

DualMS 2.0: Practical Lattice-Based Two-Round Fiat-Shamir Multi-Signature with Better Efficiency

Qiqi Lai^{1,2}, Chongshen Chen¹, Feng Hao Liu³, Tianyu Zhao³, Qi Wang⁴, and Zhedong Wang^{2,4}

¹ School of Artificial Intelligence and Computer Science, Shaanxi Normal University, Xi'an, Shaanxi, China

laiqq@snnu.edu.cn, barlogs123@gmail.com

² State Key Laboratory of Cryptography and Digital Economy Security, Shandong University, Qingdao, Shandong, China

³ Washington State University, Pullman, Washington, USA

{feng-hao.liu, tianyu.zhao}@wsu.edu

⁴ School of Computer Science, Shanghai Jiao Tong University, Shanghai, China
wangamyqi@gmail.com, wzdstill@sjtu.edu.cn

Abstract. We present a more practical lattice-based two-round Fiat-Shamir multi-signature scheme that achieves a substantial reduction in signature size compared with DualMS, the state-of-the-art two-round lattice-based Fiat-Shamir multi-signature. Our main technical contribution is a new simulation technique that enables individual signatures to be sampled from distributions with smaller parameters, yielding significant concrete efficiency gains. In addition, through applying the specific modulus-rounding optimization, we can further reduce the sizes of signature and aggregated public-key.

Concretely, compared with DualMS, our scheme reduces the signature size by 62.4%, and the aggregated public-key size by 37.5%. For example, with 32 signers, our signature and aggregated public-key are approximately 8.86 KB and 4.38 KB (versus 19.72 KB and 6.94 KB in DualMS) respectively, and for 1024 signers, our signature and aggregated public-key are approximately 11.62 KB and 6.56 KB (versus 30.91 KB and 10.5 KB in DualMS).

Moreover, in order to show the practicality of our scheme, we implement our scheme in Go language using the open-source library Lattigo, and then analyze its performance on computation and communication, through both local and WAN experiments. To the best of our knowledge, our scheme achieves the smallest concrete signature sizes among all existing two-round lattice-based multi-signature schemes following Fiat-Shamir with abort paradigm. These results suggest that our scheme is practical and well-suited for real-world deployment.

Keywords: Lattice-Based Cryptography · Multi-Signature · Fiat-Shamir · Two-Round

1 Introduction

Multi-signatures (MS) are cryptographic primitives that enable a group of n users, each holding a signing key pair $(sk_i, pk_i)_{i \in [n]}$, to jointly sign the same message μ and

produce a single succinct signature,⁵ which is significantly smaller than the sum of the corresponding individual signatures. Due to this succinctness, multi-signatures are especially useful in decentralized settings such as blockchains and cryptocurrencies.

Schnorr-type multi-signatures. In the literature, numerous Schnorr-type multi-signature schemes in the random oracle model (ROM) have been proposed, such as [3,22,25,1,26]. A common characteristic of these constructions is that participants must engage in multiple rounds of interaction. Reducing this round complexity has been a primary objective in the design of efficient MS protocols.

In addition to aggregating individual signatures, another key requirement is *key aggregation* [22], which allows the public keys of all participants to be combined in a non-interactive manner. This property enables the verifier to perform verification using only the aggregated public key, thereby simplifying the verification process and reducing overhead.

Lattice-based multi-signatures. While the Schnorr-type designs offer attractive efficiency and have been well-studied, their security fundamentally relies on pre-quantum assumptions. Thus, achieving *post-quantum security* necessitates transitioning to alternative hardness assumptions that remain secure in the presence of quantum adversaries. This motivates the exploration of lattice-based constructions, which provide plausible quantum-resistant foundations.

There are many existing lattice-based MS constructions [14,21,15,5,16,9,7,8,17] that follow the Fiat-Shamir with aborts (FSwA) paradigm [19]. In particular, the FSwA approach requires the use of rejection sampling to remove the dependence of the response z in the underlying Σ -protocol on the secret key. In this sense, all of these constructions can be viewed as lattice-based counterparts to the above Schnorr-type designs.

Our Focus: two-round lattice-based MS. In this paper, we focus on two-round lattice-based FSwA multi-signatures in the plain public-key model. Among the existing works, DOTT [9] is the first lattice-based construction with a sound security proof. However, DOTT requires large concrete parameters due to its reliance on trapdoor commitments for signature simulation. To improve efficiency, MulSig-L [7] and DualMS [8] were subsequently proposed.

All three schemes (DOTT, MulSig-L, and DualMS) follow the design principles of Dilithium-G [13], where signatures are drawn from a discrete Gaussian distribution. Nevertheless, their parameters remain far from optimal. For MulSig-L, although the trapdoor commitment is eliminated, the scheme requires setting a sufficiently large standard deviation for the signature vector z due to the use of trapdoor sampling in the simulation process. Consequently, the signature size of MulSig-L remains relatively large.

DualMS further reduces the signature size by introducing a novel simulation technique, known as “dual signing simulation.” This elegant approach allows signature simulation without trapdoors, enabling much smaller Gaussian standard deviations and

⁵ In order to distinguish the individual signatures from users, we also call such a succinct signature as aggregated signature.

therefore substantially reducing signature size. Overall, in terms of signature length, DualMS currently represents the state-of-the-art: it produces signatures of approximately 31 KB for 1024 users and 19.7 KB for 32 users.

Limitations of DualMS Despite the substantial progress achieved by DualMS, several inherent challenges still limit its overall efficiency. First, the signature of DualMS includes an additional vector $\mathbf{r}$, which has (almost) the same dimension and distribution as the vector $\mathbf{z}$ appearing in the FSwA-type signatures like Dilithium-G. Second, the Gaussian standard deviations used for both $\mathbf{z}$ and $\mathbf{r}$ must be larger than a relatively large smoothing parameter. These factors together indicate considerable remaining room for reducing the signature size of DualMS. Motivated by these limitations, this work aims to address the following:

(Main Goal) Design a two-round lattice-based MS scheme that further improves the efficiency of DualMS, overcoming the above limitations and achieving substantially smaller signature size and aggregated public-key size.

1.1 Our Contributions

This work aims at the above main goal and makes the following contributions.

Contribution 1. We propose a more practical lattice-based two-round FSwA-type multi-signature scheme in the plain public-key model and the ROM, under the standard MLWE and MSIS assumptions. Our scheme supports key aggregation and can be viewed as an enhanced version of DualMS, and is therefore called DualMS 2.0. We achieve a substantial reduction in both aggregated and individual signature sizes, enabled by a new simulation technique and a specific modulus rounding optimization. Concretely, at the 128-bit security level, our scheme reduces the sizes of aggregated signature and aggregated public-key by roughly 62.4% and 37.5% respectively, compared with the original DualMS.⁶

Table 1 compares the concrete sizes of DualMS 2.0 with those of DualMS, MulSig-L, and includes the standard signature Dilithium-G as a reference. Notice that, in order to ensure fairness in the above comparison, we also present parameters for the DualMS with modulus rounding and the non-optimized version of our DualMS 2.0, since the previous schemes DualMS and MulSig-L do not mention the concrete optimized sizes obtained by modulus rounding. Even in this setting, our non-optimized version still reduces the aggregated signature size by roughly 32% and 19% compared with the original DualMS and the optimized DualMS with rounding, respectively.

Notice that, for a FSwA-type multi-signature scheme, an obvious property is that the scheme must abort if any party aborts during rejection sampling. This can be mitigated by increasing the standard deviation σ by a factor of n , which keeps the expected number of repetitions within a practical range (e.g., about $M \approx 5$). In Table 1, we have accounted for this factor in the size computations.

⁶ For multi-signature scheme, the verifier takes as input message, aggregated public-key and aggregated signature, to verify the validity of signature. Thus, the optimizations for aggregated public-key and aggregated signature are most significant.

In fact, all these existing “two-round” schemes in this field cannot be completed in exactly two rounds, including DOTT, MulSig-L, and DualMS. For exact two-round scheme, we can implicitly use the noise-flooding or hint-MLWE technique to upgrade our scheme to be FSwoA scheme. But, the cost of such upgrading process is high (e.g., the $\text{aggpk}/\text{AggSig}$ sizes will increase to 11.25/26.19 KB, compared to 4.38/8.86 KB in our current version).

Contribution 2. As a technical contribution, we introduce a new dual signing simulation technique, which allows the partial vector in the individual signature to be sampled from significantly more efficient distributions. We believe such a new simulation technique should be of independent interest, and can be applied to improve the efficiency of other cryptographic primitives.

Contribution 3. As an additional optimization technique, we identify a specific modulus rounding optimization for multi-signature schemes. Compared with the existing ones, our advantage is that the sizes of signature and aggregated public-key can be reduced significantly, while implicitly remaining the preferable properties of robustness and identifiable abort.⁷

Table 1. Comparison with related works at the 128-bit security level. $|\text{aggpk}|$ and $|\text{AggSig}|$ represent the sizes (KB) of the aggregated public-key and the aggregated signature, respectively. Moreover, we list $|\text{pk}|$, $|\text{Sig}|$, and $|\text{msg}|$ as the sizes (KB) of individual public-key, individual signature, and the first-round message, respectively. Several unavailable entries are denoted by “–”.

Scheme	MulSig-L		DualMS (without rounding)		DualMS (with rounding)		DualMS2.0 (without rounding)		DualMS2.0 (with rounding)		Dilithium-G
$ \text{aggpk} $	26.00	32.50	6.94	10.50	4.88	8.09	6.78	10.00	4.38	6.56	–
$ \text{AggSig} $	98.47	106.53	19.72	30.91	15.97	25.88	13.62	20.90	8.86	11.62	–
$ \text{pk} $	26.00	32.50	6.94	10.50	6.94	10.50	6.78	10.00	6.78	10.00	1.30
$ \text{Sig} $	–	–	22.81	34.06	22.81	34.06	12.40	17.79	12.59	18.02	2.40
$ \text{msg} $	–	–	6.94	10.50	6.94	10.50	6.78	10.00	6.78	10.00	–
# of signers	32	1024	32	1024	32	1024	32	1024	32	1024	–

1.2 Technical Overview

We now outline the key ideas behind our improvements over DualMS. We begin by reviewing the core design principles of DualMS, with a particular focus on its “dual signing simulation” technique. We then identify the main bottlenecks in the original simulation strategy and present our new simulation method, which overcomes these limitations and enables the substantial efficiency gains achieved in DualMS 2.0.

Recap of DualMS DualMS works over cyclotomic rings $R = \mathbb{Z}[X]/(f(X))$ and $R_q = \mathbb{Z}_q[X]/(f(X))$, where N is a power of 2, and $f(X) = X^N + 1$ is the $2N$ -th cyclotomic polynomial. Below, we take the participant P_1 as an example to illustrate both key

⁷ For multi-signature schemes, this allows the users to identify at least one malicious signer if a signature generation fails due to malicious behaviour.

Signature generation of DualMS

Sign 1:

1. $\mathbf{B} := [\mathbf{b}, \hat{\mathbf{B}}] \leftarrow \text{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu) \in R_q^{m \times (\ell+1)}$
2. $\tilde{\mathbf{B}} := [\mathbf{B}, \mathbf{I}] = [\mathbf{b}, \hat{\mathbf{B}}, \mathbf{I}] \in R_q^{m \times (m+\ell+1)}$
3. $\mathbf{y}_1 \leftarrow \mathcal{D}_{\sigma}^{m+\ell}, \mathbf{r}_1 := (r_{1,1}, \mathbf{r}_{1,2}^\top)^\top \leftarrow \mathcal{D}_{\sigma'}^{m+\ell+1}$
4. $\mathbf{w}_1 = \tilde{\mathbf{A}} \cdot \mathbf{y}_1 + \tilde{\mathbf{B}} \cdot \mathbf{r}_1$

 $\xleftrightarrow{\mathbf{w}_1, \mathbf{w}_2}$

Sign 2:

5. $\tilde{\mathbf{w}} := \mathbf{w}_1 + \mathbf{w}_2$
6. $c \leftarrow \text{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$
7. $\mathbf{z}_1 = \mathbf{y}_1 + a_1 \cdot c \cdot \mathbf{s}_1$
8. $\mathbf{z}_1 \leftarrow \text{RejSamp}(\mathbf{z}_1, a_1 \cdot c \cdot \mathbf{s}_1^\top, \sigma, M)$
9. Set $\text{Sig}_1 := (\mathbf{z}_1, \mathbf{r}_1)$, if $\mathbf{z}_1 \neq \perp$;
Otherwise $\text{Sig}_1 := \perp$

 $\xleftrightarrow{\text{Sig}_1, \text{Sig}_2}$

10. If $\text{Sig}_1 = \perp \vee \text{Sig}_2 = \perp$: restart
11. Output $\text{AggSig} := (c, \mathbf{z}_1 + \mathbf{z}_2, \mathbf{r}_1 + \mathbf{r}_2)$

Signature simulation of DualMS

Sign 1:

1. $\tilde{\mathbf{B}} \leftarrow \text{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu) \in R_q^{m \times \ell}$
2. $\mathbf{b} := \mathbf{t}_1 - [\tilde{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{u}_1, \tilde{\mathbf{B}} := [\mathbf{b}|\tilde{\mathbf{B}}|\mathbf{I}] \in R_q^{m \times (m+\ell+1)}$
3. $\mathbf{z}_1 \xleftarrow{\$} \mathcal{D}_{\sigma}^{m+\ell}, \mathbf{p}_1 \xleftarrow{\$} \mathcal{D}_{\sigma'}^{m+\ell+1}$
4. $\mathbf{w}_1 = \tilde{\mathbf{A}} \cdot \mathbf{z}_1 + \tilde{\mathbf{B}} \cdot \mathbf{p}_1$

 $\xleftrightarrow{\mathbf{w}_1, \mathbf{w}_2}$

Sign 2:

5. $\tilde{\mathbf{w}} := \mathbf{w}_1 + \mathbf{w}_2$
6. $c \leftarrow \text{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$
7. $\mathbf{r}_1 := \mathbf{p}_1 + a_1 \cdot c \cdot \tilde{\mathbf{u}}_1$
8. $\mathbf{r}_1 \leftarrow \text{RejSamp}(\mathbf{r}_1, a_1 \cdot c \cdot \tilde{\mathbf{u}}_1, \sigma, M)$
9. Set $\text{Sig}_1 := (\mathbf{z}_1, \mathbf{r}_1)$, if $\mathbf{r}_1 \neq \perp$;
Otherwise $\text{Sig}_1 := \perp$

 $\xleftrightarrow{\text{Sig}_1, \text{Sig}_2}$

10. If $\text{Sig}_1 = \perp \vee \text{Sig}_2 = \perp$: restart
11. Output $\text{AggSig} := (c, \mathbf{z}_1 + \mathbf{z}_2, \mathbf{r}_1 + \mathbf{r}_2)$

Fig. 1. Simple description of DualMS Protocol and its simulation in [8]. Here, we just describe the behaviors of P_1 for saving space, due to the fact that the computations and communications of P_2 are symmetrically equivalent to these of P_1 .

generation and signature generation procedures. The key pair of DualMS is $\text{sk}_1 := \mathbf{s}_1 \in R_q^{\ell+m}$, and $\text{pk}_1 := \mathbf{t}_1 = \tilde{\mathbf{A}} \cdot \mathbf{s}_1 \in R_q^m$, where $\tilde{\mathbf{A}} = [\mathbf{A}|\mathbf{I}] \in R_q^{m \times (m+\ell)}$ with an uniform matrix $\mathbf{A} \in R_q^{m \times \ell}$ and an identity matrix $\mathbf{I} \in R_q^{m \times m}$. Here, the vector $\mathbf{s}_1$ consists of ring elements with small norms. In the first round of signature generation, P_1 first queries random oracle $\text{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu)$ to get an uniform matrix $\mathbf{B} \in R_q^{m \times (\ell+1)}$, where $\tilde{\mathbf{t}}$ is the aggregated public key of all participants, and μ is the message to be signed. Then, P_1 computes and outputs a vector $\mathbf{w}_1 = \tilde{\mathbf{A}} \cdot \mathbf{y}_1 + \tilde{\mathbf{B}} \cdot \mathbf{r}_1$, where $\mathbf{y}_1 \leftarrow \mathcal{D}_{\sigma}^{m+\ell}$, $\mathbf{r}_1 \leftarrow \mathcal{D}_{\sigma'}^{m+\ell+1}$, and $\tilde{\mathbf{B}} = [\mathbf{B}|\mathbf{I}] \in R_q^{m \times (m+\ell+1)}$.

In the second round, after receiving all $\mathbf{w}_i$ from all others, P_1 adds them up to get $\tilde{\mathbf{w}} = \sum_{i \in [n]} \mathbf{w}_i$, where n denotes the number of all participants. Then, P_1 computes $\mathbf{z}_1 = \mathbf{y}_1 + a_1 \cdot c \cdot \mathbf{s}_1$ and outputs the individual signature $\text{Sig}_1 := (\mathbf{z}_1, \mathbf{r}_1)$ according to the corresponding rejection sampling algorithm. Here $c = \text{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ is the challenge for signature, and $a_1 = \text{H}_{\text{agg}}(L, \mathbf{t}_1)$ is the aggregated coefficient for $\tilde{\mathbf{t}}$, where $L = \{\mathbf{t}_1, \dots, \mathbf{t}_n\}$ is the set of public keys of all participants. For clarity of presentation, we also present the above procedure in the left hand side of Figure 1.

For security proof, we need to consider signature simulation. In [8], we program the output of H_{com} , i.e., $\mathbf{B} = \text{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu)$, such that there exists another small vector $\tilde{\mathbf{u}}_1 \in R_q^{m+\ell+1}$ satisfying $\tilde{\mathbf{B}} \cdot \tilde{\mathbf{u}}_1 = \mathbf{t}_1$. Particularly, we set $\mathbf{B} = [\mathbf{b}|\hat{\mathbf{B}}]$ with uniformly random $\hat{\mathbf{B}}$ and $\mathbf{b} = \mathbf{t}_1 - [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{u}_1$. In this case, with $\tilde{\mathbf{u}}_1 := (1, \mathbf{u}_1^\top)^\top$, it clearly holds $\tilde{\mathbf{B}} \cdot \tilde{\mathbf{u}}_1 = \mathbf{t}_1$. Thus, we can use such dual key $\tilde{\mathbf{u}}$ to simulate the valid signatures. For clarity of presentation, the details are presented in the right hand side of Figure 1.

Moreover, if reanalyzing both sides of Figure 1, we have two findings: (1) $\mathbf{z}_1$ and $\mathbf{r}_1$ follow from the distributions $\mathcal{D}_{\sigma}^{m+\ell}$ and $\mathcal{D}_{\sigma'}^{m+\ell}$ respectively⁸. And we need to set $\sigma \geq \alpha \|\mathbf{a}_1 \cdot c \cdot \mathbf{s}\|$ and $\sigma' \geq \alpha \|\mathbf{a}_1 \cdot c \cdot \tilde{\mathbf{u}}\|$, where α is determinate by the expected abort time of the rejection sampling algorithm. (2) Both Gaussian standard deviations of $\mathbf{z}$

⁸ Notice that in the practical parameter setting, we can adjust parameters such that $\sigma = \sigma'$.

and $\mathbf{r}$ are also required to be equivalent or larger than the smoothing parameter of $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ respectively. And they are upper bounded by $2N \cdot q^{m/(m+\ell)+2/(N(m+\ell))}/\sqrt{2\pi}$ and $2N \cdot q^{m/(m+\ell+1)+2/(N(m+\ell+1))}/\sqrt{2\pi}$, respectively. This is because both $\bar{\mathbf{B}} \cdot \mathbf{r}_1$ in the real signature algorithm and $\bar{\mathbf{A}} \cdot \mathbf{z}_1$ in the simulation algorithm are required to be statically close to uniform.

Taking the recommended parameters by Chen in [8] as example, we find that the lower bound from the rejection sampling is slightly larger than the upper bound of the smoothing parameter of $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$, i.e.,

$$\begin{aligned}\sigma &\geq \alpha \|a_1 \cdot c \cdot \mathbf{s}\| > 2N \cdot q^{m/(m+\ell)+2/(N(m+\ell))}/\sqrt{2\pi}, \\ \sigma' &\geq \alpha \|a_1 \cdot c \cdot \bar{\mathbf{u}}\| > 2N \cdot q^{m/(m+\ell+1)+2/(N(m+\ell+1))}/\sqrt{2\pi}.\end{aligned}$$

Thus, we conclude that the parameters given in [8] are almost “optimal”, as the bound on the smoothing parameters of $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ is a theoretically fixed value. However, from another perspective, this conclusion implies that there is little room for further optimization of DualMS, if we still use the “dual signing simulation” presented in the right-hand side of Figure 1.

Overview of Our New Simulation Technique. After observing the bottleneck of further optimizing DualMS is its “dual signing simulation”, our idea is to introduce a more efficient simulation method, such that we can remove the limitation that the Gaussian standard deviations of $\mathbf{z}$ and $\mathbf{r}$ must be larger than the smoothing parameter. Thus, we will get more space for optimizing the efficiency of DualMS.

