DualMS: Efficient Lattice-Based Two-Round Multi-Signature with Trapdoor-Free Simulation*

Yanbo Chen[†]

June 8, 2023

Abstract

A multi-signature scheme allows multiple signers to jointly sign a common message. In recent years, two lattice-based two-round multi-signature schemes based on Dilithium-G were proposed: DOTT by Damgård, Orlandi, Takahashi, and Tibouchi (PKC'21) and MuSig-L by Boschini, Takahashi, and Tibouchi (CRYPTO'22).

In this work, we propose a lattice-based two-round multi-signature scheme called DualMS. Compared to DOTT, DualMS is likely to significantly reduce signature size, since it replaces an opening to a homomorphic trapdoor commitment with a Dilithium-G response in the signature. Compared to MuSig-L, concrete parameters show that DualMS has smaller public keys, signatures, and lower communication, while the first round cannot be preprocessed offline as in MuSig-L.

The main reason behind such improvements is a trapdoor-free "dual signing simulation" of our scheme. Signature simulation of DualMS is virtually identical the normal signing procedure and does not use lattice trapdoors like DOTT and MuSig-L.

Keywords. Multi-signature, Dilithium, Fiat-Shamir with aborts, Lattice, Post-quantum

1 Introduction

A multi-signature scheme [IN83] allows a group of signers, each with its own individual key pair, to run an interactive protocol to sign a common message. All signers authenticate the message together by producing one multi-signature, which should take much smaller space than a bunch of individual signatures. In recent years, multi-signatures have found some real-world applications in blockchain and crypto-currency.

Multi-signatures based on Schnorr signatures. An important line of research is multi-signatures based on Schnorr signatures. Bellare and Neven [BN06] made an early major step. They proposed a provably secure scheme in the *plain public-key model*, where each signer just publishes their public key in clear without any dedicated interactive key generation or proof of possession [RY07]. Their signing protocol has three rounds of interaction. Since then, a number of two-round schemes were proposed [BCJ08, MWLD10, STV+16, MPSW19]. Unfortunately, it was pointed out that these schemes are vulnerable to concurrent attacks [DEF+19, BLL+21]. After that, a number of provably secure two-round schemes against concurrent attacks were proposed [DEF+19, NRSW20, AB21, NRS21, BD21, TZ23, PW23]. Maxwell et al. [MPSW19] raised the idea of *key aggregation*. In a scheme that supports key aggregation, the public keys of signers

*An extended abstract of this paper appears in the proceedings of Crypto 2023. This is the full version.

[†]University of Ottawa. Email: ychen918@uottawa.ca. Part of this work done while at the Chinese University of Hong Kong. The author was supported by Hong Kong RGC GRF grant CUHK14207721.

can be non-interactively aggregated, and the verifier only needs the aggregated key in verification. While multi-signatures already save space for signatures, this property further reduces storage and communication for public keys. Most subsequent proposals support key aggregation.

Lattice-based multi-signatures. In recent years, some lattice-based multi-signature schemes were proposed. The earliest schemes [ES16, MJ19, FH19, FH20, BK20] are at least three-round. Moreover, the security proofs of [ES16, MJ19, FH19] are incomplete, observed by [DOTT22]. The only two lattice-based two-round proposals so far are DOTT given by Damgård, Orlandi, Takahashi, and Tibouchi [DOTT21, DOTT22] and MuSig-L recently by Boschini, Takahashi, and Tibouchi [BTT22a]. All schemes we mentioned above are based on the Fiat-Shamir with aborts (FSwA) paradigm [Lyu09, Lyu12], and they make use of many insights from Schnorr-based schemes. For example, [MJ19, BTT22b] use similar techniques to Schnorr-based schemes to support key aggregation in lattice setting. Recently, Fleischhacker et al. [FSZ22] proposed a non-interactive and concretely efficient scheme, but it only works in the synchronized model. In that setting, each signer can only produce one signature per time step, and only signatures produced in the same time step and on the same message can be aggregated.

Existing two-round lattice-based multi-signature schemes. In this work, we focus on lattice-based multi-signatures in the general setting (rather than restricted settings like the synchronized model). Existing two-round schemes, DOTT and MuSig-L, are both based on the non-optimized version of Dilithium-G [DLL+17], a FSwA signature scheme based on module SIS (MSIS) and LWE (MLWE). A Dilithium-G signature contains a relatively small challenge and a Gaussian distributed response that dominates the signature size.

DOTT is the first lattice-based two-round scheme. However, signature size of DOTT is relatively large. It takes homomorphic trapdoor commitment schemes as a building block, and its signature contains an opening to such a commitment in addition to a normal Dilithium-G signature. In the instantiations of such commitment schemes based on MSIS and MLWE [GVW15, LNTW19, DOTT22], the size of an opening is likely much larger than a Dilithium-G signature.

Compared to DOTT, signatures in MuSig-L are in the original form of Dilithium-G and do not contain extra openings. Moreover, its first round can be preprocessed offline before knowing the message to sign. However, the Gaussian width of the response in the signature is much larger, which somehow blows up the whole scheme and in particular, increases the public-key and signature size. MuSig-L also has much higher communication complexity than DOTT in typical parameter settings.

1.1 Our Contribution

In this work, we propose a lattice-based two-round multi-signature scheme, DualMS. Following DOTT and MuSig-L, DualMS is based on Dilithium-G [DLL+17]. A DualMS signature contains two responses instead of only one in Dilithium-G. Compared to DOTT signature, we replace the opening with a response and thus are likely to have much smaller signatures.

Compared to MuSig-L, DualMS has smaller public keys and signatures in our sample parameters. Aiming at about 128-bit security level, public-key size + signature size of DualMS and MuSig-L are approximately 27 kB vs. 124 kB with at most 32 signers and 41 kB vs. 139 kB with at most 1024 signers. Moreover, the communication of DualMS is smaller by an order of magnitude in such parameter settings.

Our scheme supports key aggregation using common techniques. We prove its security against concurrent attacks, in the plain public-key model and the random oracle model (ROM), based on MLWE and MSIS.

Underlying our result is a "dual signing simulation" technique that simulates multi-signatures in the security proof without trapdoors. In the rest of this section, we will review the trapdoor-based simulation techniques of DOTT and $\mathsf{MuSig\text{-}L}$, explain how they affect the performance of the schemes, and provide an overview of our scheme and simulation.

1.2 Simulation in Prior Works

Straight-line simulation. Let us consider multi-signatures based on Fiat-Shamir paradigm [FS87] or FSwA. To produce an individual signature in such a scheme, a signer first generates a random commitment, then hashes the commitment and the message to obtain a challenge, and finally gives a response to the challenge. A basic framework for multi-signature is as follows. The signers first take a round of interaction to exchange their individual commitments. They aggregate their commitments and hash the aggregated one to derive a common challenge or a bunch of per-signer challenges. After separately responding to the challenge(s), they exchange their responses in another round of interaction to finally compute an aggregated response. With fewer than two rounds of "exchange and aggregate", the size of multi-signature grows linearly with the number of signers.

However, this basic framework is not enough to construct a provably secure scheme. In the security proof, the reduction needs to simulate the signing procedure without the secret key. In the case of individual signatures, the reduction is allowed to generate the commitment, the challenge, and the response in any order. As long as it eventually outputs a valid signature, the order of simulation is hidden in a black box. Let us take Schnorr signature as an example. On input a challenge c, the reduction first samples a response z. Then it computes the commitment $R := g^z/X^c$, where g is the generator and X is the public key, and programs c into the random oracle. On the contrary, the order matters in the setting of multi-signatures. To play the part of an honest signer, the reduction has to give a commitment in the first round of interaction. At that time the challenge has not been determined yet, because it depends on those commitments given by other signers who are acted by the adversary. The reduction needs to output a correct response later when a challenge is decided. The standard simulation technique for Schnorr signatures does not work here, because the commitment R is decided after knowing c and z. This is an important observation: when we design a multi-signature scheme, we should intentionally enable such "simulation in order" or so-called *straight-line simulation*.

Trapdoor-based simulation techniques of existing schemes. We observe that DOTT and $\mathsf{MuSig\text{-}L}$ both rely on trapdoor sampling [GPV08, MP12] to enable straight-line simulation.

Let us first recall the underlying individual signature scheme, Dilithium-G. The scheme works over polynomial rings $R = \mathbb{Z}[X]/(f(X))$ and $R_q = \mathbb{Z}_q[X]/(f(X))$. In Dilithium-G, the secret key is a short vector $\mathbf{s} \in R^{l+k}$. The public key consists of a matrix $\mathbf{A} \in R_q^{k\times l}$ and a vector $\mathbf{t} := \bar{\mathbf{A}}\mathbf{s} \in R_q^k$ where $\bar{\mathbf{A}} := [\mathbf{A}|\mathbf{I}]$. To sign message μ, the signer first samples a masking vector $\mathbf{y} \in R^{l+k}$ from a discrete Gaussian distribution and computes a commitment $\mathbf{w} := \bar{\mathbf{A}}\mathbf{y} \in R_q^k$. It hashes $\mathbf{t}$, μ, and $\mathbf{w}$ to obtain a challenge $c := \mathsf{H}_{\mathrm{sig}}(\mathbf{t}, \mu, \mathbf{w})$ which is a small polynomial. It computes its response as $\mathbf{z} := \mathbf{y} + c\mathbf{s}$. Then it performs a rejection sampling: it aborts and restarts with some probability depending on $\mathbf{z}$ and $c\mathbf{s}$. As a result, the distribution of the final output $\mathbf{z}$ is independent of $\mathbf{s}$, which protects the secrecy of $\mathbf{s}$. The signature consists of challenge c and response $\mathbf{z}$. To verify it, the verifier recovers the commitment by $\mathbf{w} := \bar{\mathbf{A}}\mathbf{z} - c\mathbf{t}$ and checks whether $c = \mathsf{H}_{\mathrm{sig}}(\mathbf{t}, \mu, \mathbf{w})$.

DOTT follows the structure of mBCJ [BCJ08, DEF+19]. They utilize a homomorphic trapdoor commitment scheme to enable straight-line simulation. In the first round, each signer broadcasts nonce[1] $\mathbf{w}$ committed rather than in clear. The homomorphic property allows the signers to aggregate the commitments. In the second round, each signer opens its commitment in addition to broadcasts response $\mathbf{z}$. The reduction does not have to really decide $\mathbf{w}$ when it outputs the commitment in the first round. The trapdoor property allows it to open the commitment as any $\mathbf{w}$ of its choice later. It runs the standard simulation algorithm once the challenge is determined. In the second round,

[1] "Commitment" appears both in the context of Fiat-Shamir signatures and commitment schemes. To avoid ambiguity, here we use "nonce" to indicate the commitment $\mathbf{w}$ in signatures.

it opens its commitment as the nonce it obtains from the simulation. The authors proposed their scheme using a homomorphic trapdoor commitment scheme as a building block, while previously known instantiations of lattice based trapdoor commitment [GVW15, LNTW19] and their own instantiation [DOTT22] all rely on trapdoor sampling.

MuSig-L uses a similar structure to DWMS [AB21] and MuSig2 [NRS21], with a very different simulation technique. The signers exchange multiple pre-commitments in the first round. The actual individual commitment of each signer is a linear combination of its pre-commitments with coefficients derived from a hash function. In [BTT22b], the pre-commitment vectors $\mathbf{w}_1, \ldots, \mathbf{w}_m$ form a matrix $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_m]$. The reduction generates a trapdoor of the matrix so that it can sample a Gaussian preimage $\mathbf{b}$ satisfying $\mathbf{W}\mathbf{b} = \mathbf{w}'$ for any $\mathbf{w}'$. The reduction obtains a commitment $\mathbf{w}'$ when it runs the standard simulation. It then samples a preimage $\mathbf{b}$ and programs $\mathbf{b}$ into the random oracle as the coefficients of linear combination. Here trapdoor sampling allows the reduction to linearly combine $\mathbf{W}$ into any commitment of its choice and thus also delays the decision of the real commitment.

Performance of existing schemes. Now let us look at how the use of trapdoor sampling affects the performance of DOTT and MuSig-L. In Dilithium-G, a signature consists of a challenge and a response. The challenge is relatively small. The response $\mathbf{z}$ is a $(l + k)$-dimensional Gaussian distributed vector, where we typically set the $l \leq k$. In DOTT, a signature additionally contains an opening to a homomorphic trapdoor commitment to the k-dimensional nonce $\mathbf{w}$. In the instantiation of [DOTT22], the opening is a preimage given by trapdoor sampling of [MP12]. The trapdoor sampling requires a wide $k \times m$ matrix $\mathbf{W}$ with $m \approx k \log q$, where q is the modulus. The Gaussian widths of $\mathbf{z}$ and the preimage are not hugely different, and they only affect signature size by a logarithmic factor. On the other hand, the dimension $m \approx k \log q$ of a preimage is much larger than $\mathbf{z}$. Thus, the extra opening is notably larger than the original Dilithium-G signature.

In MuSig-L, each signer broadcasts a matrix $\mathbf{W}$ that enables trapdoor sampling instead of a single commitment vector. This increases the communication complexity by roughly $k \log q$ times. Moreover, pre-commitments $\mathbf{W}$ are commitments to Gaussian vectors, and they are combined with coefficients $\mathbf{b}$ which are again Gaussian. Both distributions need to have large enough Gaussian width to support their simulation technique. This significantly increases the Gaussian width of response $\mathbf{z}$ and affects signature size. Other parameters also have to grow to keep signature forgery hard, again increasing public-key and signature size.

