N_{\mathsf{FY}}^{\mathsf{quad}}$ | 47,092 |
| $N_{\mathsf{SM}}^{\mathsf{quad}}$ | 8,192 |
| $N_{\mathsf{L}_1}^{\mathsf{quad}}$ | 3,549,684 ($\approx 2^{21.76}$) |

The arithmetic part additionally contains the lifted Raccoon and BGV equations, the extractable-commitment equations, the $3\omega$ linear index and selector equations of the challenge sampler, the $2\ell$ linear selector equations of the nonce-coefficient sampler, and the binary reconstructions of their bounded quotients. These linear constraints do not change the nonlinear count above.

## B Methodology for the Instantiation

We follow the bounding methodology of BRaccoon [HMM+26, Appendix J], and write the relations $\mathsf{L}_1$ and $\mathsf{L}_{i,2}$ in the form of their detailed user and signer relations [HMM+26, Appendix G], adapted to the multi-signer setting. All bounds used in the proof statements are chosen so that the relations can be checked by the underlying lattice proof system while preserving correctness of the protocol computations. As in [HMM+26], $D$ denotes a number of ring elements, and $B_{\mathsf{bin}}$ and $B_{\mathsf{gauss}}$ below are the $\ell_2$-bounds used in the parameter tables. We distinguish two kinds of bounds. First, global

$\ell_2$-bounds are used for short module vectors sampled from binary or Gaussian-like distributions, such as signing keys, signer nonces, the user shift and BDLOP openings. These bounds are used in the mixed ZK-LaBRADOR statements and in the corresponding SIS/MLWE parameter selection. Second, coefficient-wise $\ell_\infty$-bounds are used whenever correctness depends on the absence of modular wrap-around, for example in rounding errors, plaintext decoding, extractable commitments and BGV decryption.

Tables 5 and 6 collect the bounds and conditions used in the relations $\mathsf{L}_1$ and $\mathsf{L}_{i,2}$, following the format of [HMM+26, Tables 5 and 6].

Table 5 covers the signature and commitment relations. The $\ell_2$-bounds $\beta_{sk}$, $\beta_w$ and $\beta_{w'}$ of the signing keys, the signer nonces and the user shift are the norm bounds proved in the NIZKs and used in the MSIS and MLWE parameter selection. The infinity-norm bounds $B_{e_w}$ and $B_{e_t}$ make the two rounding relations explicit, and $\beta_{\mathsf{bms}}$ is the final verification bound. The remaining rows concern the extractable BDLOP commitment to $m_0 = (\mathbf{r}', m)$. They give the bound on its binary opening, used both in the proof and as the binding (MSIS) bound, and the coefficient bound $B_{m_0}$ of the committed message, which is dominated by the shift $\mathbf{r}'$. The last rows are the two conditions on $p_{\mathsf{ext}}$ and $Q_{\mathsf{ext}}$ that guarantee correct extraction, given the extraction noise $B_{E\gamma}$ of the simulated setup.

Table 6 covers the encryption relations. It first bounds the plaintext $\mathbf{z}_i$ recovered by the user, from the challenge, the nonce coefficients and the coefficient bounds of the signer's key and nonces; this fixes the plaintext modulus $p_{\mathsf{he}}$. It then follows the decryption noise through the protocol: the fresh noise of the user's ciphertexts, the noise after evaluating $f_{\alpha_i}$, and the flooding noise. The flooding width $\sigma_{\mathsf{fl}}$ is chosen so that the flooding coins hide the shift $B_{\mathsf{shift}}$ that evaluation induces on the encryption coins, over all $q_s(m-k)$ evaluations. The flooding coins are bounded in $\ell_2$ by $\beta_{\theta_{\mathsf{fl}}}$, which $\mathsf{L}_{i,2}$ enforces, so that $B_{\mathsf{fl}}$ holds for every accepted signer ciphertext. The last row is the no-wrap condition that fixes the ciphertext modulus $Q_{\mathsf{he}}$.