Our new simulation method is similar to DualMS, but essentially quite different. Particularly, for $\mathbf{B} = [\mathbf{b}|\hat{\mathbf{B}}]$, we set $\mathbf{b} = (\mathbf{t}_1 - \bar{\mathbf{A}} \cdot \mathbf{x}_1)$ and $\bar{\mathbf{B}} = [\mathbf{b}|\hat{\mathbf{B}}|\mathbf{I}]$, where $\mathbf{x}_1$ is chosen from $S_\eta^{m+\ell}$, and $\hat{\mathbf{B}}$ is still a uniformly random matrix. In this case, when receiving the signature query from the adversary, we first sample $\mathbf{y}_1 \leftarrow \mathcal{D}_{\sigma'}^{\ell+m}$, $r_{1,1} \leftarrow \mathcal{D}_{\sigma'}$, and $\mathbf{r}_{1,2} \leftarrow \mathcal{D}_{\sigma'}^{\ell+m}$. Then, we compute and output $\mathbf{w}_1 = \bar{\mathbf{A}} \cdot \mathbf{y}_1 + \bar{\mathbf{B}} \cdot \begin{bmatrix} r_{1,1} \\ \mathbf{r}_{1,2} \end{bmatrix} = \bar{\mathbf{A}} \cdot \mathbf{y}_1 + r_{1,1} \cdot \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$. In the second round of signature query, with the arbitrary challenge c and a_1 , we compute

$$\begin{bmatrix} \mathbf{z}_1 \\ r_1 \end{bmatrix} := \begin{bmatrix} \mathbf{y}_1 \\ r_{1,1} \end{bmatrix} + a_1 \cdot c \cdot \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix}, \quad (1)$$

and then run the rejection sampling algorithm for such vector $(\mathbf{z}_1^\top, r_1)^\top$. If the output is non- $\perp$, we return $\text{Sig} := (\mathbf{z}_1, r_1, \mathbf{r}_{1,2})$. Otherwise, return $\perp$.

Notice that in the non- $\perp$ case, if $\mathbf{b} = (\mathbf{t}_1 - \bar{\mathbf{A}} \cdot \mathbf{x}_1)$, it always holds

$$\begin{aligned}&\bar{\mathbf{A}} \cdot \mathbf{z}_1 - a_1 \cdot c \cdot \mathbf{t}_1 + r_1 \cdot \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2} \\ &= \bar{\mathbf{A}} \cdot \mathbf{y}_1 + \bar{\mathbf{A}} \cdot a_1 \cdot c \cdot \mathbf{x}_1 - a_1 \cdot c \cdot \mathbf{t}_1 + r_1 \cdot \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2} \\ &= \bar{\mathbf{A}} \cdot \mathbf{y}_1 + \bar{\mathbf{A}} \cdot a_1 \cdot c \cdot \mathbf{x}_1 - a_1 \cdot c \cdot (\mathbf{b} + \bar{\mathbf{A}} \cdot \mathbf{x}_1) + (r_{1,1} + a_1 \cdot c) \cdot \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2} \\ &= \bar{\mathbf{A}} \cdot \mathbf{y}_1 + r_{1,1} \cdot \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2} = \mathbf{w}_1.\end{aligned} \quad (2)$$

This means the above $\text{Sig} := (\mathbf{z}_1, \mathbf{r}_1 := (r_1, \mathbf{r}_{1,2}))$ is a valid simulation signature. Next, we need to consider how to prove the indistinguishability of real and simulated signatures.

Indistinguishability of Real and Simulated Signatures. For the validity of (2), the setting of $r_1 := r_{1,1} + a_1 \cdot c$ in (1) is an essential condition. But, for the real signature in DualMS presented in Figure 1, we just have $r_1 := r_{1,1}$, which is not indistinguishable from $r_1 := r_{1,1} + a_1 \cdot c$ in general. In order to make them to be indistinguishable in an efficient parameter setting, one reasonable technique is to conduct rejection sampling for $(z_1^\top, r_1)^\top$, instead of only z_1 . Particularly, we concatenate $r_{1,1}$ to y_1 to get $(y_1^\top, r_{1,1})^\top$ as a whole randomness vector, and treat $(s_1^\top, 0)^\top$ as the whole secret key for real signature. In this case, we get

$$\begin{bmatrix} z_1 \\ r_1 \end{bmatrix} := \begin{bmatrix} y_1 \\ r_{1,1} \end{bmatrix} + a_1 \cdot c \cdot \begin{bmatrix} s_1 \\ 0 \end{bmatrix}, \quad (3)$$

and then conduct the corresponding rejection sampling. If the output is non- $\perp$, we return $\text{Sig} := (z_1, r_1 := (r_1, r_{1,2}))$. Otherwise, return $\perp$.

In order to simplify the analysis on the rejection sampling for (1) and (3), we directly set the standard deviation of $r_{1,1}$ to be the same as that of y_1 , i.e., we sample $r_{1,1} \leftarrow \mathcal{D}_\sigma$. According to the property of rejection sampling, if $\sigma \geq \alpha \|a_1 \cdot c \cdot (s_1^\top, 0)^\top\|$ and $\sigma \geq \alpha \|a_1 \cdot c \cdot (x_1^\top, 1)^\top\|$, then $(z_1^\top, r_1)^\top$ in both (1) and (3) is statistically close to $\mathcal{D}_\sigma^{\ell+m+1}$. And thus, real and simulated signatures are statistically indistinguishable.

Real Signature of DualMS 2.0

Sign 1:

1. $\mathbf{B} := [\mathbf{b}, \hat{\mathbf{B}}] \leftarrow \text{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu) \in R_q^{m \times (\ell+1)}$
2. $\tilde{\mathbf{B}} := [\mathbf{B}, \mathbf{I}] \in R_q^{m \times (m+\ell+1)}$
3. $y_1 \leftarrow \mathcal{D}_\sigma^{m+\ell}$,
 $r_1 := (r_{1,1}, r_{1,2}^\top)^\top \leftarrow \mathcal{D}_{\sigma'}^{m+\ell+1}$
4. $w_1 = \tilde{\mathbf{A}} \cdot y_1 + \tilde{\mathbf{B}} \cdot r_1$

$\xleftarrow{w_1, w_2}$

Sign 2:

5. $\tilde{w} := w_1 + w_2$
6. $c \leftarrow \text{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{w})$
7. $(z_1^\top, r_1)^\top = (y_1^\top, r_{1,1}) + a_1 \cdot c \cdot (s_1^\top, 0)^\top$
8. $(z_1^\top, r_1)^\top \leftarrow \text{RejSamp}((z_1^\top, r_1)^\top, a_1 \cdot c \cdot (s_1^\top, 0)^\top, \sigma, M)$
9. Set $\text{Sig}_1 := (z_1, r_1 := (r_1, r_{1,2}^\top)^\top)$ if $(z_1^\top, r_1)^\top \neq \perp$;
 Otherwise, set $\text{Sig}_1 := \perp$
10. If $\text{Sig}_1 = \perp \vee \text{Sig}_2 = \perp$: restart
11. Output $\text{AggSig} := (c, z_1 + z_2, r_1 + r_2)$

$\xleftarrow{\text{Sig}_1, \text{Sig}_2}$

Signature simulation of DualMS 2.0

Sign 1:

1. $\hat{\mathbf{B}} \leftarrow \text{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu) \in R_q^{m \times \ell}$
2. $\mathbf{b} := \mathbf{t}_1 - \hat{\mathbf{A}} \cdot \mathbf{x}_1$,
 $\tilde{\mathbf{B}} := [\mathbf{b} | \hat{\mathbf{B}} | \mathbf{I}] \in R_q^{m \times (m+\ell+1)}$
3. $y_1 \leftarrow \mathcal{D}_\sigma^{m+\ell}$,
 $r_1 := (r_{1,1}, r_{1,2}^\top)^\top \leftarrow \mathcal{D}_{\sigma'}^{m+\ell+1}$
4. $w_1 = \tilde{\mathbf{A}} \cdot y_1 + \tilde{\mathbf{B}} \cdot r_1$

$\xleftarrow{w_1, w_2}$

Sign 2:

5. $\tilde{w} := w_1 + w_2$
6. $c \leftarrow \text{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{w})$
7. $(z_1^\top, r_1)^\top = (y_1^\top, r_{1,1}) + a_1 \cdot c \cdot (x_1^\top, 1)^\top$
8. $(z_1^\top, r_1)^\top \leftarrow \text{RejSamp}((z_1^\top, r_1)^\top, a_1 \cdot c \cdot (x_1^\top, 1)^\top, \sigma, M)$
9. Set $\text{Sig}_1 := (z_1, r_1 := (r_1, r_{1,2}^\top)^\top)$ if $(z_1^\top, r_1)^\top \neq \perp$;
 Otherwise, set $\text{Sig}_1 := \perp$
10. If $\text{Sig}_1 = \perp \vee \text{Sig}_2 = \perp$: restart
11. Output $\text{AggSig} := (c, z_1 + z_2, r_1 + r_2)$

$\xleftarrow{\text{Sig}_1, \text{Sig}_2}$

Fig. 2. Simple description of DualMS 2.0 Protocol and our new simulation. Here, we use the box to highlight the difference between DualMS and DualMS 2.0.

Notice that for the above mentioned new simulation method, we do not need to conduct rejection sampling for the vector $r_{1,2}$ any more. This implies that the vector $r_{1,2}$ can be sampled from any distribution such that $[\hat{\mathbf{B}} | \mathbf{I}] \cdot r_{1,2}$ is statistically/computationally uniform. Clearly, if we just require $[\hat{\mathbf{B}} | \mathbf{I}] \cdot r_{1,2}$ to be computationally uniform, then $r_{1,2}$ can be sampled from $S_\eta^{m+\ell}$ while making the MLWE assumption hold. In this case, we will significantly reduce the size overhead of r_2 in $\text{Sig} = (z, r_1, r_2)$.

Up until now, an acute reader might find that if we rely on the computational uniformity of $[\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$, we will face another challenge on the rejection sampling. But we can conquer it through using rejection sampling over lattice cosets. All these details are deferred to the next paragraph.

Summing up all the above analysis, now we sample $r_{1,1} \leftarrow \mathcal{D}_{\sigma'}$, and $\mathbf{r}_{1,2} \leftarrow S_{\eta}^{\ell+m}$. For clarity of presentations, our new scheme DualMS 2.0 and new simulation method are presented in Figure 2. Notice also that, compared with the DualMS 2.0 in Figure 1, our new scheme indeed has the same number of ring multiplication in theory, with the same concrete parameters m, ℓ . This means the computation efficiency will not be negatively affected, if applying our new simulation technique.

One More Subtlety on Rejection Sampling. From the above analysis, we know that rejection sampling is the crucial technique to prove the indistinguishability of real and simulated signatures. However, up until now, we still ignore another core factor: the (conditional) distribution of $(\mathbf{y}_1^\top, r_{1,1})^\top$. For the setting of two-round multi-signature, the honest user (or the simulator) conducts rejection sampling for $(\mathbf{z}_1^\top, r_1)^\top$ in (1) (or (3)), always after the adversary receiving $\mathbf{w}_1 = \tilde{\mathbf{A}} \cdot \mathbf{y}_1 + r_{1,1} \cdot \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$. Thus, we need to consider the conditional distribution of $(\mathbf{y}_1^\top, r_{1,1})^\top$, when given $\mathbf{w}_1$.

On the one hand, if $[\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$ is set to be statistically uniform, then the conditional distribution of $(\mathbf{y}_1^\top, r_{1,1})^\top$ is statistically close to $\mathcal{D}_{\sigma}^{m+\ell+1}$. In this case, we can conduct the standard rejection sampling over ring, following [19]. However, on the other hand, if $[\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$ is set to be computationally uniform, then things are quite different. This is because the distribution of $(\mathbf{y}_1^\top, r_{1,1})^\top$ might be statistically far away from $\mathcal{D}_{\sigma}^{m+\ell+1}$, which prevents us from conducting rejection sampling directly as in [19].

In order to conquer this challenge, we can rewrite $\mathbf{w}_1 = \tilde{\mathbf{A}} \cdot \mathbf{y}_1 + r_{1,1} \cdot \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$ to get $[\tilde{\mathbf{A}}|\mathbf{b}] \cdot (\mathbf{y}_1^\top, r_{1,1})^\top = \mathbf{w}_1 - [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$. This implies that the conditional distribution of $(\mathbf{y}_1^\top, r_{1,1})^\top$ follows the discrete Gaussian over lattice coset $\Lambda_{\mathbf{w}_1}^\perp[\tilde{\mathbf{A}}|\mathbf{b}]$, where $\mathbf{w}_1' = \mathbf{w}_1 - [\hat{\mathbf{B}}|\mathbf{I}] \cdot \mathbf{r}_{1,2}$. In this case, we can leverage the more general rejection sampling over lattice cosets for $(\mathbf{z}_1^\top, r_1)^\top$, following [7].

Further Optimization through Modulus Rounding. For lattice-based Fiat-Shamir type signature schemes, modulus rounding is a widely used technique to optimize public-key and signature sizes, just as standard signature scheme Dilithium-G, threshold signature schemes T-Raccoon[10], Ringtail[6], Finally[27]. However, it is tricky to apply such optimization approach to the setting of multi-signature scheme.

For distributed signature schemes, we always need to consider the properties like robustness or identifiable abort. For lattice-based FSwa-type multi-signature schemes without applying modulus rounding techniques, such as schemes in [9,7,8], these two desired properties inherently hold. This is because their individual signatures can be verified for correctness and proven to be unforgeable in a clear way. Up until now, as far as we know, there is only one three-round multi-signature scheme in [16] that has explicitly employed modulus rounding techniques. But this scheme loses the desired robustness or identifiable abort properties. Thus, from a technical perspective, we need to consider: *How to apply modulus rounding to reduce the size overhead for two-*

round multi-signature schemes, without affecting the robustness and identifiable abort properties.

We believe that there should not be any essential technical obstacles for the above question. But, it is still necessary to notice and address it, if we want to deploy lattice-based two-round multi-signature schemes in practical scenarios.

In this paper, our optimization strategy for the above question is that of applying modulus rounding only to the aggregated public-key and the aggregated signature, rather than applying to user's public-key and individual signature as in [16].⁹ Particularly, each user still uses the originally individual public-key $\text{pk} := \mathbf{t} \in R_q^m$ in the two round signature generation procedures. This means that the first round message $\text{msg} := \mathbf{w} \in R_q^m$ and individual signature $\text{Sig}_i := (\mathbf{z}_i, r_i, \mathbf{r}_{i,2}) \in R^{2m+2\ell+1}$ remain to be the same form, where $\mathbf{z}_i = (\mathbf{z}_{i,1}, \mathbf{z}_{i,2}) \in R^\ell \times R^m$, $r_i \in R$ and $\mathbf{r} = (\mathbf{r}_{i,2,1}, \mathbf{r}_{i,2,2}) \in R^\ell \times R^m$. During the signature aggregation algorithm, we obtain the aggregated signature $\text{AggSig} := (\tilde{\mathbf{z}}, \tilde{r}, \tilde{\mathbf{r}}) \in R^{2\ell+1}$, where $\tilde{\mathbf{z}} = \sum_{i \in [n]} \mathbf{z}_{i,1}$, $\tilde{r} = \sum_{i \in [n]} r_i$, and $\tilde{\mathbf{r}} = \sum_{i \in [n]} \mathbf{r}_{i,2,1}$. At the same time, the modulus rounding optimization is applied to the aggregated public-key such that $\text{aggpk} := \lfloor \tilde{\mathbf{t}} \rfloor_{q_{v_b}} \in R_{q_{v_b}}^m$, where $q_{v_b} \ll q$. Concretely, we present the optimized parameters of DualMS 2.0 (after applying modulus rounding) in Table 1.

2 Preliminaries

Notations. In this paper, $\mathbb{Z}$ and $\mathbb{R}$ denote the sets of integers and real numbers. For positive integers n, q , let $[n]$ denote the set $\{1, \dots, n\}$ and $\mathbb{Z}_q$ denote the ring of integers modulo q . We use λ to denote the security parameter, which is the implicit input for all algorithms presented in this paper. A function $f(\lambda) > 0$ is negligible and denoted by $\text{negl}(\lambda)$ if for any $c > 0$ and sufficiently large λ , $f(\lambda) < 1/\lambda^c$. A probability is called to be overwhelming if it is $1 - \text{negl}(\lambda)$. A column vector is denoted by a bold lower case letter (e.g., $\mathbf{x}$). A matrix is denoted by a bold upper case letter (e.g., $\mathbf{A}$), and its transposition is denoted by $\mathbf{A}^\top$. Let $R = \mathbb{Z}[x]/(x^N + 1)$ be a cyclotomic ring, with N be a power of 2. The norm of an element in $R_q = \mathbb{Z}_q[x]/(x^N + 1)$ will be the norm of its unique representative with coefficients in $[-(q-1)/2, (q-1)/2]$. For positive $\beta \in \mathbb{R}$, we use S_β to denote the set of all polynomials of infinity norm less than β , i.e., $S_\beta = \{a \in R \mid \|a\|_\infty \leq \beta\}$.

For a distribution or a set X , we write $x \xleftarrow{\$} X$ to denote the operation of sampling an uniformly random x according to X . If D is not uniform distribution, we use $x \leftarrow D$ to denote the operation of sampling x randomly from D . We denote as $\text{Supp}(X)$ the support of a distribution X . For two distributions X, Y , we let $\text{SD}(X, Y) = (1/2) \cdot \sum_{x \in \Omega} |\Pr[X = x] - \Pr[Y = x]|$ denote their statistical distance, where $\Omega = \text{Supp}(X) \cup \text{Supp}(Y)$. We write $X \stackrel{s}{\approx} Y$ to mean that they are statistically close, and $X \stackrel{c}{\approx} Y$ to say that they are computationally indistinguishable.

⁹ We can also optimize our DualMS 2.0 just as in [16]. But this will make our scheme lose the robustness or identifiable abort properties, which is not preferable for the setting of distributed signatures.

2.1 The Rényi Divergence

Definition 2.1 (Rényi Divergence). For any two discrete probability distributions P and Q such that $\text{Supp}(P) \subseteq \text{Supp}(Q)$, the Rényi divergence of infinite order between P and Q is defined as

$$\text{RD}_\infty(P\|Q) = \sup_{x \in \text{Supp}(P)} \frac{\Pr[P = x]}{\Pr[Q = x]} \in [1, +\infty].$$

The properties of Rényi Divergence are listed as follows.

Lemma 2.2 (Adapted from [11,4]). Let X and Y be two random variables with probability distributions P_X and P_Y such that $\text{Supp}(P_X) \subseteq \text{Supp}(P_Y)$. The following holds:

- **Log. Positivity:** $\text{RD}_\infty(P_X\|P_Y) \geq \text{RD}_\infty(P_X\|P_X) = 1$.
- **Data Processing Inequality:** $\text{RD}_\infty(P_{f(X)}\|P_{f(Y)}) \leq \text{RD}_\infty(P_X\|P_Y)$ for any function f , where $P_{f(Z)}$ denotes the distribution of $f(Z)$ for $Z = X$ or Y .
- **Multiplicativity:** Let $X = (X_1, X_2)$ and $Y = (Y_1, Y_2)$. Let P_{X_1} and P_{Y_1} denote the probability distribution of X_1 and Y_1 . Let $P_{X_2|X_1=x}$ and $P_{Y_2|Y_1=x}$ denote the distribution of X_2 conditioned on $X_1 = x$ as well as Y_2 conditioned on $Y_1 = x$. If X_1 and X_2 are independent and Y_1 and Y_2 are independent, then $\text{RD}_\infty(P_X\|P_Y) = \text{RD}_\infty(P_{X_1}\|P_{Y_1}) \text{RD}_\infty(P_{X_2}\|P_{Y_2})$. Otherwise

$$\text{RD}_\infty(P_X\|P_Y) \leq \text{RD}_\infty(P_{X_1}\|P_{Y_1}) \cdot \max_{x \in \text{Supp}(P_{X_1})} \text{RD}_\infty(P_{X_2|X_1=x} \| P_{Y_2|Y_1=x}).$$

- **Probability Preservation:** For any event $E \subseteq \text{Supp}(P_Y)$, $P_Y(E) \geq \frac{P_X(E)}{\text{RD}_\infty(P_X\|P_Y)}$.

2.2 Discrete Gaussian Distribution

For a ring R of degree N , we can define the discrete Gaussian distribution over it in the following way.

Definition 2.3 (Definition 4.2 in [19]).