1.3 Overview of Our Scheme

Observing the inefficiency of existing schemes caused by trapdoor sampling in simulation, our idea is to construct a scheme with trapdoor-free simulation. First let us look at a variant scheme of Dilithium-G. Now the secret key contains another short vector $\bar{\mathbf{u}} \in R^{l'+k}$, and the public key contains an additional matrix $\mathbf{B} \in R_q^{k \times l'}$. We also have $\mathbf{t} := \bar{\mathbf{A}}\mathbf{s} + \bar{\mathbf{B}}\bar{\mathbf{u}}$ with $\bar{\mathbf{B}} := [\mathbf{B}|\mathbf{I}]$. The signer samples two masking vectors $\mathbf{y}$ and $\mathbf{p}$ and computes the commitment $\mathbf{w} := \bar{\mathbf{A}}\mathbf{y} + \bar{\mathbf{B}}\mathbf{p}$. It computes two responses $\mathbf{z} := \mathbf{y} + c\mathbf{s}$ and $\mathbf{r} := \mathbf{p} + c\bar{\mathbf{u}}$ and performs rejection sampling separately. The signature consists of c, $\mathbf{z}$, and $\mathbf{r}$, and the verifier can recover the commitment by $\mathbf{w} := \bar{\mathbf{A}}\mathbf{z} + \bar{\mathbf{B}}\mathbf{r} - c\mathbf{t}$. This variant scheme can be viewed as a lattice-based analogue of Okamoto signature [Oka93]. Knowing any short enough $\mathbf{s}$ and $\bar{\mathbf{u}}$ satisfying $\bar{\mathbf{A}}\mathbf{s} + \bar{\mathbf{B}}\bar{\mathbf{u}} = \mathbf{t}$ is sufficient to produce a signature. In particular, the signer can set $\mathbf{s} = \mathbf{0}$ or $\bar{\mathbf{u}} = \mathbf{0}$.

In our protocol, matrix $\mathbf{B}$ is derived by hashing the aggregated public key and message μ. The signer signs in the special case of $\bar{\mathbf{u}} = \mathbf{0}$. When signing a common message μ, the signers derive the same matrix $\mathbf{B}$. Thus, the signers can exchange their commitment $\mathbf{w}$ and responses $\mathbf{z}$ and $\mathbf{r}$ and aggregate them by summing them up. This will give a correct multi-signature by linearity. More precisely, the signers obtain a common challenge c and takes $a_i c$ as their individual challenge where

a_i is the key aggregation coefficient derived from a hash function. Then it holds that

$$\tilde{\mathbf{w}} = \bar{\mathbf{A}}\tilde{\mathbf{z}} + \bar{\mathbf{B}}\tilde{\mathbf{r}} - c\tilde{\mathbf{t}},$$

where $\tilde{\mathbf{w}} = \sum_{i=1}^{n} \mathbf{w}_i$, $\tilde{\mathbf{z}} = \sum_{i=1}^{n} \mathbf{z}_i$, $\tilde{\mathbf{r}} = \sum_{i=1}^{n} \mathbf{r}_i$, and $\tilde{\mathbf{t}} = \sum_{i=1}^{n} a_i\mathbf{t}_i$ are the aggregated commitment/responses/public key.

In the security proof, the reduction can generate $\mathbf{B}$ together with a dual secret key $\bar{\mathbf{u}}$ satisfying $\bar{\mathbf{B}}\bar{\mathbf{u}} = \mathbf{t}$. Thus, it can perform straight-line simulation by signing in the special case of $\mathbf{s} = \mathbf{0}$. To generate a random $\mathbf{B}$ with dual secret key $\bar{\mathbf{u}}$, the reduction samples short vector $\mathbf{u} \in R^{l'-1+k}$ and lets $\mathbf{B} := [\mathbf{b}|\hat{\mathbf{B}}]$ with random chosen $\hat{\mathbf{B}}$ and $\mathbf{b} := \mathbf{t} - [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{u}$. It follows that

$$\bar{\mathbf{B}} \begin{bmatrix} 1 \\ \mathbf{u} \end{bmatrix} = [\mathbf{b}|\hat{\mathbf{B}}|\mathbf{I}] \begin{bmatrix} 1 \\ \mathbf{u} \end{bmatrix} = \mathbf{b} + [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{u} = \mathbf{t}.$$

Therefore, it can take $[1, \mathbf{u}^\intercal]^\intercal$ as the dual secret key $\bar{\mathbf{u}}$. Matrix $\mathbf{B}$ generated in this way is computationally indistinguishable from a uniformly random one based on MLWE.

While DOTT follows the structure of mBCJ [BCJ08, DEF+19], and MuSig-L follows the structure of DWMS [AB21] and MuSig2 [NRS21], our DualMS has an analogous structure to HBMS proposed by Bellare and Dai [BD21]. Nevertheless, the simulation techniques of the two schemes are noticeably different. Their reduction generates the hash-derived generator h (corresponding to $\mathbf{B}$) as a random combination of the common generator g (corresponding to $\mathbf{A}$) and public key X (corresponding to $\mathbf{t}$), and it gives two responses by solving two linear equations. However, in the lattice setting, solving random equations will unlikely give short responses. Thus, our "dual signing simulation" is crucial for a lattice-based scheme. Generation and indistinguishability of the dual key are also more indirect in lattice setting than discrete-logarithm setting.

In the formal specification of our scheme, we apply a simple and effective optimization. Note that $\mathbf{z}$ and $\mathbf{r}$ both contain an MLWE error term that will be multiplied by $\mathbf{I}$ in matrices $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ in the verification. Simply adding up two error terms can significantly reduce the signature size.

2 Preliminaries

Notation. For a positive number n, $[n]$ denotes $\{1, \ldots, n\}$. If x is a variable, then $y := x$ denotes that we assign the value of x to y. If D is a distribution, then $y \leftarrow D$ denotes that we sample y from D. If S is a set, then $y \leftarrow\!\!\$\, S$ denotes that we uniformly sample y from S. If f is a real-value function and S is a set, then $f(S)$ denotes $\sum_{x\in S} f(x)$.

2.1 Polynomial Rings and Discrete Gaussian Distribution

In this paper, most operations work over polynomial rings $R = \mathbb{Z}[X]/(f(X))$ and $R_q = \mathbb{Z}_q[X]/(f(X))$, where $f(X) = X^N + 1$ with N a power of two is the $2N$-th cyclotomic polynomial, and q is a prime that satisfies $q = 5 \mod 8$. Elements over the latter ring have coefficients between $-(q-1)/2$ and $(q-1)/2$. The L^p-norm for a vector of ring elements $\mathbf{v} = [\sum_{i=0}^{N-1} v_{1,i}X^i, \ldots, \sum_{i=0}^{N-1} v_{m,i}X^i]^\intercal \in R^m$ is defined as

$$\|\mathbf{v}\|_p = \|[v_{1,0}, \ldots, v_{1,N-1}, \ldots, v_{m,0}, \ldots, v_{m,N-1}]\|_p.$$

We need the following lemma about invertibility over R_q.

Lemma 1 ([LN17], Lemma 2.2)**.** *Let $N > 1$ be a power of 2 and q a prime congruent to 5 mod 8. The ring R_q has exactly $2q^{N/2} - 1$ elements without an inverse. Moreover, every non-zero polynomial $a \in R_q$ with $\|a\|_\infty < \sqrt{q}/2$ has an inverse.*

We define the key set $S_\eta \subset R$ as

$$S_\eta = \{x \in R : \|x\|_\infty \leq \eta\}$$

and the challenge set $C = C_\kappa \subset R$ as

$$C = \{c \in R : \|c\|_\infty = 1 \wedge \|c\|_1 = \kappa\}.$$

By Lemma 1, $c - c'$ has an inverse for any c, $c' \in C$ and $c \neq c'$.

The discrete Gaussian distribution over R^m is defined as follows.

Definition 1 (Discrete Gaussian Distribution over R^m)**.** For $\mathbf{x} \in R^m$, the Gaussian function of parameter $\mathbf{v} \in R^m$ and $s \in \mathbb{R}$ is defined as $\rho_{\mathbf{v},s}(\mathbf{x}) = \exp(-\pi\|\mathbf{x} - \mathbf{v}\|_2^2/s^2)$. The discrete Gaussian distribution $D^m_{\mathbf{v},s}$ centered at $\mathbf{v}$ is defined as

$$D^m_{\mathbf{v},s}(\mathbf{x}) = \frac{\rho_{\mathbf{v},s}(\mathbf{x})}{\rho_{\mathbf{v},s}(R^m)}.$$

In this paper, we omit the subscript $\mathbf{v}$ when $\mathbf{v} = \mathbf{0}$. For any $\epsilon > 0$, the *smoothing parameter* $\eta_\epsilon(\Lambda)$ [MR04] of lattice Λ is defined as the smallest $s > 0$ such that $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \epsilon$, where Λ^* is the dual lattice of Λ. By lemma 3.2 of [MR04], $\eta_\epsilon(R^m) \leq \sqrt{Nm}$ where $\epsilon = 2^{-Nm}$. The parameter s that we use in this paper exceeds $\eta_\epsilon(R^m)$ by a factor at least $\sqrt{2}$. In this setting, the following lemma holds, which is a special case of lemma 3.3 in [MP13]. We need the lemma to understand the distribution of the response in our multi-signature, which is the sum of individual responses.

Lemma 2. *Suppose $s \geq \sqrt{2} \cdot \eta_\epsilon(R^m)$ with a negligible ϵ. Let $\mathbf{x}_i$ for $i \in [n]$ be independent samples from $D^m_{s_i}$. Then the distribution of $\mathbf{x} = \sum_{i=1}^n \mathbf{x}_i$ is statistically close to D^m_s with $s = \sqrt{\sum_{i=1}^n s_i^2}$.*

The next two lemmas are important for Fiat-Shamir with aborts. Both of them are adapted from [Lyu12] by [DOTT22].

Lemma 3 ([Lyu12])**.** *For any $\gamma > 1$,*

$$\Pr\Big[\|\mathbf{z}\|_2 > \gamma(s/\sqrt{2\pi})\sqrt{mN} : \mathbf{z} \leftarrow D^m_s\Big] < \gamma^{mN} e^{mN(1-\gamma^2)/2}.$$

Lemma 4 ([Lyu12])**.** *Fix some t such that $t = \omega(\sqrt{\log(mN)})$ and $t = o(\log(mN))$. For any $\mathbf{v} \in R^m$, if $s \geq \sqrt{2\pi}\alpha\|\mathbf{v}\|_2$ for any positive α, then*

$$\Pr\big[M \cdot D^m_{\mathbf{v},s}(\mathbf{z}) \geq D^m_s(\mathbf{z}) : \mathbf{z} \leftarrow D^m_s\big] \geq 1 - \epsilon,$$

where $M = e^{t/\alpha + 1/(2\alpha^2)}$ and $\epsilon = 2e^{-t^2/2}$.

The following regularity result adapted from [LPR13] gives the minimum Gaussian width of $\mathbf{x}$ to make $[\mathbf{A}|\mathbf{I}]\mathbf{x}$ statistically close to the uniform distribution.

Lemma 5 ([LPR13])**.** *For positive integers k and l, suppose $m = l + k \leq \mathsf{poly}(N)$. let $\bar{\mathbf{A}} = [\mathbf{A}|\mathbf{I}] \in R_q^{k \times m}$, where $\mathbf{A}$ is uniformly distributed over $R_q^{k \times l}$. Then with probability $1 - 2^{-\Omega(N)}$ over the choice of $\mathbf{A}$, the distribution of $\bar{\mathbf{A}}\mathbf{x} \in R_q^k$, where $\mathbf{x} \leftarrow D^m_s$ with parameter $s > 2N \cdot q^{k/m + 2/(Nm)}$, 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.*

2.2 Assumptions

We restate the standard lattice hard problems, module short integer solution (MSIS) (in Hermite Normal Form) and learning with error (MLWE).