**Table 5.** Bounds used in the lattice relations and BDLOP commitments.

| Symbol | Meaning | Bound / condition |
|---|---|---|
| $\beta_{sk}$ | Bound for the signing key $\mathbf{s}_i$ | $\beta_{sk} = e^{1/4}\sigma_{sk}\sqrt{Nm}$ |
| $\beta_w$ | Bound for each signer nonce $\mathbf{r}_i^{(j)}$, $j \in [0,\ell]$ | $\beta_w = e^{1/4}\sigma_w\sqrt{Nm}$ |
| $\beta_{w'}$ | Bound for the user shift $\mathbf{r}'$ | $\beta_{w'} = e^{1/4}\sigma_{w'}\sqrt{Nm}$ |
| $B_{e_w}$ | Infinity-norm bound for the nonce-rounding error | $\|\mathbf{e}_w\|_\infty < 2^{\nu_w - 1}$ |
| $B_{e_t}$ | Infinity-norm bound for the public-key rounding error | $\|\mathbf{e}_{t_i}\|_\infty < 2^{\nu_t - 1}$ |
| $\beta_{\mathsf{bms}}$ | Final verification bound, $\|(\mathbf{z}, 2^{\nu_w}\mathbf{h})\|_2 \le \beta_{\mathsf{bms}}$ | $\beta_{\mathsf{bms}} \ge \beta_{z'} = \sqrt{2}\big(n\beta_c\beta_{sk} + n(1+\ell)\beta_w + \beta_{w'}\big) + \big((3n-1)\beta_c 2^{\nu_t - 1} + 4\cdot 2^{\nu_w - 1}\big)\sqrt{Nk}$ |
| $B_{\gamma,\mathsf{ext}}$ | Extractable BDLOP opening bound | Binary: $\sqrt{Nn_r}$ |
| $B_{\Delta\gamma,\mathsf{ext}}$ | Extractable BDLOP binding/MSIS difference bound | Binary: $B_{\gamma,\mathsf{ext}}$ |
| $B_m$ | Coefficient bound for the message $m$ | $m \in \{0,1\}^N$, so $B_m = 1$ |
| $B_{m_0}$ | Coefficient bound for the committed message $m_0 = (\mathbf{r}', m)$ | $B_{m_0} = \max\{B_m, B_\infty(\sigma_{w'})\} = B_\infty(\sigma_{w'})$ |
| $B_{E\gamma}$ | Extraction noise in the simulated BDLOP setup | $B_{E\gamma} = Nn_r$ |
| $p_{\mathsf{ext}}$ | Plaintext scale for extractable BDLOP | $p_{\mathsf{ext}} > 2B_{m_0}$ |
| $Q_{\mathsf{ext}}$ | Extractable-commitment modulus | $B_{m_0} + p_{\mathsf{ext}}B_{E\gamma} < Q_{\mathsf{ext}}/2$ |

**Table 6.** Bounds used for the MLWE-BGV encryption relations.

| Symbol | Meaning | Bound / condition |
|---|---|---|
| $B_c$ | Challenge bound | $B_c = \beta_c = \|c\|_1$ |
| $B_b$ | Nonce-coefficient bound | $B_b = \|b^{(j)}\|_1 = 1$ |
| $B_{sk,\infty}$ | Plaintext-side signing-key bound | $B_\infty(\sigma_{sk})$ |
| $B_{w,\infty}$ | Plaintext-side signer-nonce bound | $B_\infty(\sigma_w)$ |
| $B_{\mathsf{plain}}$ | Bound on decrypted plaintext $\mathbf{z}_i$ | $B_c B_{sk,\infty} + (1+\ell)B_{w,\infty}$ |
| $p_{\mathsf{he}}$ | BGV plaintext scale | $p_{\mathsf{he}} > 2B_{\mathsf{plain}}$ |
| $B_{\mathsf{enc}}$ | Fresh BGV encryption noise with binary coins | $B_{\mathsf{enc}} = 2Nk_{\mathsf{he}} + 1$ |
| $B_{\mathsf{eval}}$ | Noise after evaluating $f_{\alpha_i}$ | $N\big(B_{sk,\infty} + \ell B_{w,\infty}\big)B_{\mathsf{enc}}$ |
| $\sigma_{\mathsf{fl}}$ | Flooding width | $B_{\mathsf{shift}}\sqrt{2\pi q_s(m-k)}$ with $B_{\mathsf{shift}} = e^{1/4}N(\sigma_{sk} + \ell\sigma_w)\sqrt{(2k_{\mathsf{he}}+1)N}$ |
| $\beta_{\theta_{\mathsf{fl}}}$ | $\ell_2$ bound for the flooding coins of $\mathsf{ctx}_{r_i}$ | $e^{1/4}\sigma_{\mathsf{fl}}\sqrt{(2k_{\mathsf{he}}+1)N}$ |
| $B_{\mathsf{fl}}$ | Noise-flooding bound | $\sqrt{2Nk_{\mathsf{he}}+1}\cdot\beta_{\theta_{\mathsf{fl}}}$ |
| $B_{\mathsf{final}}$ | Final BGV decryption-noise bound | $B_{\mathsf{final}} = B_{\mathsf{eval}} + B_{\mathsf{fl}}$ |
| $Q_{\mathsf{he}}$ | BGV ciphertext modulus | $B_{\mathsf{plain}} + p_{\mathsf{he}}B_{\mathsf{final}} < Q_{\mathsf{he}}/2$ |