Let $\Sigma \in \mathbb{R}^{l \times l}$ be a positive-definite symmetric matrix. The discrete Gaussian distribution $\mathcal{D}_{\Lambda, \Sigma, \mathbf{v}}$ over the lattice $\Lambda \subseteq \text{span}(\Sigma)$ with center parameter $\mathbf{v}$ and Σ is defined as

$$\mathcal{D}_{\Lambda, \Sigma, \mathbf{v}}(\mathbf{z}) = \frac{\rho_{\Sigma, \mathbf{v}}(\mathbf{z})}{\sum_{\mathbf{y} \in \Lambda} \rho_{\Sigma, \mathbf{v}}(\mathbf{y})},$$

where $\rho_{\Sigma, \mathbf{v}}(\mathbf{z}) = \exp(-(\mathbf{z} - \mathbf{v})^\top (2\Sigma)^{-1} (\mathbf{z} - \mathbf{v}))$.

If $\Sigma = \sigma^2 \cdot \mathbf{I}_l$, we write $\rho_{\sigma, \mathbf{v}}$ and $\mathcal{D}_{\Lambda, \sigma, \mathbf{v}}$. We omit $\mathbf{v}$ when it is $\mathbf{0}$. For any positive integer ℓ , the discrete Gaussian distribution over R^ℓ centered around $\mathbf{v} \in R^\ell$ with standard deviation $\sigma > 0$ is given by $\mathcal{D}_{\mathbf{v}, \sigma}^{\ell, N}(\mathbf{z}) = \frac{e^{-\|\mathbf{z} - \mathbf{v}\|^2 / 2\sigma^2}}{\sum_{\mathbf{z}' \in R^\ell} e^{-\|\mathbf{z}'\|^2 / 2\sigma^2}}$. When $\mathbf{v} = \mathbf{0}$, we just write $\mathcal{D}_\sigma^{\ell, N}$ for simplicity. Particularly, we write $\mathcal{D}_{\mathbb{Z}, \sigma}$ to denote the discrete Gaussian distribution over $\mathbb{Z}$ with standard deviation σ .

We also need to use the following facts about the discrete Gaussian distribution.

Lemma 2.4 (Lemma 4.4 in [19]). *For any positive integer ℓ and any real $\sigma > 0$, and a sample sampled from $\mathcal{D}_{\sigma}^{\ell \cdot N}$ defined as above. Then for $\mathbf{x} \leftarrow \mathcal{D}_{\sigma}^{\ell \cdot N}$, it holds $\Pr \left[\|\mathbf{x}\| > t \cdot \sigma \sqrt{\ell N} \right] \leq \left(t e^{\frac{1-t^2}{2}} \right)^{\ell N}$, where t is any constant value.*

Lemma 2.5 (Sum of Discrete Gaussian Samples, Theorem 3.3 in [23]). *Let $\mathbf{x}_i$ for $i \in [n]$ be vectors sampled independently from $\mathcal{D}_{\sigma}^m$. Suppose $\sigma \cdot \sqrt{2\pi} \geq \sqrt{2} \cdot \omega(\log m)$, then the distribution of $\sum_i \mathbf{x}_i$ is statistically close to $\mathcal{D}_{\sigma\sqrt{n}}^m$.*

2.3 q -ary Lattices and Smoothing Parameter

For any $\mathbf{A} \in R_q^{n \times m}$, let $\Lambda_q^{\perp}(\mathbf{A}) = \{\mathbf{x} \in R_q^m : \mathbf{A}\mathbf{x} \equiv 0 \pmod{q}\}$, and for any $\mathbf{u} \in R_q^n$, let $\Lambda_u^{\perp}(\mathbf{A}) = \{\mathbf{x} \in R_q^m : \mathbf{A}\mathbf{x} \equiv \mathbf{u} \pmod{q}\}$.

In the following Lemma 2.6, it shows the minimum Gaussian standard deviation of $\mathbf{x}$ that makes $[\mathbf{A} \mid \mathbf{I}]\mathbf{x}$ statistically close to the uniform distribution.

Lemma 2.6 (Corollary 7.5 in [20]). *For positive integers k and ℓ , suppose $m = \ell + k \leq \text{poly}(N)$. Let $\tilde{\mathbf{A}} = [\mathbf{A} \mid \mathbf{I}] \in R_q^{k \times m}$, where $\mathbf{A}$ is uniformly distributed over $R_q^{k \times \ell}$. Then with probability $1 - 2^{-\Omega(N)}$ over the choice of $\mathbf{A}$, the distribution of $\tilde{\mathbf{A}}\mathbf{x} \in R_q^k$, where $\mathbf{x} \leftarrow \mathcal{D}_{\sigma}^m$ with parameter $\sigma > 2N \cdot q^{k/m+2/(Nm)}$, satisfies that the probability of each of the q^{Nk} possible outcomes is in the interval $(1 \pm 2^{-\Omega(N)})q^{-Nk}$. In particular, it is with statistical distance $2^{-\Omega(N)}$ of the uniform distribution over R_q^k .*

Lemma 2.7 (Switching Lemma [7]). *Let R, N, q, k, n and σ be parameters defined as in Lemma 2.6. We consider the following two distributions $\mathcal{A}_0$ and $\mathcal{A}_1$ that output $(\mathbf{A}, \mathbf{x}, \mathbf{u})$:*

$$\begin{aligned} \mathcal{A}_0: & \mathbf{A} \xleftarrow{\$} R_q^{k \times (n-k)}; \quad \mathbf{x} \xleftarrow{\$} \mathcal{D}_{\sigma}^n; \quad \mathbf{u} = [\mathbf{A} \mid \mathbf{I}_k] \cdot \mathbf{x} \pmod{q}. \\ \mathcal{A}_1: & \mathbf{A} \xleftarrow{\$} R_q^{k \times (n-k)}; \quad \mathbf{u} \xleftarrow{\$} R_q^k; \quad \mathbf{x} \xleftarrow{\$} \mathcal{D}_{\Lambda_u^{\perp}([\mathbf{A} \mid \mathbf{I}_k]), \sigma}^n. \end{aligned}$$

Then the statistical distance between these two distributions satisfies: $\text{SD}(\mathcal{A}_0, \mathcal{A}_1) = 2^{-\Omega(N)}$.

2.4 Rejection Sampling

Lyubashevsky [19] first proposed Fiat-Shamir with abort type of signatures based on lattices, through employing rejection sampling technique to ensure the independence of the signature distribution with the secret key. Devegve et al. [11] further employ the Rényi-divergence analysis for rejection sampling, which significantly improve the efficiency of the related constructions. In order to apply rejection sampling in the setting of decentralized signatures, we need to consider $\perp$ as one explicit output, just as in [27]. To further ensure that aborted transcripts do not leak secret information, Boschini et al. [7] recently introduced a generalized rejection sampling technique over the coset of lattices, which also considers the indistinguishability even with the abort case, which is critical for the security of multi-signature schemes.

Lemma 2.8 (Rejection Sampling over Cosets[7]). Take any $\alpha, T > 0$, and $0 \leq \varepsilon \leq 1/2$. Let $\mathbf{v} \in R_q^m$ satisfy $\|\mathbf{v}\| \leq T$ and $\mathbf{A} \in R_q^{n \times m}$. Fix a parameter $\sigma \geq \max(\alpha T, \eta_\varepsilon(\Lambda_q^\perp(\mathbf{A})))$ and some t such that $t = \omega(\sqrt{\log(N)})$ and $t = o(\log(N))$. Let RejSamp^S and SimRS^S be the algorithms defined in Table 2, where we denote $f := \mathcal{D}_\sigma^m$ and $g_{\mathbf{v}} := \mathcal{D}_{\sigma, \mathbf{v}}^m$. Let S denote the coset $\Lambda_{\mathbf{w}}^\perp(\mathbf{A}) = \{\mathbf{y} \in R_q^m : \mathbf{A} \cdot \mathbf{y} = \mathbf{w}\}$, where $\mathbf{w} \in R_q^n$. f^S is the restriction of f to S , i.e., $f^S(z) = f(z)/f(S)$ if $z \in S$, and $f^S(z) = 0$ otherwise. Set

$$M := \exp\left(\frac{1}{2\alpha^2} + \frac{t}{\alpha}\right) \quad \text{and} \quad \epsilon := 2 \exp\left(-t^2/2\right).$$

By setting $v_L = \frac{4\varepsilon}{(1+\varepsilon)^2}$ and $v_U = \frac{2\varepsilon}{1-\varepsilon}$, the following properties hold:

1. $\text{SD}(\text{RejSamp}^S, \text{SimRS}^S) \leq \frac{\max\{v_L, v_U\}}{2M} + \frac{\epsilon}{2M}$, and

$$\text{RD}_\infty(\text{RejSamp}^S \parallel \text{SimRS}^S) \leq \max\left(1 + v_U, 1 + \frac{\epsilon + v_L - v_L \epsilon}{M - 1}\right).$$

2. The probability that RejSamp^S outputs a sample $\mathbf{z} \neq \perp$ is at least $\frac{(1-v_L) \cdot (1-\epsilon)}{M}$ and at most $\frac{1+v_U}{M}$.

Moreover, if $\varepsilon = 2^{-N}$, it holds

$$\text{SD}(\text{RejSamp}^S, \text{SimRS}^S) \leq \frac{\epsilon}{2M} + 2^{-\Omega(N)},$$

$$\text{RD}_\infty(\text{RejSamp}^S \parallel \text{SimRS}^S) \leq \left(1 + \frac{\epsilon}{M - 1}\right) + 2^{-\Omega(N)},$$

$$\frac{(1 - \epsilon)}{M} - 2^{-\Omega(N)} \leq \Pr[\text{RejSamp}^S \text{ outputs non-}\perp] \leq \frac{1}{M} + 2^{-\Omega(N)}.$$

Table 2. Rejection sampling over q -ary lattice cosets.

$\text{RejSamp}^S(\mathbf{v}, \mathbf{y}, f, g_{\mathbf{v}}, M)$	$\text{SimRS}^S(f, M)$
1. $\mathbf{z} = \mathbf{y} + \mathbf{v}$ ($\mathbf{y} \sim f^S$)	1. $\mathbf{z} \xleftarrow{\$} f^S$
2. Output $\mathbf{z}$ w.p. $\min\left(\frac{f(\mathbf{z})}{M \cdot g_{\mathbf{v}}(\mathbf{z})}, 1\right)$	2. Output $\mathbf{z}$ w.p. $1/M$
3. Otherwise, output $\perp$	3. Otherwise, output $\perp$

Proof. We define the normalized distributions on S as $f^S(\mathbf{z}) = f(\mathbf{z})/f(S)$ and $g_{\mathbf{v}}^S(\mathbf{z}) = g_{\mathbf{v}}(\mathbf{z})/g_{\mathbf{v}}(S)$. Define the set within the coset as $S_{\mathbf{v}}^S := \{\mathbf{z} \in S : M \cdot g_{\mathbf{v}}(\mathbf{z}) \geq f(\mathbf{z})\}$. By the properties of the lattice Gaussian distribution and the choice of parameters, we

have $f^S(S_v^S) \geq 1 - \epsilon$. Let W be the event that RejSamp outputs a sample $z \neq \perp$. The probability of this event is:

$$\Pr[W] = \sum_{z \in C} g_v^S(z) \cdot \min\left(\frac{f(z)}{M \cdot g_v(z)}, 1\right) = \sum_{z \in S_v^S} \frac{f(z)}{M \cdot g_v(S)} + \sum_{z \notin S_v^S} g_v^S(z).$$

By the definition of the smoothing parameter, for $\sigma \geq \eta_\epsilon(\Lambda^\perp(\mathbf{A}))$, the ratio of the Gaussian sums over the lattice and its cosets satisfies:

$$\frac{1 - \epsilon}{1 + \epsilon} \leq \frac{\rho_{\sigma, v}(R_q^m)}{\rho_\sigma(R_q^m)} \leq 1 \quad \text{and} \quad \frac{1 - \epsilon}{1 + \epsilon} \leq \frac{\rho_\sigma(S)}{\rho_{\sigma, v}(S)} \leq \frac{1 + \epsilon}{1 - \epsilon}.$$

By setting $\nu_L = \frac{4\epsilon}{(1+\epsilon)^2}$ and $\nu_U = \frac{2\epsilon}{1-\epsilon}$, we have

$$1 - \nu_L \leq \frac{f(S)}{g_v(S)} \leq 1 + \nu_U. \quad (4)$$

Then, we obtain the upper bound:

$$\Pr[W] \leq \sum_{z \in C} \frac{f(z)}{M \cdot g_v(S)} = \frac{f(S)}{M \cdot g_v(S)} \leq \frac{1 + \nu_U}{M}.$$

Similarly, by restricting the sum to the set S_v^S , we obtain the lower bound:

$$\Pr[W] \geq \sum_{z \in S_v^S} \frac{f(z)}{M \cdot g_v(S)} = \frac{f(S)}{M \cdot g_v(S)} \cdot \sum_{z \in S_v^S} f^S(z) \geq \frac{(1 - \nu_L) \cdot (1 - \epsilon)}{M}.$$

Moreover, the statistical distance between RejSamp^S and SimRS^S is:

$$\begin{aligned} & \text{SD}(\text{RejSamp}^S, \text{SimRS}^S) \\ &= \frac{1}{2} \left(\sum_{z \in C} \left| g_v^S(z) \cdot \min\left(\frac{f(z)}{M \cdot g_v(z)}, 1\right) - \frac{f^S(z)}{M} \right| + \left| (1 - \Pr[W]) - \left(1 - \frac{1}{M}\right) \right| \right) \\ &= \frac{1}{2} \left(\sum_{z \in S_v^S} \frac{f(z)}{M} \cdot \left| \frac{1}{g_v(S)} - \frac{1}{f(S)} \right| + \sum_{z \notin S_v^S} \left| g_v^S(z) - \frac{f^S(z)}{M} \right| + \left| \frac{1}{M} - \Pr[W] \right| \right) \\ &\leq \frac{1}{2} \left(\sum_{z \in C} \frac{f(z)}{M} \cdot \left| \frac{1}{g_v(S)} - \frac{1}{f(S)} \right| + \left| \frac{1}{M} - \Pr[W] \right| \right) \\ &= \frac{1}{2} \left(\frac{1}{M} \cdot \left| \frac{f(S)}{g_v(S)} - 1 \right| + \left| \frac{1}{M} - \Pr[W] \right| \right). \end{aligned}$$

By applying the bound from Eq.4 and the bound of the probability $\Pr[W]$, we obtain:

$$\text{SD}(\text{RejSamp}^S, \text{SimRS}^S) \leq \frac{\max\{\nu_L, \nu_U\}}{2M} + \frac{\epsilon}{2M}.$$

We now bound the Rényi divergence R_∞ , which is determined by the maximum ratio between the two distributions over the support $C \cup \{\perp\}$.

$$\begin{aligned} \text{RD}_\infty(\text{RejSamp}^S \parallel \text{SimRS}^S) &= \max \left(\sup_{z \in C} \frac{g_v^S(z) \cdot \min\left(\frac{f(z)}{M \cdot g_v(z)}, 1\right)}{f^S(z)/M}, \frac{1 - \Pr[W]}{1 - 1/M} \right) \\ &= \max \left(\sup_{z \in C} \frac{\min\left(\frac{f^S(z)}{M} \cdot \frac{f(S)}{g_v(S)}, g_v^S(z)\right)}{f^S(z)/M}, \frac{1 - \Pr[W]}{1 - 1/M} \right) \\ &= \max \left(\frac{f(S)}{g_v(S)}, \frac{1 - \Pr[W]}{1 - 1/M} \right) \end{aligned}$$

By applying the bound from Eq.4, we obtain:

$$\frac{f(S)}{g_v(S)} \leq 1 + \nu_U.$$

By applying the lower bound on $\Pr[W]$, we obtain:

$$\frac{1 - \Pr[W]}{1 - 1/M} \leq \frac{1 - \frac{(1 - \nu_L) \cdot (1 - \epsilon)}{M}}{1 - 1/M} = \frac{M - (1 - \epsilon - \nu_L + \nu_L \cdot \epsilon)}{M - 1} = 1 + \frac{\epsilon + \nu_L - \nu_L \cdot \epsilon}{M - 1}.$$

So the overall Rényi divergence is bounded by the maximum of these two terms, namely:

$$\text{RD}_\infty(\text{RejSamp}^S \parallel \text{SimRS}^S) \leq \max \left(1 + \nu_U, 1 + \frac{\epsilon + \nu_L - \nu_L \cdot \epsilon}{M - 1} \right).$$

2.5 Lattices Problems

Definition 2.9 (MSIS[18]). *The MSIS $_{q,\ell,m,\beta}$ problem (over an implicit ring R) is defined as follows. Given an uniformly random matrix $\mathbf{A} \in R_q^{m \times \ell}$, output vector $\mathbf{z} \in R^{\ell+m}$ such that $[\mathbf{A} \mid \mathbf{I}] \cdot \mathbf{z} = 0$ and $0 < \|\mathbf{z}\| \leq \beta$.*

Definition 2.10 (MLWE[18]). *The decision MLWE $_{q,\ell,m,t}$ problem (over an implicit ring R) is defined as follows. For $s \xleftarrow{\$} S_t^\ell$, use $A_{q,s}$ to denote the distribution of $(\mathbf{a}, \langle \mathbf{a}, s \rangle + e) \in R_q^\ell \times R_q$, where $\mathbf{a} \xleftarrow{\$} R_q^\ell$ and $e \xleftarrow{\$} S_t$. The goal is to distinguish m samples from either $A_{q,s}$ or $\mathcal{U}(R_q^\ell, R_q)$, i.e., distinguish $(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$ from $(\mathbf{A}, \mathbf{u})$, where $\mathbf{A} \xleftarrow{\$} R_q^{m \times \ell}$, $\mathbf{u} \xleftarrow{\$} R_q^m$, $s \xleftarrow{\$} S_t^\ell$, and $\mathbf{e} \xleftarrow{\$} S_t^m$.*

Moreover, the MLWE problems defined above are the so-called ‘‘Hermite Normal Form’’ version, as its secret key and error are chosen from the identical ‘‘small’’ distribution. And such an ‘‘Hermite Normal Form’’ can be easily reduced to the standard MLWE via the approach in [2]. For standard MLWE and the above defined MSIS, it is known to be at least as hard as certain standard lattice problems over ideal lattice in the worst case [18].

Alg MS.Sign(pp, L , S , μ)

```

01:  $L := \{pk_1, \dots, pk_n\}$ ,  $S := \{sk_1, \dots, sk_n\}$ 
02: for  $i \in [n]$ :  $(msg_i, st_i) \leftarrow \text{Sign}_1(pp, L, sk_i, \mu)$ 
03: for  $i \in [n]$ :  $Sig_i \leftarrow \text{Sign}_2(pp, st_i, \{msg_i\}_{i \in [n]})$ 
04:  $\text{AggSig} \leftarrow \text{SAgg}(pp, \{Sig_i\}_{i \in [n]})$ 
05: return  $\text{AggSig}$ 

```

Fig. 3. Algorithm MS.Sign for the completeness.

2.6 Multi-Signature

Following [7], a two-round multi-signature scheme consists of a tuple of algorithms (Setup, KeyGen, KeyAgg, Sign₁, Sign₂, SAgg, Ver).

Definition 2.11 (Multi-signature Protocol). A multi-signature protocol MS consists of the following algorithms.

- $\text{Setup}(1^\lambda) \rightarrow \text{pp}$: The setup algorithm takes a security parameter λ as input and outputs public parameters pp.
- $\text{KeyGen}(\text{pp}) \rightarrow (\text{pk}, \text{sk})$: The deterministic key generation algorithm takes the public parameters pp as input and outputs a key pair (pk, sk).
- $\text{KeyAgg}(L) \rightarrow \text{aggpk}$: The public-key aggregation algorithm takes the public-key list $L := \{pk_1, \dots, pk_n\}$ as input and outputs an aggregated public-key aggpk.
- $\text{Sign}_1(\text{pp}, L, sk_i, \mu) \rightarrow (msg_i, st_i)$: The first-stage signing algorithm is run by a party P. On input the public parameters pp, a set of public keys $L := \{pk_1, \dots, pk_n\}$, its secret key sk_i , and a message μ , it outputs a protocol message msg_i and some state information st_i .
- $\text{Sign}_2(\text{pp}, st_i, \{msg_i\}_{i \in [n]}) \rightarrow Sig_i$: The second-stage signing algorithm is run by a party P. On input the public parameter pp, state st_i and a set of protocol messages $\{msg_i\}_{i \in [n]}$, it outputs an individual signature Sig_i .
- $\text{SAgg}(\text{pp}, \{Sig_i\}_{i \in [n]}) \rightarrow \text{AggSig}$: The signature aggregation algorithm takes a set of individual signatures $\{Sig_i\}_{i \in [n]}$ as input, and outputs an aggregated signature AggSig.
- $\text{Ver}(\text{pp}, \mu, \text{AggSig}, \text{aggpk}) \rightarrow b$: The verification algorithm takes the public parameter pp, a message μ , a multi-signature AggSig, and an aggregated public-key aggpk as input. It outputs $b = 1$ if the signature is valid and otherwise $b=0$.