Definition 2 ($\mathsf{MSIS}_{q,k,l,\beta}$ problem)**.** The advantage of algorithm $\mathcal{A}$ against the $\mathsf{MSIS}_{q,k,l,\beta}$ problem is defined as

$$\mathbf{Adv}_{\mathsf{MSIS}_{q,k,l,\beta}}(\mathcal{A}) = \Pr\big[[\mathbf{A}|\mathbf{I}] \cdot \mathbf{x} = \mathbf{0} \land 0 < \|\mathbf{x}\|_2 \leq \beta : \mathbf{A} \leftarrow\!\!\$\, R_q^{k \times l}; \mathbf{x} \leftarrow \mathcal{A}(\mathbf{A}) \in R_q^{l+k}\big].$$

Definition 3 ($\mathsf{MLWE}_{q,k,l,\eta}$ problem)**.** The advantage of algorithm $\mathcal{A}$ against the $\mathsf{MLWE}_{q,k,l,\eta}$ problem is defined as

$$\begin{aligned}\mathbf{Adv}_{\mathsf{MLWE}_{q,k,l,\eta}}(\mathcal{A}) = &\Pr\big[\mathcal{A}(\mathbf{A},\mathbf{t}) = 1 : \mathbf{A} \leftarrow\!\!\$\, R_q^{k \times l}; \mathbf{s} \leftarrow\!\!\$\, S_\eta^{l+k}; \mathbf{t} := [\mathbf{A}|\mathbf{I}] \cdot \mathbf{s}\big] \\ &- \Pr\big[\mathcal{A}(\mathbf{A},\mathbf{t}) = 1 : (\mathbf{A},\mathbf{t}) \leftarrow\!\!\$\, R_q^{k \times l} \times R_q^k\big].\end{aligned}$$

2.3 Two-Round Multi-Signatures with Key Aggregation

Our definition of multi-signature schemes follows [NRS21]. The definition specially considers those signing protocols with the following features: 1) the signers interact with each other by broadcasting protocol messages round by round, 2) the round number is two, and 3) the final multi-signature is simply an aggregation of all second-round protocol messages. We regard the second-round protocol message as the individual signature of each signer. We describe the signing protocol as three algorithms Sign_1, Sign_2, and SAgg corresponding to three stages. They are locally run by each signer. The signers exchange their protocol messages between these stages. Algorithms Sign_1 and Sign_2 output a protocol message (for Sign_2 it is an individual signature) for the signer to broadcast, and Sign_2 and SAgg takes as inputs protocol messages from other signers. A multi-signature is finally output by SAgg. The signers keep states between Sign_1 and Sign_2, while SAgg does not take any secret state. Hence, any designated aggregator who collects the signatures can run SAgg to produce the multi-signature. The property of key-aggregation [MPSW19] allows to non-interactively aggregate public keys using a key aggregation algorithm KAgg. The verification algorithm takes as inputs an aggregated key instead of a list of individual keys.

Definition 4 (Two-round multi-signatures with key aggregation)**.** A two-round multi-signature scheme MS with key aggregation consists of algorithms with syntax defined as follows:

- $\mathsf{Setup}() \to \mathsf{pp}$: The parameter generation algorithm outputs a set of public parameters pp. Throughout, we assume pp is given as an implicit input to all other algorithms.
- $\mathsf{KGen}() \to (\mathsf{sk}, \mathsf{pk})$: The key generation algorithm outputs a secret key sk and a public key pk.
- $\mathsf{KAgg}(L) \to \mathsf{apk}$: The deterministic key aggregation algorithm takes as inputs a set of public keys $L = \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$ and outputs an aggregated public key apk.
- $\mathsf{Sign}_1(\mathsf{sk}_1, L, \mu) \to (\mathsf{st}_1, \mathsf{msg}_1)$: The first-stage signing algorithm takes as inputs a secret key sk_1, a set of public keys $L = \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$, and a message μ and outputs a state st_1 and a protocol message msg_1.
- $\mathsf{Sign}_2(\mathsf{st}_1, \{\mathsf{msg}_2, \ldots, \mathsf{msg}_n\}) \to \sigma_1$: The second-stage signing algorithm takes as inputs a state st_1 and a set of protocol messages $\{\mathsf{msg}_2, \ldots, \mathsf{msg}_n\}$ and outputs an individual signature σ_1.
- $\mathsf{SAgg}(\{\sigma_1, \ldots, \sigma_n\}) \to \tilde{\sigma}$: The signature aggregation (also the third-stage signing algorithm) takes as inputs a set of individual signatures $\{\sigma_1, \ldots, \sigma_n\}$ and outputs a multi-signature $\tilde{\sigma}$.
- $\mathsf{Vf}(\mathsf{apk}, \mu, \tilde{\sigma}) \to 0/1$: The deterministic verification algorithm takes as inputs an aggregated public key apk, a message μ, and a multi-signature $\tilde{\sigma}$ and outputs 0 or 1.

$\mathsf{Sign}(\{(\mathsf{sk}_1, \mathsf{pk}_1), \ldots, (\mathsf{sk}_n, \mathsf{pk}_n)\}, \mu)$

$L := \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$
for $i \in [n]$ **do** $(\mathsf{st}_i, \mathsf{msg}_i) \leftarrow \mathsf{Sign}_1(\mathsf{sk}_i, L, \mu)$
for $i \in [n]$ **do** $\sigma_i \leftarrow \mathsf{Sign}_2(\mathsf{st}_i, \{\mathsf{msg}_j\}_{j \in [n] \setminus \{i\}})$
$\tilde{\sigma} := \mathsf{SAgg}(\{\sigma_1, \ldots, \sigma_n\})$
return $\tilde{\sigma}$

Figure 1: Algorithm Sign that defines completeness.

Definition 5 (ε-completeness)**.** Let algorithm Sign be as described in Fig. 1. A two-round multi-signature scheme MS with key aggregation is said to be ε-complete if fixing any positive integer n, any $\mathsf{pp} \in \mathsf{Setup}()$, any $(\mathsf{sk}_i, \mathsf{pk}_i) \in \mathsf{KGen}()$ for $i \in [n]$, and any $\mu \in \{0,1\}^*$,

$$\Pr[\mathsf{Vf}(\mathsf{KAgg}(L), \mu, \tilde{\sigma}) = 1 : \tilde{\sigma} \leftarrow \mathsf{Sign}(\{(\mathsf{sk}_1, \mathsf{pk}_1), \ldots, (\mathsf{sk}_n, \mathsf{pk}_n)\}, \mu)] \geq \varepsilon,$$

where $L = \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$.

Below we define the unforgeability of a multi-signature scheme. In the security game, adversary $\mathcal{A}$ is given a target public key pk_1. Its goal is to forge a multi-signature under a public-key list L^* of its choice while required to contain pk_1. In a chosen-message attack game, $\mathcal{A}$ can concurrently launch many signing sessions with an honest signer having public key pk_1. In each session, $\mathcal{A}$ plays the part of all other signers, with public keys of its choices. To formalize the chosen-message attack, $\mathcal{A}$ has the access to two signing oracles $\textsc{Sign}_1$ and $\textsc{Sign}_2$, corresponding to the first and the second stages of the signing protocol. $\textsc{Sign}_1$ takes necessary inputs for launching a signing session, i.e., a public key list and a message to sign. It returns the first-round protocol message of the honest signer. $\textsc{Sign}_2$ takes as inputs the first-round protocol messages from other signers and outputs the individual signature of the honest signer. States are kept between $\textsc{Sign}_1$ and $\textsc{Sign}_2$. When $\mathcal{A}$ calls $\textsc{Sign}_2$, it is required to specify a session ID sid to indicate which session it wants to proceed. Since SAgg involves no secret state, there is no need for a corresponding oracle. Note that in the setting of multi-signatures, $\mathcal{A}$ can win by forging a signature it has queried under different public-key lists from L^*.

Definition 6 (Unforgeablility against chosen-message/key-only attack)**.** The advantage of adversary $\mathcal{A}$ against the *unforgeability against chosen-message attack (UF-CMA)* of a multi-signature scheme MS in the ROM is defined as

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{MS}}(\mathcal{A}) = \Pr[\mathsf{UF\text{-}CMA}_{\mathsf{MS}}(\mathcal{A}) = 1],$$

where the game $\mathsf{UF\text{-}CMA}_{\mathsf{MS}}$ is described in Fig. 2. The *unforgeability against key-only attack (UF-KOA)* is defined the same as UF-CMA except that $\mathcal{A}$ does not have the access to $\textsc{Sign}_1$ and $\textsc{Sign}_2$.

3 Our DualMS Scheme

3.1 Scheme Description

Fig. 3 describes our DualMS scheme. The parameters are listed in Table 1. We explain our construction as below.

$\mathsf{UF\text{-}CMA}_{\mathsf{MS}}(\mathcal{A})$

$\mathsf{pp} \leftarrow \mathsf{Setup}()$
$(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow \mathsf{KGen}()$
$\mathsf{ctr} := 0$
$\mathcal{S} := \emptyset$; $\mathcal{Q} := \emptyset$
$(L^*, \mu^*, \tilde{\sigma}^*) \leftarrow \mathcal{A}^{\textsc{Sign}_1, \textsc{Sign}_2, \mathsf{H}}(\mathsf{pp}, \mathsf{pk}_1)$
if $(\mathsf{pk}_1 \notin L^*) \vee (L^*, \mu^*) \in \mathcal{Q}$ **then**
  return 0
return $\mathsf{Vf}(\mathsf{KAgg}(L^*), \mu^*, \tilde{\sigma}^*)$

$\textsc{Sign}_1(\{\mathsf{pk}_2, \ldots, \mathsf{pk}_n\}, \mu)$

$\mathsf{ctr} := \mathsf{ctr} + 1$
$\mathsf{sid} := \mathsf{ctr}$; $\mathcal{S} := \mathcal{S} \cup \{\mathsf{sid}\}$
$L := \{\mathsf{pk}_1, \ldots, \mathsf{pk}_n\}$
$\mathcal{Q} := \mathcal{Q} \cup \{(L, \mu)\}$
$(\mathsf{msg}_1, \mathsf{st}_{\mathsf{sid}}) \leftarrow \mathsf{Sign}_1(\mathsf{sk}_1, L, \mu)$
return msg_1

$\textsc{Sign}_2(\mathsf{sid}, \{\mathsf{msg}_2, \ldots, \mathsf{msg}_n\})$

if $\mathsf{sid} \notin \mathcal{S}$ **then return** $\perp$
$\sigma_1 \leftarrow \mathsf{Sign}_2(\mathsf{st}_{\mathsf{sid}}, \{\mathsf{msg}_2, \ldots, \mathsf{msg}_n\})$
$\mathcal{S} := \mathcal{S} \setminus \{\mathsf{sid}\}$
return σ_1

Figure 2: The UF-CMA security game against multi-signature scheme MS in the ROM, where H denotes the random oracle.

Setup, key generation, and key aggregation. In the setup stage, a matrix $\bar{\mathbf{A}} := [\mathbf{A}|\mathbf{I}] \in R_q^{k\times(l+k)}$ is generated as a public parameter with $\mathbf{A}$ uniformly chosen from $R_q^{k\times l}$. The secret key $\mathbf{s}$ of each signer is a short vector uniformly chosen from S_η^{l+k}. Recall that η is the maximum L^∞-norm. The public key is $\mathbf{t} := \bar{\mathbf{A}}\mathbf{s}$. Note that $\mathbf{A}$ and $\mathbf{t}$ constitute a MLWE sample, which ensures the secrecy of $\mathbf{s}$. The key aggregation algorithm aggregates a list of public keys $L = \{\mathbf{t}_1, \ldots, \mathbf{t}_n\}$ into an aggregated public key. We use a hash function $\mathsf{H}_{\mathrm{agg}}$ to compute a small polynomial $a_i := \mathsf{H}_{\mathrm{agg}}(L, \mathbf{t}_i) \in C$ for each $\mathbf{t}_i \in L$. Then L is aggregated into $\tilde{\mathbf{t}} := \sum_{i=1}^n a_i \mathbf{t}_i$. Here L is an unordered set, and duplicate keys $\mathbf{t}_i = \mathbf{t}_j$ will make $a_i = a_j$.

Signature generation. Now we describe the signing protocol of DualMS. In the protocol, each signer runs the same procedure, so we describe the protocol by showing the behavior of one signer. We assign the signer index 1. It has secret key $\mathbf{s}_1$ and public key $\mathbf{t}_1$. First, the signer computes the aggregated key $\tilde{\mathbf{t}} := \mathsf{KAgg}(L)$. Then it uses a hash function $\mathsf{H}_{\mathrm{com}}$ to derive a matrix $\mathbf{B} := \mathsf{H}_{\mathrm{com}}(\tilde{\mathbf{t}}, \mu) \in R_q^{k\times l'}$ and lets $\bar{\mathbf{B}} := [\mathbf{B}|\mathbf{I}] \in R_q^{l'+k}$. It computes its commitment $\mathbf{w}_1 := \bar{\mathbf{A}}\mathbf{y}_1 + \bar{\mathbf{B}}\mathbf{r}_1$ with $\mathbf{y}_1 \leftarrow D_s^{l+k}$ and $\mathbf{r}_1 \leftarrow D_{s'}^{l'+k}$. It broadcasts $\mathbf{w}_1$ to other signers as its first-round protocol message.

Once the signer receives all commitments $\mathbf{w}_2, \ldots, \mathbf{w}_n$ from the other signers, it aggregates them with its own commitment into an aggregated commitment $\tilde{\mathbf{w}} := \sum_{i=1}^n \mathbf{w}_i$. Then it uses a hash function $\mathsf{H}_{\mathrm{sig}}$ to derive a short challenge $c := \mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}}) \in C$. It computes its response $\mathbf{z}_1 := \mathbf{y}_1 + a_1 c \mathbf{s}_1$ where $a_1 = \mathsf{H}_{\mathrm{agg}}(L, \mathbf{t}_1)$. Here the distribution of $\mathbf{z}_1$ is discrete Gaussian centered at $a_1 c \mathbf{s}_1$ depending on secret key $\mathbf{s}_1$. Following the FSwA paradigm [Lyu09, Lyu12], the signer runs a rejection sampling. Namely, it aborts except with probability $\min(1, D_s^{l+k}(\mathbf{z}_1)/(M \cdot D_{a_1 c\mathbf{s}_1, s}^{l+k}(\mathbf{z}_1)))$. As a result, when it passes the rejection sampling, the distribution of $\mathbf{z}_1$ will center at $\mathbf{0}$, and thus $\mathbf{s}_1$ keeps secret. See a formal analysis in Section 4.1. The signer broadcasts $(c, \mathbf{z}_1, \mathbf{r}_1)$ as its individual signature if it does not abort. Otherwise, the signers may restart the protocol until no signer aborts here.