Definition 2.12 (MS-Completeness). A two round multi-signature scheme MS is ϵ -complete if for any $n \in \text{poly}(\lambda)$, any $\text{pp} \in \text{Setup}(1^\lambda)$, any $(pk_i, sk_i) \in \text{KeyGen}(\text{pp})$ for $i \in [n]$, any message μ , and the public-key list $L := \{pk_1, \dots, pk_n\}$,

$$\Pr \left[\text{Ver}(\text{pp}, \mu, \text{AggSig}, \text{aggpk}) = 1 \mid \begin{array}{l} S = \{sk_1, \dots, sk_n\}, \\ \text{aggpk} \leftarrow \text{KeyAgg}(L), \\ \text{AggSig} \leftarrow \text{MS.Sign}(\text{pp}, L, S, \mu) \end{array} \right] \geq 1 - \text{negl}(\lambda),$$

where algorithm MS.Sign is defined in Figure 3.

Game MS-UF-CMA _{MS} ($\mathcal{A}, \lambda$)
1: $pp \leftarrow \text{Setup}(1^\lambda)$ 2: $(pk_1, sk_1) \leftarrow \text{KeyGen}(pp)$ 3: $ctr \leftarrow 0; \mathcal{S} \leftarrow \emptyset; \mathcal{Q} \leftarrow \emptyset$ 4: $(L^*, \mu^*, \text{AggSig}^*) \leftarrow \mathcal{A}^{\text{Osign}_1, \text{Osign}_2, \mathcal{H}}(pp, pk_1)$ 5: if $(pk_1 \notin L^*) \vee ((L^*, \mu^*) \in \mathcal{Q})$ then 6: return 0 7: return $\text{Ver}(\text{KeyAgg}(L^*), \mu^*, \widetilde{\text{Sig}}^*)$
Oracle Osign ₁ (L, μ)
1: $ctr \leftarrow ctr + 1$ 2: $sid \leftarrow ctr; \mathcal{S} \leftarrow \mathcal{S} \cup \{sid\}$ 3: $L \leftarrow \{pk_1, \dots, pk_n\}$ 4: $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(L, \mu)\}$ 5: $(msg_1, st_{sid}) \leftarrow \text{Sign}_1(pp, L, sk_1, \mu)$ 6: return msg_1
Oracle Osign ₂ ($sid, \{msg_i\}_{i \in [n]}$)
1: if $sid \notin \mathcal{S}$ then return $\perp$ 2: $\text{Sig}_1 \leftarrow \text{Sign}_2(pp, st_{sid}, \{msg_i\}_{i \in [n]})$ 3: $\mathcal{S} \leftarrow \mathcal{S} \setminus \{sid\}$ 4: return Sig_1

Fig. 4. The MS-UF-CMA security game.

Definition 2.13 (MS-UF-CMA Security). A two-round multi-signature scheme MS is MS-UF-CMA secure, if for any PPT adversary $\mathcal{A}$

$$\text{Adv}_{\text{MS}}^{\text{MS-UF-CMA}}(\mathcal{A}, \lambda) := \Pr [1 \leftarrow \text{MS-UF-CMA}_{\text{MS}}(\mathcal{A}, \lambda)] \leq \text{negl}(\lambda),$$

where the game MS-UF-CMA is defined in Figure 4.

2.7 Identification Schemes

Definition 2.14. A identification scheme ID is defined by a tuple (KeyGen, Prove, Verify) such that:

- KeyGen is the normal key generation algorithm which takes as input the security parameter in unary notation and outputs a pair (pk, sk) containing the publicly available verification key and the prover's secret key.
- Prove is the prover algorithm which can be split into two algorithms (Prove₁, Prove₂). For algorithm Prove₁, given sk , it produces a commitment w (first message sent to the verifier) and a state st . For algorithm Prove₂, on input (sk, w, st) and a uniformly random challenge $c \in C$ in response to commitment w , outputs an answer z .

- *Verify* is a deterministic algorithm which takes (pk, w, c, z) as input and outputs 1 to indicate acceptance or 0 otherwise.

Definition 2.15. An identification scheme is said to have soundness if it has the following properties:

- ϵ_{ls} -soundness for some $\epsilon_{ls} > 0$: for any efficient algorithm $\mathcal{A}$, we have:

$$\Pr \left[\text{Verify}(pk, w, c, z) = 1 \mid \begin{array}{l} pk \leftarrow \text{KeyGen}(1^\lambda), \\ (st, w) \leftarrow \mathcal{A}(pk) \\ c \xleftarrow{\$} C, z \leftarrow \mathcal{A}(st, c), \end{array} \right] \leq \epsilon_{ls}$$

- *special soundness*: for any efficient adversary $\mathcal{A}$, we have:

$$\Pr \left[\begin{array}{l} c_0 \neq c_1 \wedge V(pk, w, c_0, z_0) \\ = V(pk, w, c_1, z_1) = 1 \end{array} \mid \begin{array}{l} pk \leftarrow \text{KeyGen}(1^\lambda) \\ (w, c_0, z_0, c_1, z_1) \leftarrow \mathcal{A}(pk) \end{array} \right] \leq \text{negl}(\lambda).$$

To accommodate the multi-party nature of our scheme, where individual public keys are combined into a single aggregate key, we extend the notion of special soundness to the aggregated setting as follows.

Definition 2.16 (Aggregation Soundness). An identification scheme corresponding to a multi-signature scheme MS satisfies ϵ_{al} aggregation soundness, if for any efficient algorithm $\mathcal{A}$, it holds:

$$\Pr \left[\text{Verify}(\text{aggpk}, w, c, z) = 1 \mid \begin{array}{l} pk_1 \leftarrow \text{KeyGen}(1^\lambda), \\ (st, \{pk_i\}_{i \in [2, N]}) \leftarrow \mathcal{A}(pk_1) \\ \text{aggpk} \leftarrow \text{KeyAgg}(pk_1, \dots, pk_N) \\ (st', w) \leftarrow \mathcal{A}(st, \text{aggpk}) \\ c \xleftarrow{\$} C, z \leftarrow \mathcal{A}(st', c) \end{array} \right] \leq \epsilon_{al}$$

Definition 2.17 (Aggregation Special Soundness). An identification scheme corresponding to a multi-signature scheme MS satisfies aggregation special soundness, if for any efficient algorithm $\mathcal{A}$, it holds:

$$\Pr \left[\begin{array}{l} c_0 \neq c_1 \wedge \\ \text{Verify}(\text{aggpk}, \mu, (w, c_0, z_0)) = \\ \text{Verify}(\text{aggpk}, \mu, (w, c_1, z_1)) = 1 \end{array} \mid \begin{array}{l} pk_1 \leftarrow \text{KeyGen}(1^\lambda) \\ (st, \{pk_j\}_{j \in [2, n]}) \leftarrow \mathcal{A}(pk_1) \\ \text{aggpk} \leftarrow \text{KeyAgg}(pk_1, \dots, pk_N) \\ (w, c_0, z_0, c_1, z_1) \leftarrow \mathcal{A}(st, \text{aggpk}) \end{array} \right] \leq \text{negl}(\lambda)$$

Lemma 2.18 (General Forking Lemma). Let Q be the number of queries and C be a set of size $|C| > 2$. Let $\mathcal{B}$ be a randomized algorithm that, on input $(in, h_1, \dots, h_Q)$, returns an index $i \in [0, Q]$ and a side output out . Let IGen be an input generator, and let $\mathcal{F}_{\mathcal{B}}$ be the forking algorithm defined in Algorithm 2.7. Define the following probabilities:

$$\begin{aligned} acc &:= \Pr[i \geq 1 : in \leftarrow \text{IGen}(1^\lambda); h_1, \dots, h_Q \leftarrow \$C; \\ &\quad (i, out) \leftarrow \mathcal{B}(in, h_1, \dots, h_Q)] \\ frk &:= \Pr[b = 1 : in \leftarrow \text{IGen}(1^\lambda); (b, out, \hat{out}) \leftarrow \mathcal{F}_{\mathcal{B}}(in)] \end{aligned}$$

Then, the following bounds hold:

$$frk \geq acc \cdot \left(\frac{acc}{Q} - \frac{1}{|C|} \right), \quad \text{alternatively,} \quad acc \leq \frac{Q}{|C|} + \sqrt{Q \cdot frk}.$$

Algorithm 1: $\mathcal{F}_{\mathcal{B}}(\text{in})$

```

1:  $\rho \leftarrow \{0, 1\}^*$ 
2:  $h_1, \dots, h_Q \leftarrow \mathcal{C}$ 
3:  $(i, \text{out}) \leftarrow \mathcal{B}(\text{in}, h_1, \dots, h_Q; \rho)$ 
4: if  $i = 0$ , then return  $(0, \perp, \perp)$ 
5:  $\hat{h}_1, \dots, \hat{h}_Q \leftarrow \mathcal{C}$ 
6:  $(\hat{i}, \hat{\text{out}}) \leftarrow \mathcal{B}(\text{in}, h_1, \dots, h_{i-1}, \hat{h}_i, \dots, \hat{h}_Q; \rho)$ 
7: if  $i = \hat{i}$  and  $h_i \neq \hat{h}_i$ , return  $(1, \text{out}, \hat{\text{out}})$ 
8: else, return  $(0, \perp, \perp)$ 

```

3 DualMS 2.0: Practical Two-Round FSwA Multi-Signature

Before presenting our DualMS 2.0 scheme formally, we first present the preliminaries about modulus rounding, which can be used to optimize the size of our construction significantly.

3.1 Modulus Rounding

For an integer $x \in \mathbb{Z}$ and a parameter $v \in \mathbb{N} \setminus \{0\}$, we decompose x as:

$$x = 2^v \cdot x_{\top} + x_{\perp}, \quad \text{where } (x_{\top}, x_{\perp}) \in \mathbb{Z} \times [-2^{v-1}, 2^{v-1} - 1].$$

We define the rounding function $\lfloor \cdot \rfloor_v : \mathbb{Z} \rightarrow \mathbb{Z}$ such that $\lfloor x \rfloor_v = \lfloor x/2^v \rfloor = x_{\top}$, where $\lfloor \cdot \rfloor : \mathbb{R} \mapsto \mathbb{Z}$ denotes the rounding operator. With a slight overload of notation, when $q > 2^v$, we extend $\lfloor \cdot \rfloor_v$ to take inputs in $\mathbb{Z}_q$, in which case, we assume the output is an element in $\mathbb{Z}_{q_v}$, where $q_v = \lfloor q/2^v \rfloor$. Formally, we define:

$$\lfloor \cdot \rfloor_v : \mathbb{Z}_q \mapsto \mathbb{Z}_{q_v} = \mathbb{Z}_{\lfloor q/2^v \rfloor} \quad \text{s.t.} \quad \lfloor x \rfloor = \lfloor x/2^v \rfloor \bmod q_v = (x)_{\top} \bmod q_v,$$

where $x \in \mathbb{Z}_q$ is assumed to have as representative an integer in $\{1, \dots, q-1\}$. This extends coefficient-wise to vectors and matrices.

Lemma 3.1 (Rounding Offset). *Let v, q positive integers such that $q > 2^v, v \geq 4$, and set $q_v = \lfloor q/2^v \rfloor$. Moreover, assume that $q_v = \lfloor q/2^v \rfloor$, i.e. q can be decomposed as $q = 2^v \cdot q_v + q_{\perp}$ with $q_{\perp} \in [0, 2^{v-1} - 1]$. Then, for any $x \in \mathbb{Z}_q$, we have*

$$|x - 2^v \cdot \lfloor x \rfloor_v| \leq 2^v - 1 \tag{5}$$

Moreover, for any $\mathbf{x}, \delta \in \mathbb{Z}_q^n$, we have

$$\|2^v \cdot (\lfloor \mathbf{x} + \delta \rfloor_v - \lfloor \mathbf{x} \rfloor_v)\| \leq \|2^v \cdot \lfloor \delta \rfloor_v \bmod q\| + \|\mathbf{1}\| \cdot 2^v \tag{6}$$

Parameter	Description
n	Number of parties
N	Power of two defining the degree of $\Phi_{2N}(X)$
$\Phi_{2N}(X) = X^N + 1$	The $2N$ -th cyclotomic polynomial
k	Power of two with $k N$
q	Prime modulus with the form of $q = 2k + 1 \pmod{4k}$
$R = \mathbb{Z}[X]/(\Phi_{2N}(X)), R_q = \mathbb{Z}_q[X]/(\Phi_{2N}(X))$	Cyclotomic ring
$S_\eta = \{x \in R : \ x\ _\infty \leq \eta\}$	Set of small secrets
m	The height of random matrices $\mathbf{A}$
ℓ	The width of random matrices $\mathbf{A}$
σ	The standard deviation of $\mathbf{z}_i$
ν_b	The rounding-off parameter for public keys
ν_w	The rounding-off parameter for commitments
$B = \sigma\sqrt{2n\ell N} + \sqrt{mN}(\sigma\sqrt{2n} + \eta + \kappa 2^{\nu_b-1} + 2^{\nu_w})$	The upper bound of $\ (\tilde{\mathbf{z}}, \mathbf{h} \cdot 2^{\nu_w})\ $
$B'_n = \sigma\sqrt{2n\ell N}$	The upper bound of $\ \tilde{\mathbf{r}}\ $, with $\tilde{\mathbf{r}} = \sum_{i=1}^n \mathbf{r}_i$
$B''_n = \eta\sqrt{\ell \cdot N \cdot n}$	The upper bound of $\ \tilde{\mathbf{r}}\ $, with $\tilde{\mathbf{r}} = \sum_{i=1}^n \mathbf{r}_{i,2,1}$
$C = \{c \in R : \ c\ _\infty = 1 \wedge \ c\ _1 = \kappa\}$	Challenge space where $ C = \binom{N}{k} 2^\kappa$
κ	The ℓ_1 -norm of challenge $c \in C$

Table 3. Parameters of Our Rounded DualMS Scheme

3.2 Our Construction

- **Setup**(1^λ): Given the security parameter λ as input, the trusted algorithm conducts the following steps:
 1. Set the parameters just as in Table 3.
 2. Samples a random matrix $\mathbf{A} \xleftarrow{\$} R_q^{m \times \ell}$ and set $\tilde{\mathbf{A}} = [\mathbf{A} \mid \mathbf{I}] \in R_q^{m \times (\ell+m)}$.
 3. Let $H_{\text{agg}} : \{0, 1\}^* \rightarrow C$, $H_{\text{com}} : \{0, 1\}^* \rightarrow R_q^{m \times (\ell+1)}$, $H_{\text{sig}} : \{0, 1\}^* \rightarrow C$ be random oracles.
 4. Finally, the algorithm outputs $\text{pp} := \tilde{\mathbf{A}}$.
- **KeyGen**(pp): Given pp as input, the party P_1 conducts the following steps:
 1. Sample vector $\mathbf{s}_1 \xleftarrow{\$} S_\eta^{\ell+m}$ and compute $\mathbf{t}_1 = \tilde{\mathbf{A}} \cdot \mathbf{s}_1 \in R_q^m$;
 2. Output $\text{pk}_1 := \mathbf{t}_1$, $\text{sk}_1 := \mathbf{s}_1$.
- **KeyAgg**(L): Given the public-key list $L := \{\mathbf{t}_1, \dots, \mathbf{t}_n\}$ as input, the algorithm conducts the following steps:
 1. For $i \in [n]$, compute $a_i = H_{\text{agg}}(L, \mathbf{t}_i) \in C$;
 2. Compute and return the aggregated public-key $\text{aggpk} := \tilde{\mathbf{t}}_{\nu_b} = \lfloor \tilde{\mathbf{t}} \rfloor_{\nu_b} = \lfloor \sum_{i=1}^n a_i \cdot \mathbf{t}_i \rfloor_{\nu_b}$.
- **Sign**₁(pp, sk_1 , L , μ): Given pp, $\text{sk}_1 := \mathbf{s}_1$, $L := \{\mathbf{t}_1, \dots, \mathbf{t}_n\}$, and $\mu \in \mathcal{M}$, the party P_1 conducts in the following ways:
 1. $[\mathbf{b}, \mathbf{B}] = H_{\text{com}}(\tilde{\mathbf{t}}_{\nu_b}, \mu) \in R_q^{m \times (\ell+1)}$ and set $\tilde{\mathbf{B}} = [\mathbf{B} \mid \mathbf{I}] \in R_q^{m \times (\ell+m)}$, where $\tilde{\mathbf{t}}_{\nu_b} = \lfloor \tilde{\mathbf{t}} \rfloor_{\nu_b} = \lfloor \sum_{i=1}^n a_i \cdot \mathbf{t}_i \rfloor_{\nu_b}$ with $a_i = H_{\text{agg}}(L, \mathbf{t}_i) \in C$
 2. Sample $\mathbf{y}_1 \leftarrow \mathcal{D}_{\sigma}^{\ell+m}$, $\mathbf{r}_{1,1} \leftarrow \mathcal{D}_{\sigma}$, $\mathbf{r}_{1,2} \xleftarrow{\$} S_\eta^{\ell+m}$, and compute $\mathbf{w}_1 = \tilde{\mathbf{A}} \cdot \mathbf{y}_1 + \mathbf{r}_{1,1} \cdot \mathbf{b} + \tilde{\mathbf{B}} \cdot \mathbf{r}_{1,2} \in R_q^m$.
 3. Return $\text{msg}_1 := \mathbf{w}_1$, and $\text{st}_1 := (\mathbf{s}_1, \tilde{\mathbf{t}}_{\nu_b}, \mu, a_1, \mathbf{y}_1, \mathbf{r}_{1,1}, \mathbf{r}_{1,2}, \mathbf{w}_1)$.
- **Sign**₂(pp, st_1 , $\{\text{msg}_2, \dots, \text{msg}_n\}$): Given pp, $\text{st}_1 = (\mathbf{s}_1, \mathbf{t}_{\nu_b}, \mu, a_1, \mathbf{y}_1, \mathbf{r}_{1,1}, \mathbf{r}_{1,2}, \mathbf{w}_1)$, and $\{\text{msg}_2, \dots, \text{msg}_n\}$, the party P_1 can sign $\mu \in \mathcal{M}$ in the following ways:

1. Compute $\tilde{\mathbf{w}} = \sum_{i=1}^n \mathbf{w}_i$ and $\tilde{\mathbf{w}}_{v_w} := \lfloor \tilde{\mathbf{w}} \rfloor_{v_w}$, and $c := \text{H}_{\text{sig}}(\tilde{\mathbf{w}}_{v_w}, \mu, \tilde{\mathbf{t}}_{v_b})$.
 2. Set $\mathbf{v}_1 = ((a_1 \cdot c \cdot s_1)^\top, 0)^\top$.
 3. $(\mathbf{z}_1^\top, r_1)^\top \leftarrow \text{RejSamp}(\mathbf{v}_1, (\mathbf{y}_1^\top, r_{1,1})^\top, \mathcal{D}_{\sigma}^{\ell+m+1}, \mathcal{D}_{v_1, \sigma}^{\ell+m+1}, M)$.
 4. If $(\mathbf{z}_1^\top, r_1)^\top \neq \perp$, return $\text{Sig}_1 := (c, \mathbf{z}_1, r_1, \mathbf{r}_{1,2})$; Otherwise, return $\text{Sig}_1 := \perp$.
- **SAgg**(pp, aggpk, $\{\text{Sig}_i\}_{i \in [n]}$, $\{\mathbf{w}_i\}_{i \in [n]}$): Given pp, aggpk $:= \tilde{\mathbf{t}}_{v_b}$, Sig_i , and $\mathbf{w}_i$ from each party, the algorithm conducts the following steps:
1. Check if $\exists i \in [n]$, $\text{Sig}_i := \perp$, then return $\perp$.
 2. For $\text{Sig}_i := (c_i, \mathbf{z}_i, r_i, \mathbf{r}_{i,2})$, check if $\exists i \in [n]$, $c_i \neq c_1$, then return $\perp$.
 3. Compute $\tilde{\mathbf{w}} = \sum_{i=1}^n \mathbf{w}_i$, $\tilde{\mathbf{w}}_{v_w} := \lfloor \tilde{\mathbf{w}} \rfloor_{v_w}$, and $\tilde{\mathbf{r}} = \sum_{i=1}^n r_i$.
 4. For $\mathbf{z}_i = (\mathbf{z}_{i,1}^\top, \mathbf{z}_{i,2}^\top)^\top \in R^\ell \times R^m$ and $\mathbf{r}_{i,2} = (\mathbf{r}_{i,2,1}^\top, \mathbf{r}_{i,2,2}^\top)^\top \in R^\ell \times R^m$, compute $\tilde{\mathbf{z}} = \sum_{i=1}^n \mathbf{z}_{i,1} \in R^\ell$, and $\tilde{\mathbf{r}} = \sum_{i=1}^n \mathbf{r}_{i,2,1} \in R^\ell$.
 5. Compute the reconstruction target:

$$\mathbf{u} = \lfloor \mathbf{A}\tilde{\mathbf{z}} - c \cdot 2^{v_b} \cdot \tilde{\mathbf{t}}_{v_b} + \tilde{\mathbf{r}}\mathbf{b} + \mathbf{B} \cdot \tilde{\mathbf{r}} \rfloor_{v_w} \in R_{q_{v_w}}^m.$$

The hint $\mathbf{h}$ is the difference required to match $\tilde{\mathbf{w}}_{v_w}$:

$$\mathbf{h} := \tilde{\mathbf{w}}_{v_w} - \mathbf{u}.$$

6. Output $\text{AggSig} := (c_1, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \mathbf{h})$.
- **Verify**(pp, μ , AggSig, aggpk): Given pp, μ , AggSig $:= (c_1, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \mathbf{h})$ and aggpk $:= \tilde{\mathbf{t}}_{v_b}$, the algorithm performs the following steps:
1. Compute $[\mathbf{b}, \mathbf{B}] = \text{H}_{\text{com}}(\tilde{\mathbf{t}}_{v_b}, \mu) \in R_q^{m \times (\ell+1)}$.
 2. Compute $\mathbf{u} = \lfloor \mathbf{A}\tilde{\mathbf{z}} - c \cdot 2^{v_b} \cdot \tilde{\mathbf{t}}_{v_b} + \tilde{\mathbf{r}} \cdot \mathbf{b} + \mathbf{B} \cdot \tilde{\mathbf{r}} \rfloor_{v_w}$.
 3. Compute $\tilde{\mathbf{w}}_{v_w} := \mathbf{u} + \mathbf{h}$.
 4. If $\text{H}_{\text{sig}}(\tilde{\mathbf{w}}_{v_w}, \mu, \tilde{\mathbf{t}}_{v_b}) = c \wedge \|(\tilde{\mathbf{z}}, \mathbf{h} \cdot 2^{v_w})\| \leq B \wedge \|\tilde{\mathbf{r}}\| \leq B'_n \wedge \|\tilde{\mathbf{r}}\| \leq B''_n$, return 1; Otherwise return 0.

3.3 Correctness

For correctness of our DualMS 2.0 signature scheme, we show that if all individual signature and aggregates signature are generated honestly, then all four conditions in Item 4 of the algorithm Verify will hold with overwhelming probability.

The first condition holds clearly, since $\tilde{\mathbf{w}} = \sum_{i=1}^n \mathbf{w}_i$, $\tilde{\mathbf{w}}_{v_w} := \lfloor \tilde{\mathbf{w}} \rfloor_{v_w}$ and $\mathbf{h} := \tilde{\mathbf{w}}_{v_w} - \mathbf{u}$, where $\mathbf{u} = \lfloor \mathbf{A}\tilde{\mathbf{z}} - c \cdot 2^{v_b} \cdot \tilde{\mathbf{t}}_{v_b} + \tilde{\mathbf{r}} \cdot \mathbf{b} + \mathbf{B} \cdot \tilde{\mathbf{r}} \rfloor_{v_w} \in R_{q_{v_w}}^m$.

Regarding the norm bound, we know:

1. By Lemma 2.5, the sum of independent discrete Gaussian vectors $\tilde{\mathbf{z}} = \sum \mathbf{z}_{i,1}$ and $\tilde{\mathbf{r}} = \sum r_i$ is statistically close to $\mathcal{D}_{\sqrt{n}\sigma}^\ell$ and $\mathcal{D}_{\sqrt{n}\sigma}$, respectively.
2. By the tail bounds of discrete Gaussian distributions (Lemma 2.4), $\|\tilde{\mathbf{r}}\| \leq B'_n$ and $\|\tilde{\mathbf{r}}\| \leq B''_n$ hold with probability $1 - 2^{-\Omega(N)}$.
3. Since $\mathbf{r}_{i,2,1} \xleftarrow{\$} S_{\eta}^\ell$, it holds $\tilde{\mathbf{r}} = \sum_{i=1}^n \mathbf{r}_{i,2,1} \in S_{n\eta}^\ell$.

Thus, the third and fourth conditions hold with overwhelming probability. Below, we just focus on the send condition.

Let $\tilde{\mathbf{w}} = \sum_{i=1}^n \mathbf{w}_i$ be the exact aggregated commitment. Conditioned on successful rejection sampling, the relation $\mathbf{z}_i = \mathbf{y}_i + c a_i \mathbf{s}_i$ holds for all $i \in [n]$. Summing the individual unrounded commitments yields:

$$\begin{aligned} \tilde{\mathbf{w}} &= \bar{\mathbf{A}} \left(\sum_{i=1}^n \mathbf{y}_i \right) + \left(\sum_{i=1}^n r_{i,1} \right) \cdot \mathbf{b} + \bar{\mathbf{B}} \left(\sum_{i=1}^n r_{i,2} \right) \\ &= \bar{\mathbf{A}} \left(\sum_{i=1}^n \mathbf{z}_i \right) - c \bar{\mathbf{A}} \left(\sum_{i=1}^n a_i \mathbf{s}_i \right) + \tilde{\mathbf{r}} \cdot \mathbf{b} + \bar{\mathbf{B}} \left(\sum_{i=1}^n r_{i,2} \right). \end{aligned} \quad (7)$$

By the definition of the public keys, we have $\bar{\mathbf{A}} \left(\sum_{i=1}^n a_i \mathbf{s}_i \right) = \sum_{i=1}^n a_i \mathbf{t}_i = \tilde{\mathbf{t}}$. Furthermore, according to the block structure of the matrices $\bar{\mathbf{A}} = [\mathbf{A} \mid \mathbf{I}]$ and $\bar{\mathbf{B}} = [\mathbf{B} \mid \mathbf{I}]$, it holds $\bar{\mathbf{A}} \left(\sum_{i=1}^n \mathbf{z}_i \right) = \mathbf{A} \tilde{\mathbf{z}} + \sum_{i=1}^n \mathbf{z}_{i,2}$ and $\bar{\mathbf{B}} \left(\sum_{i=1}^n r_i \right) = \mathbf{B} \tilde{\mathbf{r}} + \sum_{i=1}^n r_{i,2,2}$. Putting the above relations into (7), we obtain:

$$\tilde{\mathbf{w}} = \mathbf{A} \tilde{\mathbf{z}} - c \tilde{\mathbf{t}} + \tilde{\mathbf{r}} \mathbf{b} + \mathbf{B} \tilde{\mathbf{r}} + \sum_{i=1}^n \mathbf{z}_{i,2} + \sum_{i=1}^n r_{i,2,2}. \quad (8)$$

For $\tilde{\mathbf{t}}_{v_b} = \lfloor \tilde{\mathbf{t}} \rfloor_{v_b}$, let $\delta_{\mathbf{t}} = \tilde{\mathbf{t}} - 2^{v_b} \tilde{\mathbf{t}}_{v_b}$ denote the public key rounding error. Then it holds $\|\delta_{\mathbf{t}}\|_{\infty} \leq 2^{v_b-1}$, according to (5) in Lemma 3.1. Thus, (8) implies

$$\begin{aligned} \mathbf{A} \tilde{\mathbf{z}} - c \cdot 2^{v_b} \tilde{\mathbf{t}}_{v_b} + \tilde{\mathbf{r}} \mathbf{b} + \mathbf{B} \tilde{\mathbf{r}} &= \mathbf{A} \tilde{\mathbf{z}} - c(\tilde{\mathbf{t}} - \delta_{\mathbf{t}}) + \tilde{\mathbf{r}} \mathbf{b} + \mathbf{B} \tilde{\mathbf{r}} \\ &= \tilde{\mathbf{w}} - \sum_{i=1}^n \mathbf{z}_{i,2} - \sum_{i=1}^n r_{i,2,2} + c \delta_{\mathbf{t}}. \end{aligned}$$

Consequently, we get $\mathbf{u} = \lfloor \tilde{\mathbf{w}} - \sum_{i=1}^n \mathbf{z}_{i,2} - \sum_{i=1}^n r_{i,2,2} + c \delta_{\mathbf{t}} \rfloor_{v_w}$. By the definition of the hint $\mathbf{h} = \lfloor \tilde{\mathbf{w}} \rfloor_{v_w} - \mathbf{u}$, and applying Lemma 3.1, we bound the ℓ_2 -norm of the scaled hint:

$$\begin{aligned} \|2^{v_w} \cdot \mathbf{h}\|_2 &= \left\| 2^{v_w} \cdot \left(\lfloor \tilde{\mathbf{w}} \rfloor_{v_w} - \left\lfloor \tilde{\mathbf{w}} - \sum_{i=1}^n \mathbf{z}_{i,2} - \sum_{i=1}^n r_{i,2,2} + c \delta_{\mathbf{t}} \right\rfloor_{v_w} \right) \right\|_2 \\ &\leq \left\| \sum_{i=1}^n \mathbf{z}_{i,2} + \sum_{i=1}^n r_{i,2,2} - c \delta_{\mathbf{t}} \right\|_2 + \sqrt{mN} \cdot 2^{v_w} \\ &\leq \left\| \sum_{i=1}^n \mathbf{z}_{i,2} \right\|_2 + \left\| \sum_{i=1}^n r_{i,2,2} \right\|_2 + \|c\|_1 \cdot \|\delta_{\mathbf{t}}\|_{\infty} \cdot \sqrt{mN} + \sqrt{mN} \cdot 2^{v_w}. \end{aligned}$$

According to the distribution of $\mathbf{z}_i = (\mathbf{z}_{i,1}^{\top}, \mathbf{z}_{i,2}^{\top})^{\top} \in R^{\ell} \times R^m$, together with Lemmas 2.5 and 2.4, we have $\left\| \sum_{i=1}^n \mathbf{z}_{i,2} \right\|_2 \leq \sigma \sqrt{2nmN}$, and $\|\tilde{\mathbf{z}}\|_2 \leq \sigma \sqrt{2n\ell N}$. Furthermore, due to $\|c\|_1 = \kappa$, and $\|\delta_{\mathbf{t}}\|_{\infty} \leq 2^{v_b-1}$, it holds:

$$\|2^{v_w} \cdot \mathbf{h}\|_2 \leq \sqrt{mN} \cdot \left(\sigma \sqrt{2n} + \eta + \kappa \cdot 2^{v_b-1} + 2^{v_w} \right).$$

Finally, summing up all above analysis, we conclude that

$$\|(\tilde{\mathbf{z}}, \mathbf{h} \cdot 2^{\nu_w})\|_2 \leq \sigma\sqrt{2n\ell N} + \sqrt{mN} \cdot (\sigma\sqrt{2n} + \eta + \kappa \cdot 2^{\nu_b-1} + 2^{\nu_w}) = B,$$

with overwhelming probability.

3.4 Security

Below, we focus on the security of our construction. Generally, our proof can be divided into two parts: simulation and reduction. For simulation, we introduce two procedures, Real and Sim, which represent the actual signing algorithm and its signing simulation respectively, in Table 4. Then we prove their indistinguishability in Lemma 3.2 below. During the security reduction in Theorem 3.3, the simulated signature is generated by the algorithm Sim, presented in the right-hand side of Table 4. In both of these procedures, msg denotes the output of Sign1, and Sig denotes the output of Sign2.

Private Inputs:	Public Inputs:
$s \xleftarrow{\$} S_{\eta}^{\ell+m};$ $\mathbf{x} \xleftarrow{\$} S_{\eta}^{\ell+m}$	$\bar{\mathbf{A}} = [\mathbf{A} \mid \mathbf{I}]$ where $\mathbf{A} \in R_q^{m \times \ell}, \mathbf{t} = \bar{\mathbf{A}} \cdot \mathbf{s},$ $\mathbf{b} = \mathbf{t} - \bar{\mathbf{A}} \cdot \mathbf{x} \in R_q^m$ $\bar{\mathbf{B}} = [\mathbf{B} \mid \mathbf{I}] \in R_q^{m \times (\ell+m)}$ $c \in C' = \{ab : a, b \in C\}$
Real(s) : Sign 1 : $\mathbf{y} \xleftarrow{\$} \mathcal{D}_{\sigma}^{\ell+m}, r_1 \xleftarrow{\$} \mathcal{D}_{\sigma}, \mathbf{r} \xleftarrow{\$} S_{\eta}^{\ell+m}$ $\text{msg} := \mathbf{w} = \bar{\mathbf{A}} \cdot \mathbf{y} + r_1 \cdot \mathbf{b} + \bar{\mathbf{B}} \cdot \mathbf{r}$ Sign 2 : $\mathbf{v}_1 := c \cdot \begin{bmatrix} s \\ 0 \end{bmatrix}, \begin{bmatrix} \mathbf{z} \\ r \end{bmatrix} := \begin{bmatrix} \mathbf{y} \\ r_1 \end{bmatrix} + \mathbf{v}_1$ Compute $(\mathbf{z}, r) \leftarrow \text{RejSamp}(\mathbf{v}_1, (\mathbf{y}^{\top}, r_1)^{\top}, \mathcal{D}_{\sigma}^{\ell+m+1}, \mathcal{D}_{\mathbf{v}_1, \sigma}^{\ell+m+1}, M)$	Sim(x) : Sign 1 : $\mathbf{y} \xleftarrow{\$} \mathcal{D}_{\sigma}^{\ell+m}, r \xleftarrow{\$} \mathcal{D}_{\sigma}, \mathbf{r} \xleftarrow{\$} S_{\eta}^{\ell+m}$ $\text{msg} := \mathbf{w} = \bar{\mathbf{A}} \cdot \mathbf{y} + r_1 \cdot \mathbf{b} + \bar{\mathbf{B}} \cdot \mathbf{r}$ Sign 2 : $\mathbf{v}_2 := c \cdot \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}, \begin{bmatrix} \mathbf{z} \\ r \end{bmatrix} := \begin{bmatrix} \mathbf{y} \\ r_1 \end{bmatrix} + \mathbf{v}_2$ Compute $(\mathbf{z}, r) \leftarrow \text{RejSamp}(\mathbf{v}_2, (\mathbf{y}^{\top}, r_1)^{\top}, \mathcal{D}_{\sigma}^{\ell+m+1}, \mathcal{D}_{\mathbf{v}_2, \sigma}^{\ell+m+1}, M)$
If $\mathbf{z} \neq \perp$, Return $\text{Sig} := (\mathbf{z}, r, \mathbf{r})$. Otherwise, return $\text{Sig} := \perp$	If $\mathbf{z} \neq \perp$, Return $\text{Sig} := (\mathbf{z}, r, \mathbf{r})$. Otherwise, return $\text{Sig} := \perp$

Table 4. Real and Simulation of Our Two Round FSWA Multi-Signature

Lemma 3.2. *Let integers ℓ, m satisfy $\ell + m \leq \text{poly}(N)$. Assume that the matrices $\mathbf{A}$ and $\mathbf{B}$ are chosen uniformly from $R_q^{m \times \ell}$. Let Real and Sim be the procedures with specific secret and public inputs as described in Table 4, where we denote $f := D_{\sigma}^{m+\ell+1}$ and $g_v := D_{\sigma, v}^{m+\ell+1}$. Let $T = \kappa^2 \eta \sqrt{N(\ell + m + 1)}$ be the bound on the norm of the secret vector. For any parameters $\alpha > 0$, and some t such that $t = \omega(\sqrt{\log(N)})$ and $t = o(\log(N))$,*

we choose the standard deviation σ and the repetition rate M such that:

$$\sigma \geq \max \left(\alpha T, \eta_\varepsilon \left(\Lambda_q^\perp([\bar{\mathbf{A}} \mid \mathbf{b}]) \right) / \sqrt{2\pi} \right), M := \exp \left(\frac{1}{2\alpha^2} + \frac{t}{\alpha} \right),$$

$$\epsilon := 2 \exp(-t^2/2).$$

By setting $\varepsilon = 2^{-N}$, the following properties hold:

1. The distributions of msg (i.e., $\mathbf{w}$) in Real and Sim are identical.
2. In both Real and Sim, for any $c \in C'$, it holds that

$$\frac{(1 - \epsilon)}{M} - 2^{-\Omega(N)} \leq \Pr[\text{Sig} \neq \perp] \leq \frac{1}{M} + 2^{-\Omega(N)}.$$

3. For any $c \in C'$, the statistical distance between Real and Sim is bounded as

$$\text{SD}(\text{Real}, \text{Sim}) \leq 2^{-\Omega(N)} + \frac{\epsilon}{M},$$

$$\text{RD}_\infty(\text{Real}, \text{Sim}) \leq \left(1 + \frac{\epsilon}{M-1} \right)^2 + 2^{-\Omega(N)}.$$

Proof. Claim 1 clearly holds, as $\mathbf{y}, r, \mathbf{r}$ in Real are identical to $\mathbf{y}, r, \mathbf{r}$ in Sim. Then, we analyze the distribution of the commitment $\text{msg} = \mathbf{w}$ in Real and Sim. According to the $\text{MLWE}_{q,\ell,m,\eta}$ assumption, we know

$$\mathbf{w} = \bar{\mathbf{A}}\mathbf{y} + r\mathbf{b} + \bar{\mathbf{B}}\mathbf{r}$$

is indistinguishable from

$$\mathbf{w}' = \bar{\mathbf{A}}\mathbf{y} + r\mathbf{b} + \mathbf{u},$$

where $\mathbf{u}$ is uniform over R_q^m . This is because we can establish an efficient reduction from distinguishing $\bar{\mathbf{B}}\mathbf{r}$ and $\mathbf{u}$ to distinguishing $\mathbf{w}$ and $\mathbf{w}'$. Moreover, $\mathbf{w}'$ is clearly uniform over R_q . Thus, for any efficient adversary $\mathcal{A}$, it holds $\mathbf{w}$ is computationally uniform over R_q .

Next, we prove Claim 2, through applying the result for Rejection Sampling over the lattice coset (Lemma 2.8). Before this, for $\mathbf{u} := \mathbf{w} - \bar{\mathbf{B}} \cdot \mathbf{r}$, we first define coset $\Lambda_u^\perp(\bar{\mathbf{A}})$ as

$$S = \left\{ \begin{bmatrix} \mathbf{y} \\ r \end{bmatrix} : [\bar{\mathbf{A}} \mid \mathbf{b}] \cdot \begin{bmatrix} \mathbf{y} \\ r \end{bmatrix} = \mathbf{w} - \bar{\mathbf{B}} \cdot \mathbf{r} \right\}.$$

Clearly, in Sign_2 , both Real and Sim indeed conduct rejection sampling over such coset S . Since the setting of σ is large enough to apply Lemma 2.8, we can directly conclude the upper and lower bounds for $\Pr[\text{Sig} \neq \perp]$ in Claim 2.

Finally, we prove Claim 3 via a sequence of hybrid arguments as summarized in Table 5. Before present the details, we first define the normalized distributions on S as $f^S(\cdot) = f(\cdot)/f(S) = \mathcal{D}_{\sigma,S}^{\ell+m+1}$ and $g_{v_2}^S(\cdot) = g_{v_2}(\cdot)/g_{v_2}(S) = \mathcal{D}_{v_2,\sigma,S}^{\ell+m+1}$.

Table 5. Hybrid arguments for the signature simulation. Here, we use the box to highlight the difference between Hyb_i and Hyb_{i-1} .

Hyb₀:
1. $\begin{bmatrix} y \\ r \end{bmatrix} \xleftarrow{\\$} f, r \xleftarrow{\\$} S_{\eta}^{\ell+m}$ 2. $w = \bar{\mathbf{A}} \cdot y + r \cdot \bar{\mathbf{b}} + \bar{\mathbf{B}} \cdot r$ 3. $v_2 := c \cdot \begin{bmatrix} x \\ 1 \end{bmatrix}, \begin{bmatrix} z^* \\ r^* \end{bmatrix} := \begin{bmatrix} y \\ r \end{bmatrix} + v_2$ 4. Return (z^*, r^*, r) with prob. $\min \left(1, f \left(\begin{bmatrix} z^* \\ r^* \end{bmatrix} \right) / M \cdot g_{v_2} \left(\begin{bmatrix} z^* \\ r^* \end{bmatrix} \right) \right)$ 5. Otherwise, output $\perp$
Hyb₁:
1. $\begin{bmatrix} y \\ r \end{bmatrix} \xleftarrow{\\$} f, r \xleftarrow{\\$} S_{\eta}^{\ell+m}$ 2. $w = \bar{\mathbf{A}} \cdot y + r \cdot \bar{\mathbf{b}} + \bar{\mathbf{B}} \cdot r, \boxed{\begin{bmatrix} y^* \\ r^* \end{bmatrix} \leftarrow f^S}$ 3. $v_2 := c \cdot \begin{bmatrix} x \\ 1 \end{bmatrix}, \begin{bmatrix} z^* \\ r^* \end{bmatrix} := \begin{bmatrix} y^* \\ r^* \end{bmatrix} + v_2$ 4. Return (z, r, r) with prob. $\min \left(1, f \left(\begin{bmatrix} z^* \\ r^* \end{bmatrix} \right) / M \cdot g_{v_2} \left(\begin{bmatrix} z^* \\ r^* \end{bmatrix} \right) \right)$ 5. Otherwise, output $\perp$
Hyb₂:
1. $\begin{bmatrix} y \\ r \end{bmatrix} \xleftarrow{\\$} f, r \xleftarrow{\\$} S_{\eta}^{\ell+m}$ 2. $w = \bar{\mathbf{A}} \cdot y + r \cdot \bar{\mathbf{b}} + \bar{\mathbf{B}} \cdot r$ 3. $\boxed{\begin{bmatrix} z^* \\ r^* \end{bmatrix} \leftarrow f^S}$ 4. Return (z^*, r^*, r) with prob. $\frac{1}{M}$ 5. Otherwise, output $\perp$
Hyb₃₋₅:
These hybrids follow the reverse sequence of transitions $\text{Hyb}_2 \rightarrow \text{Hyb}_1 \rightarrow \text{Hyb}_0$, except with changing $\boxed{v_2 \rightarrow v_1}$, where $v_1 := c \cdot [s^T \mid 0]^T$.