Finally, if all individual signatures have the same challenge c, then they can be aggregated into a multi-signature. This aggregating procedure does not involve any secret states of the signers and thus can be executed by a designated aggregator rather than the signers. The aggregator splits each $\mathbf{z}_i$ into $\mathbf{z}_i' \in R^l$ and $\mathbf{z}_i'' \in R^k$ and similarly, $\mathbf{r}_i$ into $\mathbf{r}_i' \in R^{l'}$ and $\mathbf{r}_i'' \in R^k$. Then it aggregates $\mathbf{z}_1', \ldots, \mathbf{z}_n'$ into $\tilde{\mathbf{z}}$, $\mathbf{r}_1', \ldots, \mathbf{r}_n'$ into $\tilde{\mathbf{r}}$, and the remaining k-dimensional vectors into $\tilde{\mathbf{e}}$.

$\underline{\mathsf{Setup}()}$

$\mathbf{A} \leftarrow\!\!\$\ R_q^{k\times l}$

$\bar{\mathbf{A}} := [\mathbf{A}|\mathbf{I}]$

return $\bar{\mathbf{A}}$

$\underline{\mathsf{KGen}()}$

$\mathbf{s} \leftarrow\!\!\$\ S_\eta^{l+k}$

$\mathbf{t} := \bar{\mathbf{A}}\mathbf{s}$

return $(\mathbf{s}, \mathbf{t})$

$\underline{\mathsf{KAgg}(L)}$

$\{\mathbf{t}_1, \ldots, \mathbf{t}_n\} := L$

for $i \in [n]$ **do**

  $a_i := \mathsf{H}_{\text{agg}}(L, \mathbf{t}_i) \in C$

$\tilde{\mathbf{t}} := \sum_{i=1}^n a_i \mathbf{t}_i$

return $\tilde{\mathbf{t}}$

$\underline{\mathsf{Vf}(\mathsf{apk}, \mu, \tilde{\sigma})}$

$\tilde{\mathbf{t}} := \mathsf{apk}$

$(c, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{e}}) := \tilde{\sigma}$

$\mathbf{B} := \mathsf{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu)$

$\tilde{\mathbf{w}} := \mathbf{A}\tilde{\mathbf{z}} + \mathbf{B}\tilde{\mathbf{r}} + \tilde{\mathbf{e}} - c\tilde{\mathbf{t}}$

return $[\![\|\tilde{\mathbf{z}}\|_2 \leq B_n \wedge \|\tilde{\mathbf{r}}\|_2 \leq B'_n$

  $\wedge \|\tilde{\mathbf{e}}\|_2 \leq B''_n \wedge \mathsf{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}}) = c]\!]$

$\underline{\mathsf{Sign}_1(\mathsf{sk}_1, L, \mu)}$

$\mathbf{s}_1 := \mathsf{sk}_1$

$\{\mathbf{t}_1, \ldots, \mathbf{t}_n\} := L$

$a_1 := \mathsf{H}_{\text{agg}}(L, \mathbf{t}_1)$

$\tilde{\mathbf{t}} := \mathsf{KAgg}(L)$

$\mathbf{B} := \mathsf{H}_{\text{com}}(\tilde{\mathbf{t}}, \mu) \in R_q^{k\times l'}$

$\bar{\mathbf{B}} := [\mathbf{B}|\mathbf{I}] \in R_q^{k\times(l'+k)}$

$\mathbf{y}_1 \leftarrow D_s^{l+k}$

$\mathbf{r}_1 \leftarrow D_{s'}^{l'+k}$

$\mathbf{w}_1 := \bar{\mathbf{A}}\mathbf{y}_1 + \bar{\mathbf{B}}\mathbf{r}_1 \in R_q^k$

return $((\mathbf{s}_1, \tilde{\mathbf{t}}, \mu, a_1, \mathbf{y}_1, \mathbf{r}_1, \mathbf{w}_1), \mathbf{w}_1)$

$\underline{\mathsf{Sign}_2(\mathsf{st}_1, \{\mathsf{msg}_2, \ldots, \mathsf{msg}_n\})}$

$(\mathbf{s}_1, \tilde{\mathbf{t}}, \mu, a_1, \mathbf{y}_1, \mathbf{r}_1, \mathbf{w}_1) := \mathsf{st}_1$

for $i = 2, \ldots, n$ **do** $\mathbf{w}_i := \mathsf{msg}_i$

$\tilde{\mathbf{w}} := \sum_{i=1}^n \mathbf{w}_i$

$c := \mathsf{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}}) \in C$

$\mathbf{z}_1 := \mathbf{y}_1 + a_1 c \mathbf{s}_1$

With prob. $\min(1, D_s^{l+k}(\mathbf{z}_1)/(M \cdot D_{a_1 c\mathbf{s}_1, s}^{l+k}(\mathbf{z}_1)))$:

  return $(c, \mathbf{z}_1, \mathbf{r}_1)$

Otherwise:

  return $\perp$

$\underline{\mathsf{SAgg}(\{\sigma_1, \ldots, \sigma_n\})}$

for $i = 1, \ldots, n$ **do** $(c_i, \mathbf{z}_i, \mathbf{r}_i) := \sigma_i$

if $\exists i \in [n], c_i \neq c_1$ **then return** $\perp$

for $i = 1, \ldots, n$ **do**

  $[\mathbf{z}_i'^{\intercal}, \mathbf{z}_i''^{\intercal}] := \mathbf{z}_i^{\intercal}$

  $[\mathbf{r}_i'^{\intercal}, \mathbf{r}_i''^{\intercal}] := \mathbf{r}_i^{\intercal}$

$\tilde{\mathbf{z}} := \sum_{i=1}^n \mathbf{z}'_i \in R^l$

$\tilde{\mathbf{r}} := \sum_{i=1}^n \mathbf{r}'_i \in R^{l'}$

$\tilde{\mathbf{e}} := \sum_{i=1}^n (\mathbf{z}''_i + \mathbf{r}''_i) \in R^k$

return $(c_1, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{e}})$

Figure 3: Our DualMS scheme.

In the aggregation procedure, we observe that $\mathbf{z}_i$ and $\mathbf{r}_i$ both contain a k-dimensional MLWE error term that will be multiplied by $\mathbf{I}$ for verification. We therefore optimize the scheme by aggregating all the k-dimensional error terms. Compared to directly aggregating $\mathbf{z}_i$'s and $\mathbf{r}_i$'s separately, the optimization cuts k dimensions from the final signature. Note that we only apply the optimization in the very last step. Alternatively, we can also let each signer just produce one error vector at the beginning. That will further improve efficiency a bit while complicate the presentation hereinafter. See Appendix C for a more detailed discussion.

Parameter	Description
n	Number of parties
N	A power of two defining the degree of $f(X)$
$f(X) = X^N + 1$	The $2N$-th cyclotomic polynomial
$q = 5 \mod 8$	Prime modulus
$R = \mathbb{Z}[X]/f(X)$	Cyclotomic ring
$R_q = \mathbb{Z}_q[X]/f(X)$	Ring
k	The height of random matrix $\mathbf{A}$
l	The width of random matrix $\mathbf{A}$
$m = l + k \leq \mathsf{poly}(N)$	The width of matrix $\bar{\mathbf{A}}$
l'	The width of matrix $\mathbf{B}$ given by H_{com}
$m' = l' + k \leq \mathsf{poly}(N)$	The width of matrix $\bar{\mathbf{B}}$
γ	Parameters defining the tail bound of Lemma 3
$B = \gamma(s/\sqrt{2\pi})\sqrt{Nl}$	The maximum L^2-norm of response $\mathbf{z}_1'$
$B_n = \sqrt{n}B$	The maximum L^2-norm of aggregated response $\tilde{\mathbf{z}}$
$B' = \gamma(s'/\sqrt{2\pi})\sqrt{Nl'}$	The maximum L^2-norm of response $\mathbf{r}_1'$
$B_n' = \sqrt{n}B'$	The maximum L^2-norm of aggregated response $\tilde{\mathbf{r}}$
$B'' = \gamma(\sqrt{s^2 + s'^2}/\sqrt{2\pi})\sqrt{Nk}$	The maximum L^2-norm of error $\mathbf{z}_1'' + \mathbf{r}_1''$
$B_n'' = \sqrt{n}B''$	The maximum L^2-norm of aggregated error $\tilde{\mathbf{r}}$
κ	The maximum L^1-norm of challenge vector c
$C = \{c \in R : \|c\|_\infty = 1 \wedge \|c\|_1 = \kappa\}$	Challenge set where $\lvert C\rvert = \binom{N}{\kappa}2^\kappa$
η	The maximum L^∞-norm of the secret $\mathbf{s}$
S_η	Key set
$T = \kappa^2\eta\sqrt{Nm}$	The maximum L^2-norm of $a_1 c\mathbf{s}_1$
η'	The maximum L^∞-norm of the secret $\mathbf{u}$
$S_{\eta'}$	Dual key set
$T' = \kappa^2\eta'\sqrt{Nm'}$	The maximum L^2-norm of $a_1 c\bar{\mathbf{u}}$
α	Parameter defining s and M based on Lemma 4
$t = \omega(\sqrt{\log(N)}) \wedge t = o(\log(N))$	Parameter defining M based on Lemma 4
$s > \max(\sqrt{2\pi}\alpha T, 2N \cdot q^{k/m+2/(Nm)})$	Deviation parameter of the Gaussian distribution of $\mathbf{y}_1$
$s' > \max(\sqrt{2\pi}\alpha T', 2N \cdot q^{k/m'+2/(Nm')})$	Deviation parameter of the Gaussian distribution of $\mathbf{r}_1$
$M = e^{t/\alpha + 1/(2\alpha^2)}$	The expected number of restarts of a single party
$M_n = M^n$	The expected number of restarts of all n parties

Table 1: Parameters of DualMS.

Verification. Given an aggregated key $\tilde{\mathbf{t}}$, a message μ and a multi-signature $(c, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{e}})$, the verifier recovers the aggregated commitment $\tilde{\mathbf{w}} := \mathbf{A}\tilde{\mathbf{z}} + \mathbf{B}\tilde{\mathbf{r}} + \tilde{\mathbf{e}} - c\tilde{\mathbf{t}}$. It then verifies that $c = \mathsf{H}_{\text{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ and that $\tilde{\mathbf{z}}$, $\tilde{\mathbf{r}}$, and $\tilde{\mathbf{e}}$ are short enough. We will show the completeness of DualMS in the next subsection.

Simulation. We will give a formal security proof for DualMS in the next section. Here let us briefly sketch how the reduction performs straight-line simulation. When the adversary queries H_{com}, the reduction answers with $\mathbf{B} := [\mathbf{b}|\hat{\mathbf{B}}]$, where $\hat{\mathbf{B}}$ is uniformly chosen from $R_q^{k\times(l'-1)}$ and $\mathbf{b} := \mathbf{t}_1 - [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{u}$ with $\mathbf{u} \leftarrow\$ S_{\eta'}^{l'-1+k}$. Consequently, the reduction knows a dual secret key $\bar{\mathbf{u}} := [1, \mathbf{u}^\intercal]^\intercal$ satisfying $\bar{\mathbf{B}}\bar{\mathbf{u}} = \mathbf{t}_1$.

In the signing protocol, the reduction computes its commitment as $\mathbf{w}_1 := \bar{\mathbf{A}}\mathbf{z}_1 + \bar{\mathbf{B}}\mathbf{p}$ with $\mathbf{z}_1 \leftarrow D_s^{l+k}$ and $\mathbf{p} \leftarrow D_{s'}^{l'+k}$. In the second stage, the reduction generates its individual signature with $\mathbf{r}_1 := \mathbf{p} + a_1 c\bar{\mathbf{u}}$. The reduction also performs rejection sampling here to keep $\bar{\mathbf{u}}$ secret.

3.2 Correctness and Number of Repetitions

We show that DualMS is correct and the expected number of restarts is approximately M_n, i.e., DualMS is ε-complete with $\varepsilon \approx 1/M_n$.

We need the following results implied by Lemma 8 in Section 4.1: conditioned on any commitment $\mathbf{w}_1$ sent out in the first round and any a_1, c: 1) the signer passes rejection sampling with probability approximately $1/M$; 2) the distribution of $\mathbf{z}_1$ is statistically close to D_s^{l+k}. The fact that these results hold for any $\mathbf{w}_1$, a_1, and c_1 means that no malicious signer can affect them. We know immediately from the first result that all signers pass rejection sampling together with probability about $1/M_n$.