- Hyb_0 : This hybrid is identical to $\text{Sim}(x)$.
- Hyb_1 : This hybrid is identical to Hyb_0 , except that $[y^{*\top}, r^{*\top}]^\top$ is sampled from f^S . According to the Switching Lemma (Lemma 2.7), Hyb_0 and Hyb_1 are statistically close. Thus

$$\text{SD}(\text{Hyb}_0, \text{Hyb}_1) \leq 2^{-\Omega(N)}.$$

- Hyb_2 : This hybrid is identical to Hyb_1 , except that we invoke the Rejection Sampling Lemma over coset (Lemma 2.8). The distribution of $[y^{*\top}, r^{*\top}]^\top$ is transformed from the shifted discrete Gaussian $g_{v_2}^S$ to the target Gaussian f^S centered at zero, and the

output probability is fixed to $1/M$. Thus,

$$\text{SD}(\text{Hyb}_2, \text{Hyb}_3) \leq 2^{-\Omega(N)} + \frac{\epsilon}{2M},$$

and the Rényi divergence is bounded as:

$$\text{RD}_\infty(\text{Hyb}_2, \text{Hyb}_3) \leq \left(1 + \frac{\epsilon}{M-1}\right) + 2^{-\Omega(N)}.$$

- Hyb_{3-5} : These hybrids are identical to the reverse sequence $\text{Hyb}_2 \rightarrow \dots \rightarrow \text{Hyb}_0$, except that the simulated target vector $\mathbf{v}_2 = [\mathbf{x}^T, 1]^T$ is substituted with the real target $\mathbf{v}_1 = [\mathbf{s}^T, 0]^T$. The hybrid Hyb_5 is identical to $\text{Real}(\mathbf{s})$. Thus, the statistical distance is bounded as

$$\text{SD}(\text{Hyb}_3, \text{Hyb}_5) = \text{SD}(\text{Hyb}_0, \text{Hyb}_2).$$

Finally, the total statistical distance is:

$$\text{SD}(\text{Real}, \text{Sim}) \leq 2^{-\Omega(N)} + \frac{\epsilon}{M}.$$

$$\text{RD}_\infty(\text{Real}, \text{Sim}) \leq \left(1 + \frac{\epsilon}{M-1}\right)^2 + 2^{-\Omega(N)}.$$

With the above simulation result in Lemma 3.2, we can prove the unforgeability of our construction through the following theorem.

Theorem 3.3. *With the parameters specified in Table 3, our scheme is MS-UF-CMA-secure under $\text{MLWE}_{q,\ell,m,\eta}$ and $\text{MSIS}_{q,m,2\ell+m+2,\gamma}$ assumptions. Concretely, for any PPT adversary $\mathcal{A}$ against MS-UF-CMA-security that makes at most Q_h queries to each random oracle and at most Q_s queries to Osign_1 and Osign_2 , there exist PPT adversaries $\mathcal{B}_1, \mathcal{B}_2$ such that*

$$\begin{aligned} \text{Adv}_{\text{MS}_2}^{\text{UF-CMA}}(\mathcal{A}) &\leq Q_h \cdot \left[\left(Q_h \cdot \left(\sqrt{\frac{Q_h}{|C|}} + \sqrt{Q_h \cdot \text{Adv}_{\text{MSIS}_{q,m,2\ell+m+2,\gamma}}(\mathcal{B}_1)} + \frac{1}{|C|} \right) \right. \right. \\ &\quad \left. \left. + \text{Adv}_{\text{MLWE}_{q,\ell,m,\eta}}(\mathcal{B}_2) + \frac{Q_h(Q_h-1)}{|C|} + \left(\frac{2}{q^{N/2}}\right)^m \right) \cdot \left(1 + \frac{\epsilon}{M-1}\right)^{2Q_s} \right. \\ &\quad \left. + (Q_h - 1) \cdot \text{Adv}_{\text{MLWE}_{q,\ell,m,\eta}}(\mathcal{B}_2) \right], \end{aligned}$$

where $\gamma \leq 8\kappa\sqrt{B^2 + B_n'^2 + B_n''^2 + n^{*2}\kappa + 2^{2\nu_b} \cdot N \cdot m}$, and n^* denotes the maximum number of duplicate key in a public key list.

Proof. The theorem can be proven through using hybrid experiments, as summarized in Figure 5.

Game $G_0 - G_4$

```

1:  $\text{pp} \leftarrow \text{Setup}(1^\lambda)$ 
2:  $i^* \xleftarrow{\$} [Q_h]$  //  $G_1$ -
3:  $\text{sk}^* := s^* \xleftarrow{\$} S_{\eta}^{\ell+m}$ 
4:  $t^* := \bar{\mathbf{A}}s^*$  //  $G_0 - G_3$ 
5:  $t^* \xleftarrow{\$} R_q^m$  //  $G_4$ -
6:  $\text{pk}^* := t^*$ 
7:  $\text{ctr} \leftarrow 0; \mathcal{S} \leftarrow \emptyset; \mathcal{Q} \leftarrow \emptyset$ 
8:  $(L^*, \mu^*, \text{AggSig}^*) \leftarrow \mathcal{A}^{\text{Osign}_1, \text{Osign}_2, \mathcal{H}}(\text{pk}^*)$ 
9: if  $(\text{pk}^* \notin L^*) \vee ((L^*, \mu^*) \in \mathcal{Q})$ 
10:   return 0
11: if  $\tilde{t} \neq \text{KeyAgg}(L^*)$  return 0 //  $G_1$ -
12: return  $\text{Verify}(\mu^*, \text{AggSig}^*, \text{KeyAgg}(L^*))$ 

```

Oracle $\mathcal{H}_{\text{com}}(\tilde{t}, \mu)$

```

1: if  $\mathcal{T}_{\text{com}}[\tilde{t}, \mu] \neq \perp$  return  $\mathcal{T}_{\text{com}}[\tilde{t}, \mu]$ 
2: if this is  $i^*$ -th query then
3:    $b \xleftarrow{\$} R_q^m$ 
4: else //  $G_2$  -
5:    $x_j \xleftarrow{\$} S_{\eta}^{\ell+m}$ 
6:    $b := (t^* - \bar{\mathbf{A}}x_j)$ 
7:    $\mathcal{T}_{\text{sk}}[\tilde{t}, \mu] := x_j$ 
8:  $\mathcal{T}_{\text{com}}[\tilde{t}, \mu] := b$ 
9: return  $b$ 

```

Oracle $\text{Osign}_1(L, \mu)$

```

1:  $\text{ctr} \leftarrow \text{ctr} + 1; \text{sid} \leftarrow \text{ctr}$ 
2:  $\mathcal{S} \leftarrow \mathcal{S} \cup \{\text{sid}\}; \mathcal{Q} \leftarrow \mathcal{Q} \cup \{(L, \mu)\}$ 
3:  $(\text{msg}_1, \text{st}_{\text{sid}}) \leftarrow \text{Sign}_1(L, \text{sk}^*, \mu)$  //  $G_0$ - $G_2$ 
4:  $(\text{msg}_1, \text{st}_{\text{sid}}) \leftarrow \text{Sim}_1(L, \mathcal{T}_{\text{sk}}, \mu)$  //  $G_3$ -
5: return  $\text{msg}_1$ 

```

Oracle $\text{Osign}_2(\text{sid}, \{\text{msg}_i\}_{i \in [n]})$

```

1: if  $\text{sid} \notin \mathcal{S}$  return  $\perp$ 
2:  $\text{Sig}_1 \leftarrow \text{Sign}_2(\text{st}_{\text{sid}}, \{\text{msg}_i\}_{i \in [n]})$  //  $G_0$ - $G_2$ 
3:  $\text{Sig}_1 \leftarrow \text{Sim}_2(\text{st}_{\text{sid}}, \{\text{msg}_i\}_{i \in [n]})$  //  $G_3$ -
4: return  $\text{Sig}_1$ 

```

Fig. 5. The sequence of hybrid games for MS-UF-CMA security. For conciseness, we omit listing the public parameters pp when specifying the inputs to the oracle. Sim_1 and Sim_2 denote Sign_1 and Sign_2 in Sim , respectively.

Particularly, for games $\mathbf{G}_0$ - $\mathbf{G}_4$, we use $\Pr[\mathbf{G}_i]$ to denote the probability of $\mathbf{G}_i$ halting with output 1.

$\mathbf{G}_0$ This is the real experiment just as defined in Figure 4. To fix notations, we first recall this game as follows.

- First, the game samples public parameters pp and a pair (pk, sk) , with $\text{pp} := \bar{\mathbf{A}} = [\mathbf{A}, \mathbf{I}]$ with $\mathbf{A} \xleftarrow{\$} R_q^{m \times \ell}$, $\text{sk}^* := s^* \xleftarrow{\$} S_{\eta}^{\ell+m}$, and $\text{pk}^* := t^* = \bar{\mathbf{A}} \cdot s^* \in R_q^m$.
- Then, the adversary $\mathcal{A}$ gets pp, pk^* , and makes at most Q_h queries to each random oracle and at most Q_s queries to Osign_1 and Osign_2 in the following way.
 - * Random oracles $\text{H}_{\text{agg}}, \text{H}_{\text{com}}, \text{H}_{\text{sig}}$ can be simulated in a lazy sampling method. For that, the game initializes three empty tables $\mathcal{T}_{\text{agg}}, \mathcal{T}_{\text{com}}$ and $\mathcal{T}_{\text{sig}}$, which map the inputs to their outputs. For example, if $\mathcal{A}$ queries $\text{H}_{\text{com}}(x)$ for any bit string $x \in \{0, 1\}^*$, the game checks if $\mathcal{T}_{\text{com}}[x]$ has been defined. If it is not defined, the game samples $\mathbf{b} \xleftarrow{\$} R_q^m$, and defines $\mathcal{T}_{\text{com}}[x] := \mathbf{b}$. Then, the game returns $\mathcal{T}_{\text{com}}[x]$.
 - * Signing oracles Osign_1 and Osign_2 . With the honestly generated secret key s^* , the game answers $\mathcal{A}$'s signature generation queries, through directly invoking the algorithms Sign_1 and Sign_2 . Particularly, in order to start a new interaction, the adversary $\mathcal{A}$ conducts a query $\text{Osign}_1(L, \mu)$, where $L = \{\text{pk}_1, \dots, \text{pk}_n\}$ is the set of public keys, and μ is the message to be signed. Here, without loss of generality, we set $\text{pk}^* = \text{pk}_1$. At the same time, (L, μ) is added into the list Q .
- Finally, $\mathcal{A}$ outputs a forgery $(L^*, \mu^*, \text{AggSig}^*)$. The game outputs 1 if and only if $\text{pk}^* \in L^*$, $(L^*, \mu^*) \notin Q$ and $\text{Verify}(\text{pp}, \mu^*, \text{AggSig}^*, \text{aggpk}^*) = 1$, with $\text{aggpk}^* := \tilde{\mathbf{t}}_{v_b}^* = \lfloor \tilde{\mathbf{t}}^* \rfloor_{v_b} \leftarrow \text{KeyAgg}(L^*)$ and $\text{Sig}^* := (c^*, z^*, r^*, r^*, h^*)$. Here, we assume that the public key pk^* is equal to pk_1 for $L^* := \{\text{pk}_1, \dots, \text{pk}_n\}$. Besides, we assume that before it outputs the forgery, $\mathcal{A}$ has queried $\text{H}_{\text{agg}}(L^*, t_j)$ for all $t_j \in L^*$, $\text{H}_{\text{com}}(\tilde{\mathbf{t}}_{v_b}^*, \mu^*)$, and $\text{H}_{\text{sig}}(\tilde{\mathbf{w}}_{v_w}^*, \mu^*, \tilde{\mathbf{t}}_{v_b}^*)$ for arbitrary $\tilde{\mathbf{w}}_{v_w}^*$, where $\tilde{\mathbf{w}}_{v_w}^* := \lfloor \tilde{\mathbf{w}}^* \rfloor_{v_w}$ and $\tilde{\mathbf{w}}^* = \sum_{i=1}^n \mathbf{w}_i^*$. By definition, we have

$$\Pr[\mathbf{G}_0] = \text{Adv}_{\text{MS}_2}^{\text{UF-CMA}}(\mathcal{A}).$$

$\mathbf{G}_1$ This experiment is identical to $\mathbf{G}_0$, except that the adversary selects the index of a H_{com} query without making any oracle queries, and its goal is to forge a multi-signature corresponding to such H_{com} query. In this case, $\mathcal{A}$ indeed conducts the selective UF-CMA experiment. Particularly, such selective experiment has the following differences from the UF-CMA one defined in Figure 4. After receiving pp and pk^* , $\mathcal{A}$ first outputs an index $i^* \in [Q_h]$, and then makes queries to random oracles and $\text{Osign}_1, \text{Osign}_2$ just as $\mathbf{G}_0$. Suppose the i^* -th H_{com} query of $\mathcal{A}$ is with respect to $(\tilde{\mathbf{t}}_{v_b}^*, \mu^*)$. $\mathcal{A}$ finally outputs a forgery $(L^*, \mu^*, \text{AggSig}^*)$. The game outputs 1 if and only if $\text{pk} \in L^*$, $(L^*, \mu^*) \notin Q$ and $\text{Verify}(\text{pp}, \mu^*, \text{AggSig}^*, \tilde{\mathbf{t}}_{v_b}^*) = 1$, with $\tilde{\mathbf{t}}_{v_b}^* = \text{KeyAgg}(L^*)$.

Clearly, we can use complexity leverage to reduce UF-CMA security to the selective one, just with Q_h security loss. Thus, it holds

$$\Pr[\mathbf{G}_0] \leq Q_h \cdot \Pr[\mathbf{G}_1].$$

G₂ This experiment is identical to **G₁**, except for the simulation of H_{com} . Notice that, without loss of generality, we assume the adversary has conducted the $H_{\text{com}}(\tilde{t}_{v_b}, \mu)$ query before it queries $\text{Osign}_1(L, \mu)$ and $H_{\text{sig}}(\tilde{w}_{v_w}, \mu, \tilde{t}_{v_b})$, where $\tilde{t}_{v_b} = \text{KeyAgg}(L)$ and $\tilde{w}_{v_w}$ is arbitrary vector. Particularly, the simulator first defines an additional empty table $\mathcal{T}_{\text{sk}}$. And then for the i^* -th H_{com} query of $\mathcal{A}$ with respect to $(\tilde{t}_{v_b}^*, \mu^*)$, the simulator still returns a random vector $\mathbf{b}^* \xleftarrow{\$} R_q^m$, and defines $\mathcal{T}_{\text{com}}[\tilde{t}_{v_b}^*, \mu^*] := \mathbf{b}^*$. But, for all $j \in [Q_h] \setminus \{i^*\}$, the j -th H_{com} query of $\mathcal{A}$ is answered in a different way. Particularly, when getting $(\tilde{t}_{v_b, j}, \mu_j)$ from $\mathcal{A}$, the simulator conducts as follows:

1. Sample $\mathbf{x}_j \xleftarrow{\$} S_{\eta}^{\ell+k}$.
2. Compute and return $\mathbf{b}_j = (\mathbf{t} - \tilde{\mathbf{A}} \cdot \mathbf{x}_j) \in R_q^k$.
3. Define $\mathcal{T}_{\text{com}}[\tilde{t}_{v_b, j}, \mu_j] := \mathbf{b}_j$.
4. Define $\mathcal{T}_{\text{sk}}[\tilde{t}_{v_b, j}, \mu_j] := \mathbf{x}_j$.

According to the $\text{MLWE}_{q, \ell, m, \eta}$ assumption, $\tilde{\mathbf{A}} \cdot \mathbf{x}_j$ is indistinguishable from uniform. Furthermore, $\mathbf{b}_j = (\mathbf{t} - \tilde{\mathbf{A}} \cdot \mathbf{x}_j)$ should be indistinguishable from uniform.

As a result, it holds

$$|\Pr[\mathbf{G}_2] - \Pr[\mathbf{G}_1]| \leq (Q_h - 1) \cdot \text{Adv}_{\text{MLWE}_{q, \ell, m, \eta}}(\mathcal{B}_2).$$

G₃ This experiment is identical to **G₂**, except that the signing oracles Osign_1 and Osign_2 are answered through using the Sim procedure in Table 4, which means the signing queries are answered without using the original secret key s^* . Particularly, after receiving the signature query with respect to (L, μ) with $L = \{\text{pk}_1, \dots, \text{pk}_n\}$, the simulator conducts as follows:

1. Compute $\tilde{t}_{v_b} \leftarrow \text{KeyAgg}(L)$.
2. Use the related value $\mathcal{T}_{\text{sk}}[\tilde{t}_{v_b}, \mu] = \mathbf{x}$ to generate $\mathbf{w}$ and Sig, through running the Sim procedure in Table 4.

As a result, it holds

- Statistical Distance:

$$|\Pr[\mathbf{G}_3] - \Pr[\mathbf{G}_2]| \leq Q_s \cdot \left(2^{-\Omega(N)} + \frac{\epsilon}{M}\right) + \frac{Q_h(Q_h - 1)}{|C|} + \left(\frac{2}{q^{N/2}}\right)^m,$$

- Renyi Divergence:

$$\Pr[\mathbf{G}_2] \leq \Pr[\mathbf{G}_3] \cdot \left(1 + \frac{\epsilon}{M-1}\right)^{2Q_s} + 2^{-\Omega(N)} + \frac{Q_h(Q_h - 1)}{|C|} + \left(\frac{2}{q^{N/2}}\right)^m,$$

where $\frac{Q_h(Q_h - 1)}{|C|} + \left(\frac{2}{q^{N/2}}\right)^m$ is to deal with the event that $\mathcal{A}$ ever queried two different public-key list $L \neq L^*$ satisfying $\text{KeyAgg}(L) = \text{KeyAgg}(L^*)$, as analysis in [8].

G₄ This experiment is identical to **G₃**, except that the public key $\text{pk}^* := \mathbf{t}^*$ is chosen uniformly at random, i.e., $\mathbf{t}^* \xleftarrow{\$} R_q^m$. According to the $\text{MLWE}_{q, \ell, m, \eta}$ assumption, it holds

$$|\Pr[\mathbf{G}_4] - \Pr[\mathbf{G}_3]| \leq \text{Adv}_{\text{MLWE}_{q, \ell, m, \eta}}(\mathcal{B}_2),$$

Next, we need to analyze the upper bound of $\Pr[\mathbf{G}_4]$. For this, we consider the aggregation special soundness of the underlying identification scheme. Particularly, given $(\mathbf{A}, \mathbf{t}_1^*, \mathbf{b}^*, \mathbf{B}) \xleftarrow{\$} R_q^{m \times \ell} \times R_q^m \times R_q^m \times R_q^{m \times \ell}$, an efficient adversary $\mathcal{A}_{\text{spesound}}$ chooses arbitrary public key $\mathbf{t}_i^* \in R_q^m$ for $2 \leq i \leq n$, which will derive the aggregated public key $\tilde{\mathbf{t}}_{v_b}^* \leftarrow \text{KeyAgg}(\mathbf{t}_1^*, \{\mathbf{t}_j^*\}_{2 \leq j \leq n})$. Then $\mathcal{A}_{\text{spesound}}$ outputs a tuple $(\tilde{\mathbf{w}}_{v_w}, c, c', \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{h}}, \tilde{\mathbf{z}}', \tilde{\mathbf{r}}', \tilde{\mathbf{h}}')$, such that $(c, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{h}})$ and $(c', \tilde{\mathbf{z}}', \tilde{\mathbf{r}}', \tilde{\mathbf{h}}')$ are two different but valid transcripts with respect to the same commitment $\tilde{\mathbf{w}}_{v_w}$.