Then we show that if no signer aborts, then the signing protocol outputs a valid multi-signature except with small probability bounded by Lemma 3. Consider each individual signature produced by the signing protocol, for all $i \in [n]$ we have

$$\mathbf{w}_i = \bar{\mathbf{A}}\mathbf{y}_i + \bar{\mathbf{B}}\mathbf{r}_i = \bar{\mathbf{A}}(\mathbf{z}_i - a_i c\mathbf{s}_i) + \bar{\mathbf{B}}\mathbf{r}_i = \bar{\mathbf{A}}\mathbf{z}_i + \bar{\mathbf{B}}\mathbf{r}_i - a_i c\mathbf{t}_i = \mathbf{A}\mathbf{z}'_i + \mathbf{B}\mathbf{r}'_i + \mathbf{z}''_i + \mathbf{r}''_i - a_i c\mathbf{t}_i.$$

For the multi-signature given by the protocol, we have

$$\tilde{\mathbf{w}} = \sum_{i=1}^{n} \mathbf{w}_i = \mathbf{A}\sum_{i=1}^{n} \mathbf{z}'_i + \mathbf{B}\sum_{i=1}^{n} \mathbf{r}'_i + \sum_{i=1}^{n}(\mathbf{z}''_i + \mathbf{r}''_i) - c\sum_{i=1}^{n} a_i\mathbf{t}_i = \mathbf{A}\tilde{\mathbf{z}} + \mathbf{B}\tilde{\mathbf{r}} + \tilde{\mathbf{e}} - c\tilde{\mathbf{t}}.$$

Thus, the verifier correctly recovers the aggregated commitment given such a multi-signature, so the condition $c = \mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ always holds.

It remains to consider conditions $\|\tilde{\mathbf{z}}\|_2 \le B_n$, $\|\tilde{\mathbf{r}}\|_2 \le B'_n$, and $\|\tilde{\mathbf{e}}\|_2 \le B''_n$. For all $i \in [n]$, we know that $\mathbf{r}_i$ is sampled from $D_{s'}^{l'+k}$, and by Lemma 8, the distribution of $\mathbf{z}_i$ is statistically close to D_s^{l+k}. By Lemma 2, the distributions of $\tilde{\mathbf{z}}$, $\tilde{\mathbf{r}}$, and $\tilde{\mathbf{e}}$ are statistically close to $D^l_{s\sqrt{n}}$, $D^{l'}_{s'\sqrt{n}}$, and $D^k_{\sqrt{n(s^2+s'^2)}}$ respectively. Then Lemma 3 bounds the probability that $\|\tilde{\mathbf{z}}\|_2$, $\|\tilde{\mathbf{r}}\|_2$, and $\|\tilde{\mathbf{e}}\|_2$ exceed B_n, B'_n, B''_n, respectively. Setting the parameter γ in Lemma 3 as 1.1 gives a reasonably small probability, and $\gamma = \sqrt{3}$ is enough to yield an unnecessarily small bound $(\sqrt{3}/e)^{Nm}$.

3.3 Security

We have the following security result for DualMS.

Theorem 6 (UF-CMA of DualMS). *For any τ-time adversary $\mathcal{A}$ against the UF-CMA of* DualMS *that makes at most Q_h queries to each random oracle and launches at most Q_s sessions with the signing oracles, there exist algorithms $\mathcal{B}$, $\mathcal{D}$, and $\mathcal{D}'$ such that*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{DualMS}}(\mathcal{A}) \le Q_h \cdot \Bigg(& \sqrt{\frac{Q_h^2}{|C|} + Q_h\sqrt{Q_h \mathbf{Adv}_{\mathsf{MSIS}_{q,k,1+l+l',\beta}}(\mathcal{B})} \\ & + \mathbf{Adv}_{\mathsf{MLWE}_{q,k,l,\eta}}(\mathcal{D}) + (Q_h - 1)\mathbf{Adv}_{\mathsf{MLWE}_{q,k,l'-1,\eta'}}(\mathcal{D}')} \\ & + Q_s(\frac{3\epsilon}{2M} + 2^{-\Omega(N)}) + \frac{2Q_h^2}{|C|} + 3(\frac{2}{q^{N/2}})^k \Bigg),\end{aligned}$$

where $\beta = 8\kappa\sqrt{\hat{n}^2\kappa^3 + B_n^2 + B_n'^2 + B_n''^2}$, $\hat{n}$ is the maximum number of duplicate keys in a public key list, $\epsilon = 2e^{-t^2/2}$, t is a parameter as specified in Table 1, and the running time of $\mathcal{B}$, $\mathcal{D}$, and $\mathcal{D}'$ are essentially 4τ, τ, and τ, respectively.

Let us explain the security guarantees given by this theorem. We can choose t to make ϵ and hence the term $Q_{\mathrm{s}}(2\epsilon/M + 2^{-\Omega(N)})$ small enough. The term $3(2/q^{N/2})^k$ is clearly small. We also set κ to make $|C|$ large enough. A common setting when $N = 256$ is $\kappa = 60$, which gives $|C| > 2^{256}$. Among the two $Q_{\mathrm{h}}^2/|C|$ terms in the formula, the square-rooted one is dominant. Due to the quadratic loss and the outer factor Q_{h}, $|C| > 2^{256}$ only permits at most 64 bits of security (i.e., the adversary need $Q_{\mathrm{h}} \geq 2^{64}$ to achieve constant success probability). It remains to set the parameters to make $\mathsf{MSIS}_{q,k,1+l+l',\beta}$, $\mathsf{MLWE}_{q,k,l,\eta}$, and $\mathsf{MLWE}_{q,k,l'-1,\eta'}$ hard enough. Note that $\mathbf{Adv}_{\mathsf{MSIS}_{q,k,1+l+l',\beta}}(\mathcal{B})$ is also affected by quadratic and multiplicative loss. For about 64-bit security, we need $\mathbf{Adv}_{\mathsf{MSIS}_{q,k,1+l+l',\beta}}(\mathcal{B}) \approx 2^{-448}$. $\mathbf{Adv}_{\mathsf{MLWE}_{q,k,l'-1,\eta'}}(\mathcal{D}')$ is affected by an extra factor $(Q_{\mathrm{h}}-1)$ compared to $\mathbf{Adv}_{\mathsf{MLWE}_{q,k,l,\eta}}(\mathcal{D})$. They are required to be about 2^{-128} and 2^{-64} respectively.

Our scheme allows duplicate public keys unlike [BTT22b]. We consider the number of duplicates as an extra parameter $\hat{n}$. By doing this we can more accurately show how security is affected by allowing duplicates.

An important type of attacks against multi-signature schemes is to concurrently launch many signing sessions and linearly combine those signatures from the honest signer into a forged one [DEF+19, BLL+21]. Our scheme resists such attacks for a similar reason to [DEF+19, BD21, DOTT22]. Linear combination works only when matrix $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ are both fixed. However, different messages lead to different matrix $\bar{\mathbf{B}}$ with high probability, which prevents the attacker from forging signatures on new messages.

4 Proof of Security

In this section, we prove Theorem 6.

Assumptions about random oracle queries. Before we begin our proofs, let us make the following assumptions about the adversary's random oracle queries.

- The adversary queries $\mathsf{H}_{\mathrm{com}}(\tilde{\mathbf{t}}, \mu)$ before it queries $\mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ for any $\tilde{\mathbf{w}}$.
- The adversary queries $\mathsf{H}_{\mathrm{agg}}(L, \mathbf{t}_i)$ for every $\mathbf{t}_i \in L$ before it queries $\textsc{Sign}_1(\{\mathbf{t}_2, \ldots, \mathbf{t}_n\}, \mu)$, where $L = \{\mathbf{t}_1^*, \mathbf{t}_2, \ldots, \mathbf{t}_n\}$.
- The adversary queries $\mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ before it queries $\textsc{Sign}_2(\mathsf{sid}, \{\mathbf{w}_2, \ldots, \mathbf{w}_n\})$, where $\tilde{\mathbf{t}}$ is the aggregated public key corresponding to that signing session, $\tilde{\mathbf{w}} = \sum_{i=1}^{n} \mathbf{w}_i$, and $\mathbf{w}_1$ is the nonce returned by $\textsc{Sign}_1$ in that session.
- The adversary queries all hash queries that related to its forgery (i.e., all queries that will be made in verification) before it outputs the forgery.

These assumptions are without loss of generality in the sense that given an arbitrary adversary $\mathcal{A}$ making at most Q_{h} queries to each random oracles and launching Q_{s} sessions with the signing oracles, we can easily construct an adversary $\mathcal{A}'$ as a "random oracle middle man" that satisfies the assumptions, wins with the same probability as $\mathcal{A}$, and makes at most $2Q_{\mathrm{h}} + n(Q_{\mathrm{s}} + 1)$ queries to each random oracles. It is reasonable to consider Q_{h} as the dominant term, as Q_{h} is related to the local computation time of a real-world attacker. Hence, the security loss introduced here is not essential.

Selective security. We prove Theorem 6 in a modular way. We first reduce UF-CMA to *selective UF-CMA (sel-UF-CMA)* (Lemma 7), then sel-UF-CMA to *selective UF-KOA (sel-UF-KOA)* and MLWE (Lemma 9), and finally sel-UF-KOA to MSIS and MLWE (Lemma 10). Let us define the sel-UF-CMA and the sel-UF-KOA of our DualMS. In the selective security game, the adversary selects at the beginning the index of a $\mathsf{H}_{\mathrm{com}}$ query, and its goal is to forge a multi-signature corresponding to that $\mathsf{H}_{\mathrm{com}}$ query. Precisely, the $\mathsf{sel\text{-}UF\text{-}CMA}_{\mathsf{DualMS}}$ security game has the following differences from $\mathsf{UF\text{-}CMA}_{\mathsf{DualMS}}$. The adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is split into two stages. The first stage $\mathcal{A}_1$ outputs an index i^* without making any oracle queries. The second stage $\mathcal{A}_2$ has the access to the random oracles and the signing oracles and outputs its forgery but without a message (namely, it outputs a public-key list L^* and a multi-signature $\tilde{\sigma}^*$). Suppose the i^*-th $\mathsf{H}_{\mathrm{com}}$ query is $\mathsf{H}_{\mathrm{com}}(\tilde{\mathbf{t}}^*, \mu^*)$. For $\mathcal{A}$ to win, we require that L^* includes target public key $\mathbf{t}_1^*$, $\mathcal{A}$ has not queried (L^*, μ^*) to the signing oracle, L^* is aggregated into $\tilde{\mathbf{t}}^*$, and $\mathsf{Vf}(\tilde{\mathbf{t}}^*, \mu^*, \tilde{\sigma}^*) = 1$. Also see the definition in Fig. 5 where G_0 is exactly $\mathsf{sel\text{-}UF\text{-}CMA}_{\mathsf{DualMS}}$. The $\mathsf{sel\text{-}UF\text{-}KOA}_{\mathsf{DualMS}}$ security game is $\mathsf{sel\text{-}UF\text{-}CMA}_{\mathsf{DualMS}}$ without the signing oracles.

Apparently, UF-CMA reduces to sel-UF-CMA with factor Q_{h} loss of success probability. A reduction that guesses $i^* \in [Q_{\mathrm{h}}]$ at the beginning is sufficient to prove that.

Lemma 7 (UF-CMA to sel-UF-CMA)**.** *For any τ-time adversary $\mathcal{A}$ against the UF-CMA of* DualMS *that makes at most Q_h queries to each random oracle, there exists an adversary $\mathcal{B}$ such that $\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{DualMS}}(\mathcal{A}) \le Q_h \cdot \mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}CMA}}_{\mathsf{DualMS}}(\mathcal{B})$ and the running time of $\mathcal{B}$ is essentially τ.*

4.1 Straight-Line Simulation

This subsection is a preparation for reducing sel-UF-CMA to sel-UF-KOA. We bound the statistical distance between the output distributions of the normal signing oracle and the simulated one. Note that an adversary can query $\textsc{Sign}_1$ to obtain a commitment $\mathbf{w}$ and then choose what challenge c it wants $\textsc{Sign}_2$ to respond according to $\mathbf{w}$. We have to prevent the adversary from distinguishing the output distributions of $\textsc{Sign}_2$ with strategically chosen c. Therefore, we need to analyze: 1) the distributions of $\mathbf{w}$ output by $\textsc{Sign}_1$ and 2) the distributions of $\mathbf{z}$ and $\mathbf{r}$ output by $\textsc{Sign}_2$ for any c, conditioned on any $\mathbf{w}$ output by $\textsc{Sign}_1$ in the same session.

We define two procedures Trans and Sim in the following lemma, corresponding to the normal signing procedure and the dual signing simulation of DualMS, respectively. In both procedures, out_1, out_2 correspond to the output of $\textsc{Sign}_1$, $\textsc{Sign}_2$, respectively. Besides bounding the statistical distance between normal and simulated signing, the lemma also bounds the success probability of rejection sampling.