Below, we prove that the following probability is negligible

$$\Pr \left[\begin{array}{l} (c \neq c') \wedge (\|\tilde{\mathbf{z}}, \mathbf{h} \cdot 2^{v_w}\| \leq B) \wedge (\|\tilde{\mathbf{r}}\| \leq B'_n) \wedge (\|\tilde{\mathbf{r}}'\| \leq B''_n) \\ \wedge (\|\tilde{\mathbf{z}}', \mathbf{h}' \cdot 2^{v_w}\| \leq B) \wedge (\|\tilde{\mathbf{r}}'\| \leq B'_n) \wedge (\|\tilde{\mathbf{r}}'\| \leq B''_n) \\ \wedge (\tilde{\mathbf{w}}_{v_w} = \lfloor \mathbf{A}\tilde{\mathbf{z}} - c \cdot 2^{v_b} \cdot \tilde{\mathbf{t}}_{v_b}^* + \tilde{\mathbf{r}} \cdot \mathbf{b}^* + \mathbf{B} \cdot \tilde{\mathbf{r}} \rfloor_{v_w} + \mathbf{h}) \\ \wedge (\tilde{\mathbf{w}}_{v_w} = \lfloor \mathbf{A}\tilde{\mathbf{z}}' - c' \cdot 2^{v_b} \cdot \tilde{\mathbf{t}}_{v_b}^* + \tilde{\mathbf{r}}' \cdot \mathbf{b}^* + \mathbf{B} \cdot \tilde{\mathbf{r}}' \rfloor_{v_w} + \mathbf{h}') \end{array} \right] \quad (9)$$

For the verification equations in (9), according to (5) in Lemma 3.1, we have $2^{v_w} \tilde{\mathbf{w}}_{v_w} = \tilde{\mathbf{w}} - \delta_w \pmod{q}$, where $\|\delta_w\|_\infty \leq 2^{v_w} - 1$. Thus, we obtain the exact equation over R_q as follows:

$$\begin{aligned} 2^{v_w} \tilde{\mathbf{w}}_{v_w} &= \mathbf{A}\tilde{\mathbf{z}} - c \cdot 2^{v_b} \tilde{\mathbf{t}}_{v_b}^* + \tilde{\mathbf{r}} \mathbf{b}^* + \mathbf{B}\tilde{\mathbf{r}} + 2^{v_w} \mathbf{h} - \delta_w \pmod{q} \\ &= \mathbf{A}\tilde{\mathbf{z}}' - c' \cdot 2^{v_b} \tilde{\mathbf{t}}_{v_b}^* + \tilde{\mathbf{r}}' \mathbf{b}^* + \mathbf{B}\tilde{\mathbf{r}}' + 2^{v_w} \mathbf{h}' - \delta_w \pmod{q}. \end{aligned} \quad (10)$$

Moreover, according to $2^{v_b} \tilde{\mathbf{t}}_{v_b}^* = \tilde{\mathbf{t}}^* - \delta_t \pmod{q}$, where $\|\delta_t\|_\infty \leq 2^{v_b} - 1$ and $\tilde{\mathbf{t}}^* = \sum_{i=1}^n a_i^* \mathbf{t}_i^*$ with $a_i^* = \text{H}_{\text{agg}}(\{\mathbf{t}_i\}_{i \in [n]}, \mathbf{t}_i) \in C$, we get the following equations from (10)

$$\begin{aligned} 2^{v_w} \tilde{\mathbf{w}}_{v_w} &= \mathbf{A}\tilde{\mathbf{z}} - c \cdot \tilde{\mathbf{t}}^* + c \cdot \delta_t + \tilde{\mathbf{r}} \mathbf{b}^* + \mathbf{B}\tilde{\mathbf{r}} + 2^{v_w} \mathbf{h} - \delta_w \pmod{q} \\ &= \mathbf{A}\tilde{\mathbf{z}}' - c' \cdot \tilde{\mathbf{t}}^* + c' \cdot \delta_t + \tilde{\mathbf{r}}' \mathbf{b}^* + \mathbf{B}\tilde{\mathbf{r}}' + 2^{v_w} \mathbf{h}' - \delta_w \pmod{q}. \end{aligned} \quad (11)$$

Let Δ denote the difference between the two transcripts (e.g., $\Delta\tilde{\mathbf{z}} = \tilde{\mathbf{z}} - \tilde{\mathbf{z}}'$, $\Delta c = c - c'$, $\Delta\mathbf{h} = \mathbf{h} - \mathbf{h}'$). Subtracting the corresponding Eq. (11) of the two valid transcripts yields:

$$\mathbf{A}\Delta\tilde{\mathbf{z}} + \Delta\tilde{\mathbf{r}} \cdot \mathbf{b}^* + \mathbf{B}\Delta\tilde{\mathbf{r}} + \Delta c \cdot \delta_t + 2^{v_w} \Delta\mathbf{h} = \Delta c \cdot \tilde{\mathbf{t}}^* \pmod{q}. \quad (12)$$

Notice that $\tilde{\mathbf{t}}^* = n^* a_1^* \mathbf{t}_1^* + \sum_{\mathbf{t}_i \in L^* \setminus \{\mathbf{t}_1^*\}} a_i^* \mathbf{t}_i^*$, where n^* denotes the number of times $\mathbf{t}_1^*$ occurs in L^* . Then, (12) implies:

$$\begin{aligned} [\mathbf{A} \mid \mathbf{b}^* \mid \mathbf{B}] \cdot \begin{bmatrix} \Delta\tilde{\mathbf{z}} \\ \Delta\tilde{\mathbf{r}} \\ \Delta\tilde{\mathbf{r}} \end{bmatrix} + \Delta c \cdot \delta_t + 2^{v_w} \Delta\mathbf{h} \\ = \Delta c \cdot \left(n^* \cdot a_1^* \cdot \mathbf{t}_1^* + \sum_{\mathbf{t}_i \in L^* \setminus \{\mathbf{t}_1^*\}} a_i^* \cdot \mathbf{t}_i^* \right) \pmod{q}. \end{aligned} \quad (13)$$

Then, we apply the rewinding technique and treat $a_1^* \leftarrow \text{H}_{\text{agg}}(L^*, \mathbf{t}_1^*)$ as the crucial query. Particularly, we consider the forking algorithm $\mathcal{B}$ that runs $\mathcal{A}_{\text{spesound}}$ twice. During $\mathcal{B}$'s second run of $\mathcal{A}_{\text{spesound}}$, $\mathcal{B}$ assigns a different value $a_1^{*'} as the output of$

$H_{\text{agg}}(L^*, \mathbf{t}_1^*)$, which consequently alters $\tilde{\mathbf{t}}^*$ and its rounding error δ'_t . According to the forking lemma, $\mathcal{A}_{\text{spesound}}$ will output a second set of collisions. Thus, from the second run, we obtain a similar relation:

$$\begin{aligned} [\mathbf{A} \mid \mathbf{b}^* \mid \mathbf{B}] \cdot \begin{bmatrix} \Delta \tilde{\mathbf{z}}' \\ \Delta \tilde{\mathbf{r}}' \\ \Delta \tilde{\mathbf{r}}' \end{bmatrix} + \Delta c' \delta'_t + 2^{\nu_w} \Delta \mathbf{h}' \\ = \Delta c' \cdot \left(n^* \cdot a_1^* \cdot \mathbf{t}_1^* + \sum_{\mathbf{t}_i \in L^* \setminus \{\mathbf{t}_1^*\}} a_i^* \cdot \mathbf{t}_i^* \right) \pmod{q}. \end{aligned} \quad (14)$$

To eliminate the terms unrelated to $\mathbf{t}_1^*$, we multiply (13) by $\Delta c'$, (14) by Δc , and subtract the latter from the former. This yields a clean MSIS instance:

$$[\mathbf{A} \mid \mathbf{b}^* \mid \mathbf{B} \mid \mathbf{t}_1^* \mid \mathbf{I}] \cdot \mathbf{v}_{\text{sol}} = \mathbf{0} \pmod{q}, \quad (15)$$

where the solution vector $\mathbf{v}_{\text{sol}} \in R_q^{2\ell+2}$ is constructed as:

$$\mathbf{v}_{\text{sol}} = \begin{bmatrix} \Delta c' \Delta \tilde{\mathbf{z}} - \Delta c \Delta \tilde{\mathbf{z}}' \\ \Delta c' \Delta \tilde{\mathbf{r}} - \Delta c \Delta \tilde{\mathbf{r}}' \\ \Delta c' \Delta \tilde{\mathbf{r}} - \Delta c \Delta \tilde{\mathbf{r}}' \\ -\Delta c \cdot \Delta c' \cdot n^* \cdot (a_1^* - a_1'^*) \\ \Delta c' \cdot 2^{\nu_w} \Delta \mathbf{h}' - \Delta c \cdot 2^{\nu_w} \Delta \mathbf{h}' + \Delta c' \cdot \Delta c \cdot \delta'_t - \Delta c \cdot \Delta c' \cdot \delta'_t \end{bmatrix}.$$

It remains to rigorously bound the norm of $\mathbf{v}_{\text{sol}}$. Recall that the challenges $c, c', a_i^* \in C$ have ℓ_1 -norm at most κ , implying $\|\Delta c\|_1 \leq 2\kappa$. From the verification constraints, we have $\|(\tilde{\mathbf{z}}, \mathbf{h} \cdot 2^{\nu_w})\| \leq B$, $\|\tilde{\mathbf{r}}\| \leq B'_n$, and $\|\tilde{\mathbf{r}}'\| \leq B''_n$. The rounding errors inherently satisfy $\|\delta_t\|_\infty \leq 2^{\nu_b} - 1$. We analyze each component:

- Since $\Delta c \neq 0, \Delta c' \neq 0, a_1^* \neq a_1'^*, \mathbf{v}_{\text{sol}}$ is a non-zero vector.
- Since $\|(\tilde{\mathbf{z}}, \mathbf{h} \cdot 2^{\nu_w})\| \leq B$, we have $\|(\Delta \tilde{\mathbf{z}}, 2^{\nu_w} \Delta \mathbf{h}')\| \leq 2B$. Furthermore, we have $\|(\Delta c' \Delta \tilde{\mathbf{z}}, \Delta c' \cdot 2^{\nu_w} \Delta \mathbf{h}')\| \leq 4\kappa \cdot B$, which implies $\|(\Delta c' \Delta \tilde{\mathbf{z}} - \Delta c \Delta \tilde{\mathbf{z}}', \Delta c' \cdot 2^{\nu_w} \Delta \mathbf{h}' - \Delta c \cdot 2^{\nu_w} \Delta \mathbf{h}')\| \leq 8\kappa \cdot B$.
- Similarly, the second and third blocks are bounded by $8\kappa B'_n$ and $8\kappa B''_n$, respectively.
- The fourth component is bounded by $n^* (2\sqrt{\kappa})^3 = 8n^* \kappa^{3/2}$.
- The remain part $\|\Delta c' \cdot \Delta c \cdot \delta'_t - \Delta c \cdot \Delta c' \cdot \delta'_t\|$ is upper bounded by $8\kappa(2^{\nu_b} - 1) \cdot \sqrt{N \cdot m}$.

Consequently, the extracted vector $\mathbf{v}_{\text{sol}}$ is non-zero and satisfies $\|\mathbf{v}_{\text{sol}}\| \leq \gamma$, where

$$\gamma := 8\kappa \sqrt{B^2 + B_n'^2 + B_n''^2 + n^{*2} \kappa + 2^{2\nu_b} \cdot N \cdot m}. \quad (16)$$

Since γ is polynomially bounded, the algorithm $\mathcal{B}_1$ successfully extracts a valid solution to the $\text{MSIS}_{q,m,2\ell+m+2,\gamma}$ problem. Thus, the probability that the Forking algorithm succeeds is $\Pr[\mathcal{B}_1 \text{ wins}] \leq \text{Adv}_{\text{MSIS}_{q,m,2\ell+m+2,\gamma}}$. Finally, applying the General Forking Lemma and the standard relation between soundness and special soundness, we tightly bound the advantage of the original adversary against $\mathbf{G}_4$:

$$\begin{aligned} \Pr[\mathbf{G}_4] &\leq Q_h \cdot \left(\text{Adv}^{\text{sound}}(\widehat{\mathcal{B}}) \right) \leq Q_h \cdot \left(\sqrt{\text{Adv}^{\text{special-sound}}(\mathcal{A}_{\text{spesound}})} \right. \\ &\quad \left. + \frac{1}{|C|} \right) \leq Q_h \cdot \left(\sqrt{\frac{Q_h}{|C|}} + \sqrt{Q_h \cdot \text{Adv}_{\text{MSIS}_{q,m,2\ell+m+2,\gamma}}(\mathcal{B}_1)} + \frac{1}{|C|} \right), \end{aligned} \quad (17)$$

which concludes the proof of the theorem. Here, we use $\widehat{\mathcal{B}}$ (or $\mathcal{A}_{\text{spesound}}$) to denote the adversaries against the aggregation (special) soundness of the underlying identification scheme.

4 Parameter Selection, Comparison, and Implementation

In this section, we provide concrete parameter settings for our scheme, considering 128 bits classical security and the number of signers to be $n = 32$ and $n = 1024$. Then, we compare these parameters with those of DualMS. Finally, we implement our scheme and analyze its performance on computation and communication.

4.1 Parameters selection

For our DualMS 2.0, we fix the polynomial ring degree $N = 256$ and the maximum ℓ_1 norm of the challenge C as $\kappa = 60$. The matrix $\mathbf{A} \in R_q^{m \times \ell}$ has height m and width ℓ , with $m = 7, \ell = 6$ for $n = 32$ signers and $m = 10, \ell = 6$ for $n = 1024$ signers. The secret key bound is $\eta = 1$ in all cases.

For the rejection sampling algorithm in our DualMS 2.0 scheme, we set the standard deviation of $\mathbf{y}$ to be $\sigma = \alpha \cdot T$. Here $T = \kappa^2 \eta \sqrt{N(\ell + m + 1)}$ denotes the upper bound on the norm of the secret vector, and $\alpha = 4.917n$ is to make the expected repetition times $M \approx 5$, according to the Rényi divergence analysis of the Lemma C.2 in [12].

Furthermore, we use the modulus rounding technique, to optimize the sizes of aggregated public-key and aggregated signature. Particularly, the rounding-off parameters ν_b and ν_w are selected following the approach in Table 3 of [27].

We evaluate the security of MLWE with the lwe-estimator tool¹⁰, and MSIS with Dilithium security estimator¹¹, we set $\gamma \approx \sqrt{B^2 + B_n'^2 + B_n''^2 + n^{*2}\kappa + 2^{2\nu_b} \cdot N \cdot m}$, similar to the setting in [8]. The challenge size in the signature is set to be 32 bytes, following [8]. The parameters are selected to minimize the total size of public key and signature while achieving the target security level. Concretely, all parameter settings are listed in Table 6.

¹⁰ <https://github.com/malb/lattice-estimator>

¹¹ <https://github.com/pq-crystals/security-estimates>

Table 6. Concrete parameter settings and comparison between DualMS, DualMS2.0, and rounded DualMS2.0 for $n = 32$ and $n = 1024$ signers. Here we list $|\text{pk}|$ and $|\text{sig}|$ as the sizes (KB) of the individual public-key and individual signature, respectively, while $|\text{aggpk}|$ and $|\text{AggSig}|$ denote the sizes (KB) of the aggregated public-key and aggregated signature, respectively. “–” means not applicable. MSIS and MLWE denote the bit-security levels of the underlying hard problems.

Parameter	DualMS		DualMS2.0 (without rounding)		DualMS2.0 (with rounding)	
n	32	1024	32	1024	32	1024
N	256	256	256	256	256	256
$\lceil \log(q) \rceil$	37	48	31	32	31	32
m	6	7	7	10	7	10
ℓ	7	9	6	6	6	6
η	1	1	1	1	1	1
ν_b	–	–	–	–	11	11
ν_w	–	–	–	–	26	28
$ \text{pk} $	6.94	10.50	6.78	10.00	6.78	10.00
$ \text{aggpk} $	6.94	10.50	6.78	10.00	4.38	6.56
$ \text{sig} $	22.81	34.06	12.40	17.79	12.59	18.02
$ \text{AggSig} $	19.72	30.91	13.62	20.90	8.86	11.62
$ \text{aggpk} + \text{AggSig} $	26.66	41.41	20.41	30.90	13.23	18.18
$\log(\sigma)$	27.13	32.27	25.02	30.16	25.02	30.16
MSIS	130	132	129.7	126.2	128.0	126.1
MLWE	130	131	132.6	127.3	132.6	127.3

4.2 Comparison

Table 6 also presents the comparison between our scheme and DualMS. For the DualMS scheme, we adopt the parameter settings from the Table 2 of [8]. It shows that, with the same target security level about 128 bits, our rounded DualMS2.0 achieves roughly 55.1% and 62.4% reduction in aggregated signatures for $n = 32$ and $n = 1024$, respectively; and roughly 36.9% and 37.5% reduction in aggregated public keys for $n = 32$ and $n = 1024$, respectively, compared with DualMS.

Notice that, in order to ensure the fairness for the above comparison, we also present parameters for the non-optimization version of our DualMS 2.0, since all the previous schemes DualMS, MulSig-L does not apply modulus rounding to optimize size. Even in this setting, our non-optimization version still reduces the aggregated signature size by roughly 32% compared with the original DualMS. For completeness, we present such non-optimization version in Section A.

4.3 Implementation

In order to evaluate the benchmark performance of DualMS and our rounded DualMS2.0, we implement both schemes in Go using the open-source library Lattigo¹² [24], and use the codebase of Ringtail¹³ [6] as a reference.

In particular, we instantiate our schemes over power-of-two cyclotomic rings and replace the paper modulus q with an NTT-friendly modulus $q \equiv 1 \pmod{2N}$ of the same bit-length in the implementation.

¹² <https://github.com/tuneinsight/lattigo>

¹³ <https://github.com/daryakaviani/ringtail>

For the local benchmark experiments, we run both implementations on a laptop with an AMD Ryzen 6800H 8-core 4.3 GHz CPU and 32 GB RAM. All measurements use a single core and do not include any parallelism optimization. We instantiate the schemes with the parameter settings listed in Table 6. Table 7 shows the average performance over 50 independent benchmark runs.

Table 7. Local benchmark results (ms) for DualMS and our rounded DualMS2.0. **KeyGen** denotes the per-signer local runtime, and **KeyAgg** denotes the runtime of the key aggregation phase. **Sign1** and **Sign2** include the per-signer local runtime spent in aborted attempts together with the per-signer local runtime in the final successful attempt. **SAgg** and **Verify** denote the total runtime of the corresponding phase in the final successful attempt, and t denotes the mean number of aborts before outputting a valid aggregated signature.

n	Scheme	KeyGen	KeyAgg	Sign1	Sign2	SAgg	Verify	t
32	DualMS	0.36	2.27	$2.07t + 2.07$	$3.89t + 3.89$	1.30	3.27	3.4
	DualMS2.0 (with rounding)	0.40	3.20	$2.34t + 2.35$	$5.12t + 5.12$	3.73	4.54	4.1
1024	DualMS	0.49	74.57	$3.00t + 3.00$	$11.13t + 11.14$	46.18	4.09	4.4
	DualMS2.0 (with rounding)	0.59	100.42	$3.35t + 3.34$	$14.05t + 14.05$	24.51	5.19	4.3

Notice that in the **KeyAgg** phase, our scheme is slower than DualMS. This is because the public key t_i has a larger dimension than in DualMS (for example, $m = 10$ versus 7 when $n = 1024$), which means that our DualMS 2.0 needs to conduct many more ring multiplication operations. But we believe such additional time cost will not affect the overall practical performance, because it is still completed in about 0.1 second. Similarly, the **Sign1** and **Sign2** phases are also slightly slower due to the concrete parameter settings. We also note that, our **SAgg** algorithm performs much better than that of DualMS. This is because our algorithm only needs to sum $z_{i,1}$ and $r_{i,2,1}$, which are the first part of z_i and $r_{i,2}$, respectively.

To show that our implementation remains practical in a real network setting, we evaluate its WAN latency by deploying our rounded DualMS2.0 on servers across four locations: Los Angeles in the United States, Dubai in the United Arab Emirates, Amsterdam in the Netherlands, and Hong Kong, China. In this WAN experiment, we set $n = 8$ active signers. We use the Hong Kong server as the coordinator. It collects public keys, runs **KeyAgg**, executes its own local signing steps, collects the round-1 messages and partial signatures from the other servers, and finally runs **SAgg** and **Verify**.

We average the WAN benchmark results over 10 independent executions in Table 8.

Table 8. Global WAN latency (ms) for DualMS and rounded DualMS2.0 with $n = 8$. Local Compute and Network Latency denote the coordinator’s local computation and the network communication in the successful attempt, respectively. Combine denotes the final aggregation step, and End-to-End is the per-round sum of the above three procedures. t denotes the average abort times before outputting a valid aggregated signature, and Traffic denotes the coordinator’s mean total traffic per signing session. “-” denotes not applicable.