Lemma 8. *Let integers k, l, l', m, and m' satisfy $l + k = m \le \mathsf{poly}(N)$ and $l' + k = m' \le \mathsf{poly}(N)$. Fix some t such that $t = \omega(\sqrt{\log(N)})$ and $t = \omega(\sqrt{\log(N)})$ and $t = o(\log(N))$.*[2] *Let $T = \kappa^2\eta\sqrt{N(l+k)} \ge \max\|c\mathbf{s}\|_2$ and $T' = \kappa^2\eta'\sqrt{N(l'+k)} \ge \max\|c\bar{\mathbf{u}}\|_2$. For any α, let $s > \max(\sqrt{2\pi}\alpha T, 2N \cdot q^{k/m+2/(Nm)})$, $s' > \max(\sqrt{2\pi}\alpha T', 2N \cdot q^{k/m'+2/(Nm')})$, $M = e^{t/\alpha+1/(2\alpha^2)}$, and $\epsilon = 2e^{-t^2/2}$. Let* Trans *and* Sim *be procedures with specific and public inputs as described in Fig. 4. Then with probability $1 - 2^{-\Omega(N)}$ over the choices of $\mathbf{A}$ and $\hat{\mathbf{B}}$ uniformly over $R_q^{k\times l} \times R_q^{k\times(l'-1)}$, for any $\mathbf{s} \in S_\eta^{l+k}$, $\mathbf{u} \in S_{\eta'}^{l'-1+k}$, the following claims hold:*

1. *The distributions of* out_1 *(i.e., $\mathbf{w}$) in* Trans *and* Sim *are identical.*
2. *In both* Trans *and* Sim*, for any $c \in C'$ and conditioned on any* out_1 *(i.e., $\mathbf{w}$), it holds that*
$$\frac{1-\epsilon}{M} - 2^{-\Omega(N)} \le \Pr[\mathsf{out}_2 \ne \perp \mid \mathbf{w}] \le \frac{1}{M} + 2^{-\Omega(N)}.$$
3. *For any $c \in C'$ and conditioned on any* out_1 *(i.e., $\mathbf{w}$), the statistical distance between the distributions of* out_2 *in* Trans *and* Sim *is at most $3\epsilon/(2M) + 2^{-\Omega(N)}$.*

Proof. Claim 1 is obvious, as $\mathbf{y}$, $\mathbf{r}$ in Trans are identical to $\mathbf{z}$, $\mathbf{p}$ in Sim. However, we still need to examine the distribution of $\mathsf{out}_1 = \mathbf{w}$ for proving other claims. We first look at $\mathbf{w}$ in Trans. Split $\mathbf{r}$ into $\mathbf{r} = [r_1, \mathbf{r}_2^\mathsf{T}]^\mathsf{T}$. Then we have

$$\bar{\mathbf{B}}\mathbf{r} = r_1(\mathbf{t} - [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{u}) + [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{r}_2.$$

[2] Since $m \le \mathsf{poly}(N)$ and $m' \le \mathsf{poly}(N)$, we have $\log(Nm) = \Theta(\log N)$ and $\log(Nm') = \Theta(\log N)$, so $t = o(\log(N))$ is enough for invoking Lemma 4.

Specific Inputs:

$\mathbf{s} \in S_\eta^{l+k}$; $\bar{\mathbf{u}} := [1, \mathbf{u}^\intercal]^\intercal$ where $\mathbf{u} \in S_{\eta'}^{l'-1+k}$

Public Inputs:

$c \in C' = \{ab : a, b \in C\}$

$\bar{\mathbf{A}} := [\mathbf{A}|\mathbf{I}]$ where $\mathbf{A} \in R_q^{k\times l}$; $\mathbf{t} := \bar{\mathbf{A}}\mathbf{s}$

$\bar{\mathbf{B}} := [\mathbf{b}|\hat{\mathbf{B}}|\mathbf{I}]$ where $\hat{\mathbf{B}} \in R_q^{k\times(l'-1)}$ and $\mathbf{b} := \mathbf{t} - [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{u}$

$\mathsf{Trans}(\mathbf{s})$	$\mathsf{Sim}(\bar{\mathbf{u}})$
$\mathbf{y} \leftarrow D_s^{l+k}$	$\mathbf{p} \leftarrow D_{s'}^{l'+k}$
$\mathbf{r} \leftarrow D_{s'}^{l'+k}$	$\mathbf{z} \leftarrow D_s^{l+k}$
$\mathsf{out}_1 := \mathbf{w} := \bar{\mathbf{A}}\mathbf{y} + \bar{\mathbf{B}}\mathbf{r}$	$\mathsf{out}_1 := \mathbf{w} := \bar{\mathbf{A}}\mathbf{z} + \bar{\mathbf{B}}\mathbf{p}$
$\mathbf{z} := \mathbf{y} + c\mathbf{s}$	$\mathbf{r} := \mathbf{p} + c\bar{\mathbf{u}}$
With prob. $\min(1, D_s^{l+k}(\mathbf{z})/(M \cdot D_{c\mathbf{s},s}^{l+k}(\mathbf{z})))$:	With prob. $\min(1, D_{s'}^{l'+k}(\mathbf{r})/(M \cdot D_{c\bar{\mathbf{u}},s'}^{l'+k}(\mathbf{r})))$:
$\mathsf{out}_2 := (\mathbf{z}, \mathbf{r})$	$\mathsf{out}_2 := (\mathbf{z}, \mathbf{r})$
Otherwise: // Abort	Otherwise: // Abort
$\mathsf{out}_2 := \perp$	$\mathsf{out}_2 := \perp$

Figure 4: Procedures Trans and Sim of Lemma 8.

By Lemma 5, with probability $1 - 2^{-\Omega(N)}$ over the choice of $\hat{\mathbf{B}}$, $[\hat{\mathbf{B}}|\mathbf{I}]\mathbf{r}_2$ is within statistical distance $2^{-\Omega(N)}$ of the uniform distribution. Hence, $\bar{\mathbf{B}}\mathbf{r}$ and $\mathbf{w}$ are also within distance $2^{-\Omega(N)}$ of the uniform distribution. Similarly, in Sim, $\bar{\mathbf{A}}\mathbf{z}$ and $\mathbf{w}$ are within statistical distance $2^{-\Omega(N)}$ of the uniform distribution.

Let us turn to claim 2 and look at Trans first. The conditional distribution of $\mathbf{y}$ on any $\mathbf{w}$ is within statistical distance $2^{-\Omega(N)}$ of D_s^m, since[3]

$$\begin{aligned}
\Pr[\mathbf{y} = \mathbf{y}^* \mid \mathbf{w} = \mathbf{w}^*] &= \frac{\Pr[\mathbf{w} = \mathbf{w}^* \mid \mathbf{y} = \mathbf{y}^*] \cdot \Pr[\mathbf{y} = \mathbf{y}^*]}{\Pr[\mathbf{w} = \mathbf{w}^*]} \\
&= \frac{\Pr\left[\bar{\mathbf{B}}\mathbf{r} = \mathbf{w}^* - \bar{\mathbf{A}}\mathbf{y}^*\right] \cdot \Pr[\mathbf{y} = \mathbf{y}^*]}{\Pr[\mathbf{w} = \mathbf{w}^*]} \\
&= \frac{(1 \pm 2^{-\Omega(N)})q^{-Nk} \cdot D_s^m(\mathbf{y}^*)}{(1 \pm 2^{-\Omega(N)})q^{-Nk}} \\
&= (1 \pm 2^{-\Omega(N)})D_s^m(\mathbf{y}^*).
\end{aligned}$$

Consider an arbitrary $c \in C'$. Let $\mathbf{v} = c\mathbf{s}$, and $S_{\mathbf{v}} = \{\mathbf{z} \in R^m : M \cdot D_{\mathbf{v},s}^m(\mathbf{z}) \geq D_s^m(\mathbf{z})\}$. Since $\mathbf{z} = \mathbf{y} + c\mathbf{s}$, we have

$$\Pr[\mathbf{z} = \mathbf{z}^* \mid \mathbf{w}] = (1 \pm 2^{-\Omega(N)})D_{\mathbf{v},s}^m(\mathbf{z}^*). \tag{1}$$

[3]In this proof, notation $\mathbf{y}^*$ (and $\mathbf{z}^*$, $\mathbf{r}^*$, $\mathbf{w}^*$, etc.) appear in an equation to denote some specific value if we view $\mathbf{y}$ as a random variable distributed according to the Trans or Sim.

Thus,

$$\begin{aligned}\Pr[\mathsf{out}_2 \neq \perp \mid \mathbf{w}] &\geq \sum_{\mathbf{z}\in R^m} (1-2^{-\Omega(N)}) D^m_{\mathbf{v},s}(\mathbf{z}) \cdot \min(1, \frac{D^m_s(\mathbf{z})}{M \cdot D^m_{\mathbf{v},s}(\mathbf{z})}) \\ &\geq \sum_{\mathbf{z}\in S_{\mathbf{v}}} \frac{D^m_s(\mathbf{z})}{M} + \sum_{\mathbf{z}\notin S_{\mathbf{v}}} D^m_{\mathbf{v},s}(\mathbf{z}) - 2^{-\Omega(N)} \\ &\geq \sum_{\mathbf{z}\in S_{\mathbf{v}}} \frac{D^m_s(\mathbf{z})}{M} - 2^{-\Omega(N)} \geq \frac{1-\epsilon}{M} - 2^{-\Omega(N)}.\end{aligned}$$

In the last inequality, we have used Lemma 4. It also holds that

$$\begin{aligned}\Pr[\mathsf{out}_2 \neq \perp \mid \mathbf{w}] &\leq \sum_{\mathbf{z}\in R^m} (1+2^{-\Omega(N)}) D^m_{\mathbf{v},s}(\mathbf{z}) \cdot \min(1, \frac{D^m_s(\mathbf{z})}{M \cdot D^m_{\mathbf{v},s}(\mathbf{z})}) \\ &\leq \sum_{\mathbf{z}\in R^m} D^m_{\mathbf{v},s}(\mathbf{z}) \cdot \frac{D^m_s(\mathbf{z})}{M \cdot D^m_{\mathbf{v},s}(\mathbf{z})} + 2^{-\Omega(N)} = \frac{1}{M} + 2^{-\Omega(N)}.\end{aligned}$$

Similar arguments can show that in Sim, the conditional distribution of $\mathbf{p}$ on any $\mathbf{w}$ is within statistical distance $2^{-\Omega(N)}$ of $D^{m'}_{s'}$ and give the same bound for $\Pr[\mathsf{out}_2 \neq \perp \mid \mathbf{w}]$.

It remains to prove claim 3. We already have $\Pr[\mathsf{out}_2 = \perp \mid \mathbf{w}]$ in Trans and Sim. It suffices to only consider $\Pr[\mathsf{out}_2 = (\mathbf{z}, \mathbf{r}) \mid \mathbf{w}]$ with $\mathbf{z} \in R^m$ and $\mathbf{r} \in R^{m'}$. In both procedures, $\Pr[\mathsf{out}_2 = (\mathbf{z}, \mathbf{r}) \mid \mathbf{w}] = 0$ when $\bar{\mathbf{A}}\mathbf{z} + \bar{\mathbf{B}}\mathbf{r} \neq \mathbf{w} + c\mathbf{t}$. Define function P over $R^m \times R^{m'}$ as

$$P(\mathbf{z}, \mathbf{r}) = \frac{D^m_s(\mathbf{z}) D^{m'}_{s'}(\mathbf{r})}{M q^{-Nk}}.$$

We are going to show that both in Trans and Sim,

$$\sum_{\substack{\mathbf{z}\in R^m, \mathbf{r}\in R^{m'} \\ \bar{\mathbf{A}}\mathbf{z}+\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}}} |\Pr[\mathsf{out}_2 = (\mathbf{z}, \mathbf{r}) \mid \mathbf{w}] - P(\mathbf{z}, \mathbf{r})| \leq \frac{\epsilon}{M} + 2^{-\Omega(N)}. \tag{2}$$

This will prove claim 3 when combined with claim 2. Again, we prove the bound for Trans, and a similar argument applies to Sim.