Scheme	Round	Local Compute	Network Latency	Combine	End-to-End	t	Traffic (MB)
DualMS	Sign1	3.790	332.061	-	335.851	5.1	6.250
	Sign2	16.200	495.428	1.209	512.836		
DualMS2.0 (rounding)	Sign1	5.539	278.665	-	284.204	4.6	5.839
	Sign2	22.922	436.705	6.670	466.297		

Because the servers are located far apart, the network latency dominates the end-to-end performance. For rounded DualMS2.0 the network latency accounts for about 97.8% of the round-1 Sign1 latency and 93.7% of the round-2 Sign2 latency, respectively. We also note that our servers have limited computing resources, with only a single-core CPU and 0.5 GB of RAM. Therefore, their performance is lower than the local benchmark results.

Since DualMS does not provide a WAN benchmark, we also implement it under the same experimental setting for comparison. We note that our DualMS2.0 achieves lower WAN latency than DualMS. This is mainly because DualMS2.0 sends ring elements with a smaller modulus q in both rounds.

5 Open Science

The scripts and the proof-of-concept implementation used for the benchmark in this paper are available at <https://github.com/barlogs/DualMS2.0-CCS-sub>. This implementation and benchmark should not be used in real-world applications.

Acknowledgments. We would like to thank the anonymous reviewers of CCS 2026 for their insightful advices. This work was supported by National Cryptologic Science Fund of China (No. 2025NCSF02036), National Natural Science Foundation of China (62172266), NSF Career Award CNS-2402031, the Key Research and Development Program of Shaanxi under Grant 2024GXZDCYL-01-15, State Key Laboratory of Cryptography and Digital Economy Security, Shandong University (No. KFYB2507).

References

1. Alper, H.K., Burdges, J.: Two-round trip Schnorr multi-signatures via delinearized witnesses. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part I. LNCS, vol. 12825, pp. 157–188. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84242-0_7
2. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 595–618. Springer, Berlin, Heidelberg (Aug 2009). https://doi.org/10.1007/978-3-642-03356-8_35

3. Bagherzandi, A., Cheon, J.H., Jarecki, S.: Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM CCS 2008. pp. 449–458. ACM Press (Oct 2008). <https://doi.org/10.1145/1455770.1455827>
4. Bai, S., Lepoint, T., Roux-Langlois, A., Sakzad, A., Stehlé, D., Steinfeld, R.: Improved security proofs in lattice-based cryptography: Using the Rényi divergence rather than the statistical distance. *Journal of Cryptology* **31**(2), 610–640 (Apr 2018). <https://doi.org/10.1007/s00145-017-9265-9>
5. Boneh, D., Kim, S.: One-time and interactive aggregate signatures from lattices. *Proceedings of the 6th on ASIA Public-Key Cryptography Workshop (2020)*, https://crypto.stanford.edu/~skim13/agg_ots.pdf
6. Boschini, C., Kaviani, D., Lai, R.W.F., Malavolta, G., Takahashi, A., Tibouchi, M.: Ringtail: Practical two-round threshold signatures from learning with errors. In: Blanton, M., Enck, W., Nita-Rotaru, C. (eds.) 2025 IEEE Symposium on Security and Privacy. pp. 149–164. IEEE Computer Society Press (May 2025). <https://doi.org/10.1109/SP61157.2025.00070>
7. Boschini, C., Takahashi, A., Tibouchi, M.: MuSig-L: Lattice-based multi-signature with single-round online phase. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 276–305. Springer, Cham (Aug 2022). https://doi.org/10.1007/978-3-031-15979-4_10
8. Chen, Y.: DualMS: Efficient lattice-based two-round multi-signature with trapdoor-free simulation. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part V. LNCS, vol. 14085, pp. 716–747. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38554-4_23
9. Damgård, I., Orlandi, C., Takahashi, A., Tibouchi, M.: Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. *Journal of Cryptology* **35**(2), 14 (Apr 2022). <https://doi.org/10.1007/s00145-022-09425-3>
10. Del Pino, R., Katsumata, S., Maller, M., Mouhartem, F., Prest, T., Saarinen, M.J.O.: Threshold raccoon: Practical threshold signatures from standard lattice assumptions. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part II. LNCS, vol. 14652, pp. 219–248. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58723-8_8
11. Devevey, J., Fawzi, O., Passelègue, A., Stehlé, D.: On rejection sampling in Lyubashevsky’s signature scheme. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part IV. LNCS, vol. 13794, pp. 34–64. Springer, Cham (Dec 2022). https://doi.org/10.1007/978-3-031-22972-5_2
12. Devevey, J., Fawzi, O., Passelègue, A., Stehlé, D.: On rejection sampling in lyubashevsky’s signature scheme. *Cryptology ePrint Archive, Report 2022/1249* (2022), <https://eprint.iacr.org/2022/1249>
13. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR TCHES* **2018**(1), 238–268 (2018). <https://doi.org/10.13154/tches.v2018.i1.238-268>, <https://tches.iacr.org/index.php/TCHES/article/view/839>
14. El Bansarkhani, R., Sturm, J.: An efficient lattice-based multisignature scheme with applications to Bitcoins. In: Foresti, S., Persiano, G. (eds.) CANS 16. LNCS, vol. 10052, pp. 140–155. Springer, Cham (Nov 2016). https://doi.org/10.1007/978-3-319-48965-0_9
15. Fukumitsu, M., Hasegawa, S.: A tightly-secure lattice-based multisignature. *Proceedings of the 6th on ASIA Public-Key Cryptography Workshop (2019)*, <https://api.semanticscholar.org/CorpusID:195848927>
16. Fukumitsu, M., Hasegawa, S.: A lattice-based provably secure multisignature scheme in quantum random oracle model. In: Nguyen, K., Wu, W., Lam, K.Y., Wang, H. (eds.) ProvSec 2020. LNCS, vol. 12505, pp. 45–64. Springer, Cham (Nov / Dec 2020). https://doi.org/10.1007/978-3-030-62576-4_3

17. Lai, Q., Liu, F.H., Lu, Y., Xue, H., Yu, Y.: Scalable two-round n -out-of- n and multi-signatures from lattices in the quantum random oracle model. Cryptology ePrint Archive, Report 2024/1574 (2024), <https://eprint.iacr.org/2024/1574>
18. Langlois, A., Stehle, D.: Worst-case to average-case reductions for module lattices. Designs, Codes and Cryptography (2015)
19. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Berlin, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_43
20. Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-LWE cryptography. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 35–54. Springer, Berlin, Heidelberg (May 2013). https://doi.org/10.1007/978-3-642-38348-9_3
21. Ma, C., Jiang, M.: Practical lattice-based multisignature schemes for blockchains. IEEE Access **7**, 179765–179778 (2019). <https://doi.org/10.1109/ACCESS.2019.2958816>
22. Maxwell, G., Poelstra, A., Seurin, Y., Wuille, P.: Simple Schnorr multi-signatures with applications to Bitcoin. Cryptology ePrint Archive, Report 2018/068 (2018), <https://eprint.iacr.org/2018/068>
23. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 21–39. Springer, Berlin, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_2
24. Mouchet, C.V., Bossuat, J.P., Troncoso-Pastoriza, J.R., Hubaux, J.P.: Lattigo: A multiparty homomorphic encryption library in go. In: Proceedings of the 8th Workshop on Encrypted Computing and Applied Homomorphic Cryptography. pp. 64–70 (2020)
25. Nick, J., Ruffing, T., Seurin, Y., Wuille, P.: MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 1717–1731. ACM Press (Nov 2020). <https://doi.org/10.1145/3372297.3417236>
26. Pan, J., Wagner, B.: Chopsticks: Fork-free two-round multi-signatures from non-interactive assumptions. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 597–627. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30589-4_21
27. Pino, R.D., Niot, G.: Finally! A compact lattice-based threshold signature. In: Jager, T., Pan, J. (eds.) PKC 2025, Part III. LNCS, vol. 15676, pp. 169–199. Springer, Cham (May 2025). https://doi.org/10.1007/978-3-031-91826-1_6

For completeness, we present the underlying DualMS 2.0 without modulus rounding in this section.

A Underlying DualMS 2.0 without Modulus Rounding

For completeness, we present the underlying DualMS 2.0 without modulus rounding in this section.

Parameter	Description
n	Number of parties
N	Power of two defining the degree of $\Phi_{2N}(X)$
$\Phi_{2N}(X) = X^N + 1$	The $2N$ -th cyclotomic polynomial
k	Power of two with $k N$
q	Prime modulus with the form of $q = 2k + 1 \pmod{4k}$
$R = \mathbb{Z}[X]/(\Phi_{2N}(X)), R_q = \mathbb{Z}_q[X]/(\Phi_{2N}(X))$	Cyclotomic ring
$S_\eta = \{x \in R : \ x\ _\infty \leq \eta\}$	Set of small secrets
m	The height of random matrices $\mathbf{A}$
ℓ	The width of random matrices $\mathbf{A}$
$\sigma \geq \max(\alpha \cdot \kappa^2 \eta \sqrt{N(\ell + m + 1)}, \frac{2N \cdot q^{m/(m+\ell)+2/(N(m+\ell))}}{\sqrt{2\pi}})$	The standard deviation of $\mathbf{y}$ and $\mathbf{r}$
$B = \sigma \sqrt{2(\ell + m)N}$	The upper bound of $\ z_i\ $
$B_n = \sqrt{n}B$	The upper bound of $\ \tilde{z}\ $, with $\tilde{z} = \sum_{i=1}^n z_i$
$B' = \sigma \sqrt{2N}$	The upper bound of $\ r_i\ $
$B'_n = \sqrt{n}B'$	The upper bound of $\ \tilde{r}\ $, with $\tilde{r} = \sum_{i=1}^n r_i$
$B'' = \eta \sqrt{(\ell + m)N}$	The upper bound of $\ r_i\ $
$B''_n = \sqrt{n}B''$	The upper bound of $\ \tilde{r}\ $, with $\tilde{r} = \sum_{i=1}^n r_i$
$C = \{c \in R : \ c\ _\infty = 1 \wedge \ c\ _1 = \kappa\}$	Challenge space where $ C = \binom{N}{\kappa} 2^\kappa$
κ	The ℓ_1 -norm of challenge $c \in C$
$\mathcal{M}$	Message space
α	Parameter defining σ and M based on Lemma 2.8
$M := \exp\left(\frac{1}{2\alpha^2} + \frac{t}{\alpha}\right)$	The expected number of restarts of a single part

Table 9. Parameters of Our Two Round FSWA Multi-Signature

- **Setup**(1^λ): Given the security parameter λ as input, the trusted algorithm conducts the following steps:
 1. Set the parameters just as in Table 9.
 2. Samples a random matrix $\mathbf{A} \xleftarrow{\$} R_q^{m \times \ell}$ and set $\bar{\mathbf{A}} = [\mathbf{A} \mid \mathbf{I}] \in R_q^{m \times (\ell+m)}$.
 3. Let $H_{\text{agg}} : \{0, 1\}^* \rightarrow C$, $H_{\text{com}} : \{0, 1\}^* \rightarrow R_q^m$, $H_{\text{sig}} : \{0, 1\}^* \rightarrow C$ be random oracles.
 4. Finally, the algorithm outputs $\text{pp} := \bar{\mathbf{A}}$.
- **KeyGen**(pp): Given pp as input, the party P_1 conducts the following steps:
 1. Sample vector $s_1 \xleftarrow{\$} S_\eta^{\ell+m}$ and compute $t_1 = \bar{\mathbf{A}} \cdot s_1 \in R_q^m$;
 2. Output $\text{pk}_1 := t_1$, $\text{sk}_1 := s_1$.
- **KeyAgg**(L): Given the public-key list $L := \{t_1, \dots, t_n\}$ as input, the algorithm conducts the following steps:
 1. For $i \in [n]$, compute $a_i = H_{\text{agg}}(L, t_i) \in C$;
 2. Compute and return the aggregated public-key $\text{aggpk} := \tilde{t} = \sum_{i=1}^n a_i \cdot t_i \in R_q^m$.
- **Sign**₁($\text{pp}, \text{sk}_1, L, \mu$): Given $\text{pp}, \text{sk}_1 := s_1, L := \{t_1, \dots, t_n\}$, and $\mu \in \mathcal{M}$, the party P_1 conducts in the following ways:
 1. $[\mathbf{b}, \mathbf{B}] = H_{\text{com}}(\tilde{t}, \mu) \in R_q^{m \times (\ell+1)}$ and set $\bar{\mathbf{B}} = [\mathbf{B} \mid \mathbf{I}] \in R_q^{m \times (\ell+m)}$, where $\tilde{t} = \sum_{i=1}^n a_i \cdot t_i$ with $a_i = H_{\text{agg}}(L, t_i) \in C$.
 2. Sample $y_1 \xleftarrow{\$} \mathcal{D}_\sigma^{\ell+m}$, $r_1 \xleftarrow{\$} \mathcal{D}_\sigma$, $r_1 \xleftarrow{\$} S_\eta^{\ell+m}$, and compute $w_1 = \bar{\mathbf{A}} \cdot y_1 + r_1 \cdot \mathbf{b} + \bar{\mathbf{B}} \cdot r_1 \in R_q^m$.
 3. Return $\text{msg}_1 := w_1$, and $\text{st}_1 := (s_1, \tilde{t}, \mu, a_1, y_1, r_1, r_{1,2}, w_1)$.
- **Sign**₂($\text{pp}, \text{st}_1, \{\text{msg}_2, \dots, \text{msg}_n\}$): Given $\text{pp}, \text{st}_1 = (s_1, \tilde{t}, \mu, a_1, y_1, r_1, r_{1,2}, w_1)$, and $\{\text{msg}_2, \dots, \text{msg}_n\}$, the party P_1 can sign $\mu \in \mathcal{M}$ in the following ways:

1. Compute $\tilde{w} = \sum_{i=1}^n w_i$.
 2. Compute $c := H_{\text{sig}}(\tilde{w}, \mu, \tilde{t}) \in C$, and set $v_1 = ((a_1 \cdot c \cdot s_1)^\top, 0)^\top$.
 3. $(z_1^\top, r_1)^\top \leftarrow \text{RejSamp}(v_1, (y_1^\top, r_1)^\top, \mathcal{D}_{\sigma}^{\ell+m+1}, \mathcal{D}_{v_1, \sigma}^{\ell+m+1}, M)$.
 4. If $(z_1^\top, r_1)^\top \neq \perp$, return $\text{Sig}_2 := (c, z_1, r_1, \mathbf{r}_1)$; Otherwise, return $\text{Sig}_2 := \perp$.
- **SAgg**(pp, $\{\text{Sig}_i\}_{i \in [n]}$): Given pp, and $\sigma_i = (c_i, z_i, r_i, \mathbf{r}_i)$ from each party, the algorithm conducts the following steps:
1. Check if $\exists i \in [n], c_i \neq c_1$, then return $\perp$.
 2. Compute $\tilde{z} = \sum_{i=1}^n z_i$, $\tilde{r} = \sum_{i=1}^n r_i$, and $\tilde{\mathbf{r}} = \sum_{i=1}^n \mathbf{r}_i$.
 3. Output $\text{AggSig} := (c_1, \tilde{z}, \tilde{r}, \tilde{\mathbf{r}})$.¹⁴
- **Verify**(pp, μ , AggSig , aggpk): Given pp, μ , $\text{AggSig} := (c, \tilde{z}, \tilde{r}, \tilde{\mathbf{r}})$, and $\text{aggpk} := \tilde{t}$, the algorithm performs the following steps:
1. Compute $[\mathbf{b}, \mathbf{B}] = H_{\text{com}}(\tilde{t}, \mu) \in R_q^{m \times (\ell+1)}$.
 2. Compute $\tilde{w} := \tilde{\mathbf{A}} \cdot \tilde{z} - c \cdot \tilde{t} + \tilde{r} \cdot \mathbf{b} + \tilde{\mathbf{B}} \cdot \tilde{\mathbf{r}}$.
 3. If $H(\tilde{w}, \mu, \tilde{t}) = c \wedge \|\tilde{z}\| \leq B_n \wedge \|\tilde{r}\| \leq B'_n \wedge \|\tilde{\mathbf{r}}\| \leq B''_n$, return 1; Otherwise return 0.

A.1 Correctness

For correctness, we show that if each Sig_i is generated honestly, then all three conditions in Item 3 of the algorithm **Verify** will hold with overwhelming probability.

First, we analyze the algebraic structure of the aggregated signature. For each individual signature $\text{Sig}_i := (c, z_i, r_i, \mathbf{r}_i)$ generated by party P_i , we have:

$$z_i = y_i + a_i \cdot c \cdot s_i$$

The aggregated values are computed as:

$$\tilde{z} = \sum_{i=1}^n z_i, \quad \tilde{r} = \sum_{i=1}^n r_i, \quad \tilde{\mathbf{r}} = \sum_{i=1}^n \mathbf{r}_i$$

Substituting the aggregated public key $\tilde{t} = \sum_{i=1}^n a_i \cdot t_i = \tilde{\mathbf{A}} \cdot (\sum_{i=1}^n a_i \cdot s_i)$:

$$\begin{aligned}
\tilde{w} &= \tilde{\mathbf{A}} \cdot \left(\sum_{i=1}^n y_i + c \cdot \sum_{i=1}^n a_i \cdot s_i \right) - c \cdot \left(\tilde{\mathbf{A}} \cdot \sum_{i=1}^n a_i \cdot s_i \right) + \left(\sum_{i=1}^n r_i \right) \cdot \mathbf{b} + \tilde{\mathbf{B}} \cdot \left(\sum_{i=1}^n \mathbf{r}_i \right) \\
&= \sum_{i=1}^n \tilde{\mathbf{A}} \cdot y_i + c \cdot \tilde{\mathbf{A}} \cdot \sum_{i=1}^n a_i \cdot s_i - c \cdot \tilde{\mathbf{A}} \cdot \sum_{i=1}^n a_i \cdot s_i + \sum_{i=1}^n r_i \cdot \mathbf{b} + \sum_{i=1}^n \tilde{\mathbf{B}} \cdot \mathbf{r}_i \\
&= \sum_{i=1}^n (\tilde{\mathbf{A}} \cdot y_i + r_i \cdot \mathbf{b} + \tilde{\mathbf{B}} \cdot \mathbf{r}_i) \\
&= \sum_{i=1}^n w_i
\end{aligned}$$

Thus, $\tilde{w}$ recovered by the verifier matches the sum of individual commitments. Consequently, the hash check $H_{\text{sig}}(\tilde{w}, \mu, \tilde{t}) = c$ holds.

Regarding the norm bounds:

¹⁴ For better efficiency, **AggSig** can be further optimized using the compression technique from DualMS [8]. We defer the details to the parameter selection part in Section 4.1.

1. By Lemma 2.5, the sum of independent discrete Gaussian vectors $\tilde{\mathbf{z}} = \sum \mathbf{z}_i$ is statistically close to $\mathcal{D}_{\sqrt{n}\sigma}$. The same applies to $\tilde{\mathbf{r}}$ and $\tilde{\mathbf{r}}'$.
2. By the tail bounds of discrete Gaussian distributions (Lemma 2.4), $\|\tilde{\mathbf{z}}\| \leq B_n$, $\|\tilde{\mathbf{r}}'\| \leq B'_n$ and $\|\tilde{\mathbf{r}}\| \leq B''_n$ hold with probability $1 - 2^{-\Omega(N)}$.

Therefore, the correctness of our construction holds with overwhelming probability.

A.2 Security

Theorem A.1. *With the parameters specified in Table 9, our scheme is MS-UF-CMA-secure under $\text{MLWE}_{q,\ell,m,\eta}$ and $\text{MSIS}_{q,m,2\ell+m+2,\gamma}$ assumptions. Concretely, for any PPT adversary $\mathcal{A}$ against MS-UF-CMA-security that makes at most Q_h queries to each random oracle and at most Q_s queries to Osign_1 and Osign_2 , there exist PPT adversaries $\mathcal{B}_1, \mathcal{B}_2$ such that*

$$\begin{aligned} \text{Adv}_{\text{MS}_2}^{\text{UF-CMA}}(\mathcal{A}) &\leq Q_h \cdot \left[\left(Q_h \cdot \left(\sqrt{\frac{Q_h}{|C|}} + \sqrt{Q_h \cdot \text{Adv}_{\text{MSIS}_{q,m,2\ell+m+2,\gamma}}(\mathcal{B}_1)} + \frac{1}{|C|} \right) \right. \right. \\ &\quad \left. \left. + \text{Adv}_{\text{MLWE}_{q,\ell,m,\eta}}(\mathcal{B}_2) + \frac{Q_h(Q_h-1)}{|C|} + \left(\frac{2}{q^{N/2}}\right)^m \right) \cdot \left(1 + \frac{\epsilon}{M-1}\right)^{2Q_s} \right. \\ &\quad \left. + (Q_h-1) \cdot \text{Adv}_{\text{MLWE}_{q,\ell,m,\eta}}(\mathcal{B}_2) \right], \end{aligned}$$

where $\gamma \leq 8\kappa\sqrt{B_n^2 + B_n'^2 + B_n''^2 + n^{*2} \cdot \kappa^3}$, and n^* denotes the maximum number of duplicate key in a public key list.