Define $P_{\mathbf{w}}$ as

$$P_{\mathbf{w}}(\mathbf{z}^*, \mathbf{r}^*) = \frac{D^m_s(\mathbf{z}^*) \cdot D^{m'}_{s'}(\mathbf{r}^*)}{M \cdot \Pr\big[\bar{\mathbf{B}}\mathbf{r} = \mathbf{w} + c\mathbf{t} - \bar{\mathbf{A}}\mathbf{z}^*\big]},$$

and $P'_{\mathbf{w}}$ as

$$P'_{\mathbf{w}}(\mathbf{z}^*, \mathbf{r}^*) = D^m_{\mathbf{v},s}(\mathbf{z}^*) \cdot \min(1, \frac{D^m_s(\mathbf{z}^*)}{M \cdot D^m_{c\mathbf{s},s}(\mathbf{z}^*)}) \cdot \frac{D^{m'}_{s'}(\mathbf{r}^*)}{\Pr\big[\bar{\mathbf{B}}\mathbf{r} = \mathbf{w} + c\mathbf{t} - \bar{\mathbf{A}}\mathbf{z}^*\big]}.$$

By Lemma 5, we have

$$\sum_{\substack{\mathbf{z}\in R^m, \mathbf{r}\in R^{m'} \\ \bar{\mathbf{A}}\mathbf{z}+\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}}} |P_{\mathbf{w}}(\mathbf{z}, \mathbf{r}) - P(\mathbf{z}, \mathbf{r})| \leq 2^{-\Omega(N)}. \tag{3}$$

For $\mathbf{z}^*$ and $\mathbf{r}^*$ satisfying $\bar{\mathbf{A}}\mathbf{z}^* + \bar{\mathbf{B}}\mathbf{r}^* = \mathbf{w} + c\mathbf{t}$, we have

$$\begin{aligned}&\Pr[\mathsf{out}_2 = (\mathbf{z}^*, \mathbf{r}^*) \mid \mathbf{w}] \\ &\quad = \Pr[\mathbf{z} = \mathbf{z}^* \wedge \mathsf{out}_2 \neq \perp \mid \mathbf{w}] \cdot \Pr[\mathbf{r} = \mathbf{r}^* \mid \mathbf{w} \wedge \mathbf{z} = \mathbf{z}^*] \\ &\quad = \Pr[\mathbf{z} = \mathbf{z}^* \wedge \mathsf{out}_2 \neq \perp \mid \mathbf{w}] \cdot \Pr\big[\mathbf{r} = \mathbf{r}^* \,\big|\, \bar{\mathbf{B}}\mathbf{r} = \mathbf{w} + c\mathbf{t} - \bar{\mathbf{A}}\mathbf{z}^*\big] \\ &\quad = \Pr[\mathbf{z} = \mathbf{z}^* \mid \mathbf{w}] \cdot \min(1, \frac{D^m_s(\mathbf{z}^*)}{M \cdot D^m_{\mathbf{v},s}(\mathbf{z}^*)}) \cdot \frac{D^{m'}_{s'}(\mathbf{r}^*)}{\Pr\big[\bar{\mathbf{B}}\mathbf{r} = \mathbf{w} + c\mathbf{t} - \bar{\mathbf{A}}\mathbf{z}^*\big]}.\end{aligned}$$

By Eq. (1),

$$\sum_{\substack{\mathbf{z}\in R^m,\mathbf{r}\in R^{m'}\\ \bar{\mathbf{A}}\mathbf{z}+\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}}} |\Pr[\mathsf{out}_2=(\mathbf{z},\mathbf{r}) \mid \mathbf{w}] - P'_{\mathbf{w}}(\mathbf{z},\mathbf{r})| \le 2^{-\Omega(N)}. \tag{4}$$

Finally,

$$\begin{aligned}
&\sum_{\substack{\mathbf{z}^*\in R^m,\mathbf{r}^*\in R^{m'}\\ \bar{\mathbf{A}}\mathbf{z}^*+\bar{\mathbf{B}}\mathbf{r}^*=\mathbf{w}+c\mathbf{t}}} |P'_{\mathbf{w}}(\mathbf{z}^*,\mathbf{r}^*) - P_{\mathbf{w}}(\mathbf{z}^*,\mathbf{r}^*)| \\
&= \sum_{\substack{\mathbf{z}^*\in R^m,\mathbf{r}^*\in R^{m'}\\ \bar{\mathbf{A}}\mathbf{z}^*+\bar{\mathbf{B}}\mathbf{r}^*=\mathbf{w}+c\mathbf{t}}} \frac{D_{s'}^{m'}(\mathbf{r}^*)}{\Pr\big[\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}-\bar{\mathbf{A}}\mathbf{z}^*\big]} \left| D^m_{\mathbf{v},s}(\mathbf{z}^*)\cdot\min(1,\frac{D_s^m(\mathbf{z}^*)}{M\cdot D^m_{\mathbf{v},s}(\mathbf{z}^*)}) - \frac{D_s^m(\mathbf{z}^*)}{M}\right| \\
&= \sum_{\substack{\mathbf{z}^*\in S_{\mathbf{v}},\mathbf{r}^*\in R^{m'}\\ \bar{\mathbf{A}}\mathbf{z}^*+\bar{\mathbf{B}}\mathbf{r}^*=\mathbf{w}+c\mathbf{t}}} \frac{D_{s'}^{m'}(\mathbf{r}^*)}{\Pr\big[\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}-\bar{\mathbf{A}}\mathbf{z}^*\big]} \left|\frac{D_s^m(\mathbf{z}^*)}{M} - \frac{D_s^m(\mathbf{z}^*)}{M}\right| \\
&\quad + \sum_{\substack{\mathbf{z}^*\notin S_{\mathbf{v}},\mathbf{r}^*\in R^{m'}\\ \bar{\mathbf{A}}\mathbf{z}^*+\bar{\mathbf{B}}\mathbf{r}^*=\mathbf{w}+c\mathbf{t}}} \frac{D_{s'}^{m'}(\mathbf{r}^*)}{\Pr\big[\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}-\bar{\mathbf{A}}\mathbf{z}^*\big]} \left| D^m_{\mathbf{v},s}(\mathbf{z}^*) - \frac{D_s^m(\mathbf{z}^*)}{M}\right| \\
&\le \sum_{\substack{\mathbf{z}^*\notin S_{\mathbf{v}},\mathbf{r}^*\in R^{m'}\\ \bar{\mathbf{A}}\mathbf{z}^*+\bar{\mathbf{B}}\mathbf{r}^*=\mathbf{w}+c\mathbf{t}}} \frac{D_s^m(\mathbf{z}^*)D_{s'}^{m'}(\mathbf{r}^*)}{M\cdot\Pr\big[\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}-\bar{\mathbf{A}}\mathbf{z}^*\big]} \\
&= \frac{1}{M}\sum_{\mathbf{z}^*\notin S_{\mathbf{v}}} \frac{D_s^m(\mathbf{z}^*)}{\Pr\big[\bar{\mathbf{B}}\mathbf{r}=\mathbf{w}+c\mathbf{t}-\bar{\mathbf{A}}\mathbf{z}^*\big]} \sum_{\substack{\mathbf{r}^*\in R^{m'}\\ \bar{\mathbf{B}}\mathbf{r}^*=\mathbf{w}+c\mathbf{t}-\bar{\mathbf{A}}\mathbf{z}^*}} D_{s'}^{m'}(\mathbf{r}^*) \\
&= \frac{1}{M}\sum_{\mathbf{z}^*\notin S_{\mathbf{v}}} D_s^m(\mathbf{z}^*) \le \frac{\epsilon}{M}.
\end{aligned}$$

Combined with Eqs. (3) and (4), this proves Eq. (2) . A similar argument can show Eq. (2) for Sim. □

4.2 Reduction from sel-UF-CMA to sel-UF-KOA and MLWE

Lemma 9 (sel-UF-CMA to sel-UF-KOA and MLWE)**.** *For any τ-time adversary $\mathcal{A}$ against the sel-UF-CMA of* DualMS *that makes at most Q_h queries to each random oracle and launches at most Q_s sessions with the signing oracles, there exist algorithms $\mathcal{B}$ and $\mathcal{D}$ such that*

$$\begin{aligned}
\mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}CMA}}_{\mathsf{DualMS}}(\mathcal{A}) &\le \mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}KOA}}_{\mathsf{DualMS}}(\mathcal{B}) + (Q_h-1)\mathbf{Adv}_{\mathsf{MLWE}_{q,k,l'-1,\eta'}}(\mathcal{D}) \\
&+ Q_s(\frac{3\epsilon}{2M} + 2^{-\Omega(N)}) + \frac{Q_h(Q_h-1)}{|C|} + (\frac{2}{q^{N/2}})^k,
\end{aligned}$$

where $\epsilon = 2e^{-t^2/2}$, t is a parameter as specified in Table 1, and the running time of $\mathcal{B}$ and $\mathcal{D}$ are essentially τ.

Proof. Let G_0 be the original $\mathsf{sel\text{-}UF\text{-}CMA}_{\mathsf{DualMS}}$ game. We define a series of hybrid games G_1, $\mathsf{G}_{2,0}$, ..., G_{2,Q_h}, and G_3. They are all described in Fig. 5, where we omit normal random oracles $\mathsf{H}_{\mathrm{agg}}$ and $\mathsf{H}_{\mathrm{sig}}$.

$\mathsf{G}_0(\mathcal{A})$ – $\mathsf{G}_3(\mathcal{A})$

$\mathbf{A} \leftarrow\$ R_q^{k\times l}$
$\bar{\mathbf{A}} := [\mathbf{A}|\mathbf{I}]$
$\mathbf{s}_1 \leftarrow\$ S_\eta^{l+k}$
$\mathbf{t}_1^* \leftarrow\$ \bar{\mathbf{A}}\mathbf{s}_1$
$\mathcal{T}_\mathrm{u} := \emptyset$ // $\mathsf{G}_{2,0}$ – G_3
$\mathsf{ctr} := 0$; $\mathcal{S} := \emptyset$; $\mathcal{Q} := \emptyset$
$\textsc{Orc} := \{\textsc{Sign}_1, \textsc{Sign}_2,$
  $\mathsf{H}_\mathrm{agg}, \mathsf{H}_\mathrm{com}, \mathsf{H}_\mathrm{sig}\}$
$(\mathsf{st}, i^*) \leftarrow \mathcal{A}_1(\bar{\mathbf{A}}, \mathbf{t}_1)$ // selective
$(L^*, \tilde{\sigma}^*) \leftarrow \mathcal{A}_2^{\textsc{Orc}}(\mathsf{st})$
if $\mathbf{t}_1^* \notin L^* \vee (L^*, \mu^*) \in \mathcal{Q}$
  $\vee \mathsf{KAgg}(L^*) \neq \tilde{\mathbf{t}}^*$ **then**
  return 0
return $\mathsf{Vf}(\tilde{\mathbf{t}}^*, \mu^*, \tilde{\sigma}^*)$

$\mathsf{H}_\mathrm{com}(\tilde{\mathbf{t}}, \mu)$

if $\mathcal{T}_\mathrm{com}[\tilde{\mathbf{t}}, \mu] \neq \perp$ **then**
  return $\mathcal{T}_\mathrm{com}[\tilde{\mathbf{t}}, \mu]$
$\mathbf{B} \leftarrow\$ R_q^{k\times l'}$ // G_0
if $|\mathcal{T}_\mathrm{com}| = i^* - 1$ **then** // G_1 – G_3
  $\mathbf{B} \leftarrow\$ R_q^{k\times l'}$ // G_1 – G_3
else // G_1 – G_3
  $\hat{\mathbf{B}} \leftarrow\$ R_q^{k\times(l'-1)}$ // G_1 – G_3
  $\mathbf{v} \leftarrow\$ R_q^k$ // G_1
  if $|\mathcal{T}_\mathrm{com}| < j$ **then** // $\mathsf{G}_{2,j}$ – G_3
    $\mathbf{u} \leftarrow\$ S_{\eta'}^{l'-1+k}$ // $\mathsf{G}_{2,j}$ – G_3
    $\mathcal{T}_\mathrm{u}[\tilde{\mathbf{t}}, \mu] := \mathbf{u}$ // $\mathsf{G}_{2,j}$ – G_3
    $\mathbf{v} := [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{u}$ // $\mathsf{G}_{2,j}$ – G_3
  else // $\mathsf{G}_{2,j}$ – G_3
    $\mathbf{v} \leftarrow\$ R_q^k$ // $\mathsf{G}_{2,j}$ – G_3
  $\mathbf{b} := \mathbf{t}_1^* - \mathbf{v}$ // G_1 – G_3
  $\mathbf{B} := [\mathbf{b}|\hat{\mathbf{B}}]$ // G_1 – G_3
$\mathcal{T}_\mathrm{com}[\tilde{\mathbf{t}}, \mu] := \mathbf{B}$
if $|\mathcal{T}_\mathrm{com}| = i^*$ **then**
  $(\tilde{\mathbf{t}}^*, \mu^*) := (\tilde{\mathbf{t}}, \mu)$ // selective
return $\mathcal{T}_\mathrm{com}[\tilde{\mathbf{t}}, \mu]$

$\textsc{Sign}_1(\{\mathbf{t}_2, \ldots, \mathbf{t}_n\}, \mu)$ // G_0

$\mathsf{ctr} := \mathsf{ctr} + 1$
$\mathsf{sid} := \mathsf{ctr}$; $\mathcal{S} := \mathcal{S} \cup \{\mathsf{sid}\}$
$L := \{\mathbf{t}_1^*, \ldots, \mathbf{t}_n\}$
$\mathcal{Q} := \mathcal{Q} \cup \{(L, \mu)\}$
$\tilde{\mathbf{t}} := \mathsf{KAgg}(L)$
if $(\tilde{\mathbf{t}}, \mu) = (\tilde{\mathbf{t}}^*, \mu^*)$ **then** // G_3
  return $\perp$ // G_3
$a_1 := \mathsf{H}_\mathrm{agg}(L, \mathbf{t}_1^*)$
$\mathbf{B} := \mathsf{H}_\mathrm{com}(\tilde{\mathbf{t}}, \mu)$
$\bar{\mathbf{B}} := [\mathbf{B}|\mathbf{I}]$
$\mathbf{y}_1 \leftarrow D_s^{l+k}$ // G_0 – $\mathsf{G}_{2,Q_\mathrm{h}}$
$\mathbf{r}_1 \leftarrow D_{s'}^{l'+k}$ // G_0 – $\mathsf{G}_{2,Q_\mathrm{h}}$
$\mathbf{w}_1 := \bar{\mathbf{A}}\mathbf{y}_1 + \bar{\mathbf{B}}\mathbf{r}_1$ // G_0 – $\mathsf{G}_{2,Q_\mathrm{h}}$
$\mathsf{st}_\mathsf{sid} := (\tilde{\mathbf{t}}, \mu, a_1, \mathbf{y}_1, \mathbf{r}_1, \mathbf{w}_1)$ // G_0 – $\mathsf{G}_{2,Q_\mathrm{h}}$
$\mathbf{p} \leftarrow D_{s'}^{l'+k}$ // G_3
$\mathbf{z}_1 \leftarrow D_s^{l+k}$ // G_3
$\mathbf{w}_1 := \bar{\mathbf{A}}\mathbf{z}_1 + \bar{\mathbf{B}}\mathbf{p}$ // G_3
$\mathsf{st}_\mathsf{sid} := (\tilde{\mathbf{t}}, \mu, a_1, \mathbf{p}, \mathbf{z}_1, \mathbf{w}_1)$ // G_3
return $\mathbf{w}_1$

$\textsc{Sign}_2(\mathsf{sid}, \{\mathbf{w}_2, \ldots, \mathbf{w}_n\})$

if $\mathsf{sid} \notin \mathcal{S}$ **then return** $\perp$
$\mathcal{S} := \mathcal{S} \setminus \{\mathsf{sid}\}$
$(\tilde{\mathbf{t}}, \mu, a_1, \mathbf{y}_1, \mathbf{r}_1, \mathbf{w}_1) := \mathsf{st}_\mathsf{sid}$ // G_0 – $\mathsf{G}_{2,Q_\mathrm{h}}$
$(\tilde{\mathbf{t}}, \mu, a_1, \mathbf{p}, \mathbf{z}_1, \mathbf{w}_1) := \mathsf{st}_\mathsf{sid}$ // G_3
$\tilde{\mathbf{w}} := \sum_{i=1}^n \mathbf{w}_i$
$c := \mathsf{H}_\mathrm{sig}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$
$\mathbf{z}_1 := \mathbf{y}_1 + a_1 c \mathbf{s}_1$ // G_0 – $\mathsf{G}_{2,Q_\mathrm{h}}$
$\mathbf{u} := \mathcal{T}_\mathrm{u}[\tilde{\mathbf{t}}, \mu]$ // G_3
$\bar{\mathbf{u}} := [1, \mathbf{u}^\intercal]^\intercal$ // G_3
$\mathbf{r}_1 := \mathbf{p} + a_1 c \bar{\mathbf{u}}$ // G_3
With prob. $\min(1, \frac{D_s^{l+k}(\mathbf{z}_1)}{M \cdot D_{a_1 c \mathbf{s}_1, s}^{l+k}(\mathbf{z}_1)})$: // G_0 – $\mathsf{G}_{2,Q_\mathrm{h}}$
With prob. $\min(1, \frac{D_{s'}^{l'+k}(\mathbf{r}_1)}{M \cdot D_{a_1 c \bar{\mathbf{u}}, s'}^{l'+k}(\mathbf{r}_1)})$: // G_3
  return $(c, \mathbf{z}_1, \mathbf{r}_1)$
Otherwise:
  return $\perp$

Figure 5: The hybrid games G_0 – G_3.

G_1 differs from G_0 only in H_com. In G_0, on any fresh query, H_com returns $\mathbf{B}$ uniformly chosen from $R_q^{k\times l'}$. In G_1, except on the i^*-th fresh query, the first column $\mathbf{b}$ of $\mathbf{B}$ is instead decided by

$\mathbf{b} := \mathbf{t}_1^* - \mathbf{v}$ with $\mathbf{v} \leftarrow\$ R_q^k$. Apparently the view of $\mathcal{A}$ in G_0 and G_1 are identical, so we have

$$\Pr[\mathsf{G}_0(\mathcal{A}) = 1] = \Pr[\mathsf{G}_1(\mathcal{A}) = 1].$$

$\mathsf{G}_{2,0}$ only differs from G_1 in the game initialization. To set up, $\mathsf{G}_{2,0}$ initializes an empty table $\mathcal{T}_{\mathbf{u}}$. For $1 \le j \le Q_{\mathrm{h}}$, $\mathsf{G}_{2,j}$ differs from $\mathsf{G}_{2,j-1}$ in the j-th fresh $\mathsf{H}_{\mathrm{com}}$ query. To answer the j-th fresh $\mathsf{H}_{\mathrm{com}}$ query in $\mathsf{G}_{2,j-1}$, if $j \neq i^*$, then the first column of $\mathbf{B} = [\mathbf{b}|\hat{\mathbf{B}}]$ is decided by $\mathbf{b} := \mathbf{t}_1^* - \mathbf{v}$ with uniform $\mathbf{v}$. In $\mathsf{G}_{2,j}$, $\mathbf{v}$ is instead decided by $\mathbf{v} := [\hat{\mathbf{B}}|\mathbf{I}]\mathbf{u}$ with a short $\mathbf{u}$ uniformly chosen from $S_{\eta'}^{l'-1+k}$. Then $\mathbf{u}$ is stored in $\mathcal{T}_{\mathbf{u}}[\tilde{\mathbf{t}}, \mu]$.

Note that $\hat{\mathbf{B}}$ and $\mathbf{v}$ constitute a $\mathsf{MLWE}_{q,k,l'-1,\eta'}$ instance. A distinguisher $\mathcal{D}$ against $\mathsf{MLWE}_{q,k,l'-1,\eta'}$ can simulate $\mathsf{G}_{2,j-1}$ / $\mathsf{G}_{2,j}$ for $1 \le j \le Q_{\mathrm{h}}$ and $j \neq i^*$ for $\mathcal{A}$, since in $\mathsf{G}_{2,j}$ the short secret $\mathbf{u}$ is never used elsewhere. Thus, we have

$$\Pr[\mathsf{G}_{2,0}(\mathcal{A}) = 1] = \Pr[\mathsf{G}_1(\mathcal{A}) = 1], \quad \Pr[\mathsf{G}_{2,i^*}(\mathcal{A}) = 1] = \Pr[\mathsf{G}_{2,i^*-1}(\mathcal{A}) = 1],$$

and for $1 \le j \le Q_{\mathrm{h}}$ and $j \neq i^*$

$$\Pr[\mathsf{G}_{2,j-1}(\mathcal{A}) = 1] - \Pr[\mathsf{G}_{2,j}(\mathcal{A}) = 1] \le \mathbf{Adv}_{\mathsf{MLWE}_{q,k,l'-1,\eta'}}(\mathcal{D}).$$

G_3 differs from $\mathsf{G}_{2,Q_{\mathrm{h}}}$ only in the signing oracles. In $\mathsf{G}_{2,Q_{\mathrm{h}}}$, the signing oracles use the secret key $\mathbf{s}_1$ to execute the signing protocol like an honest party. In G_3, the signing oracles retrieve $\mathbf{u} = \mathcal{T}_{\mathbf{u}}[\tilde{\mathbf{t}}, \mu]$ to obtain a vector $\bar{\mathbf{u}} = [1, \mathbf{u}^\intercal]^\intercal$. Unless $\mathsf{H}_{\mathrm{com}}(\tilde{\mathbf{t}}, \mu)$ is the i^*-th fresh $\mathsf{H}_{\mathrm{com}}$ query, $\bar{\mathbf{B}}$ and $\bar{\mathbf{u}}$ will have relation as specified in Fig. 4. The oracles use $\bar{\mathbf{u}}$ to perform straight-line simulation following algorithm Sim in Fig. 4. There will be no item $\mathcal{T}_{\mathbf{u}}[\tilde{\mathbf{t}}^*, \mu^*]$ in $\mathcal{T}_{\mathbf{u}}$, so the signing oracles in G_3 fail when dealing with $(\tilde{\mathbf{t}}, \mu) = (\tilde{\mathbf{t}}^*, \mu^*)$.

In each signing session, if $(\tilde{\mathbf{t}}, \mu) \neq (\tilde{\mathbf{t}}^*, \mu^*)$, it can be verified that the outputs of the signing oracles in $\mathsf{G}_{2,Q_{\mathrm{h}}}$ and G_3 are distributed respectively according to Trans and Sim in Fig. 4. We already bounded their statistical distance by $3\epsilon/(2M) + 2^{-\Omega(N)}$. On the other hand, G_3 will be distinguishable from $\mathsf{G}_{2,Q_{\mathrm{h}}}$ if $(\tilde{\mathbf{t}}, \mu) = (\tilde{\mathbf{t}}^*, \mu^*)$. Nevertheless, in this case $\mathsf{G}_{2,Q_{\mathrm{h}}}$ outputs 1 only if $\mathcal{A}$ at the end outputs a different L^* from the queried public-key list L satisfying $\mathsf{KAgg}(L^*) = \mathsf{KAgg}(L) = \tilde{\mathbf{t}}^*$. Let AggCol denote the event that $\mathcal{A}$ ever queried two different public-key lists $L^* \neq L$ satisfying $\mathsf{KAgg}(L^*) = \mathsf{KAgg}(L)$. Lemma 12 in Appendix A upper-bounds the probability of AggCol by $Q_{\mathrm{h}}(Q_{\mathrm{h}} - 1)/|C| + (2/q^{N/2})^k$. Therefore, we have

$$\Pr[\mathsf{G}_{2,Q_{\mathrm{h}}}(\mathcal{A}) = 1] - \Pr[\mathsf{G}_3(\mathcal{A}) = 1] \le Q_{\mathrm{s}}(\frac{3\epsilon}{2M} + 2^{-\Omega(N)}) + \frac{Q_{\mathrm{h}}(Q_{\mathrm{h}} - 1)}{|C|} + (\frac{2}{q^{N/2}})^k.$$

Finally, note that $\mathbf{s}_1$ is only used to compute $\mathbf{t}_1^*$ in G_3. Hence, an adversary $\mathcal{B}$ against the sel-UF-KOA of DualMS can perfectly simulate G_3 for $\mathcal{A}$. It uses the corresponding random oracle responses in its own sel-UF-KOA game to answer $\mathsf{H}_{\mathrm{agg}}$ and $\mathsf{H}_{\mathrm{sig}}$ queries from $\mathcal{A}$ and to answer the i^*-th fresh query to $\mathsf{H}_{\mathrm{com}}$. It outputs the same i^*, L^*, and $\tilde{\sigma}^*$ as $\mathcal{A}$. Consequently, we have

$$\begin{aligned}\mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}CMA}}_{\mathsf{DualMS}}(\mathcal{A}) \le\ & \mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}KOA}}_{\mathsf{DualMS}}(\mathcal{B}) + (Q_{\mathrm{h}} - 1)\mathbf{Adv}_{\mathsf{MLWE}_{q,k,l'-1,\eta'}}(\mathcal{D}) \\ & + Q_{\mathrm{s}}(\frac{3\epsilon}{2M} + 2^{-\Omega(N)}) + \frac{Q_{\mathrm{h}}(Q_{\mathrm{h}} - 1)}{|C|} + (\frac{2}{q^{N/2}})^k.\end{aligned}$$

□

4.3 Reduction from sel-UF-KOA to MSIS and MLWE

Lemma 10 (sel-UF-KOA to MSIS and MLWE)**.** *For any τ-time adversary $\mathcal{A}$ against the sel-UF-KOA of* DualMS *that makes at most Q_h queries to each random oracle, there exist algorithms $\mathcal{B}$ and*

$\mathcal{D}$ *such that*

$$\mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}KOA}}_{\mathsf{DualMS}}(\mathcal{A}) \le \sqrt{\frac{Q_h^2}{|C|} + Q_h\sqrt{Q_h \mathbf{Adv}_{\mathsf{MSIS}_{q,k,1+l+l',\beta}}(\mathcal{B})}}$$
$$+ \mathbf{Adv}_{\mathsf{MLWE}_{q,k,l,\eta}}(\mathcal{D}) + \frac{Q_h(Q_h+1)}{|C|} + 2(\frac{2}{q^{N/2}})^k,$$

where $\beta = 8\kappa\sqrt{\hat{n}^2\kappa^3 + B_n^2 + B_n'^2 + B_n''^2}$, $\hat{n}$ *is the maximum number of duplicate keys in a public key list, and the running time of* $\mathcal{B}$ *and* $\mathcal{D}$ *are essentially* 4τ *and* τ*, respectively.*

The proof follows the "double forking" framework of [MPSW19, NRS21, BTT22b]. In particular, a double-forking proof for lattice-based scheme was given in [BTT22b], and our proof is analogous to theirs. Therefore, we only sketch the proof here and defer the complete version to Appendix A. We first consider the inner forking algorithm $\mathcal{B}'$ that runs $\mathcal{A}$ twice. If adversary $\mathcal{A}$ wins in the first time, $\mathcal{B}'$ can obtain L^*, $\tilde{\mathbf{t}}^*$ μ, $\tilde{\mathbf{w}}$, $\tilde{\mathbf{z}}$, $\tilde{\mathbf{r}}$, and $\tilde{\mathbf{e}}$ satisfying $\tilde{\mathbf{t}}^* = \mathsf{KAgg}(L^*)$ and

$$\mathbf{A}\tilde{\mathbf{z}} + \mathbf{B}\tilde{\mathbf{r}} + \tilde{\mathbf{e}} = \tilde{\mathbf{w}} + c\tilde{\mathbf{t}}^*,$$

where $c = \mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ is what we call the "crucial" query corresponding to the forgery. Algorithm $\mathcal{B}'$ then forks the adversary at the crucial query, assigning another hash value c' to $\mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ in the second execution. In this execution, with probability lower-bounded by the forking lemma [PS96, BN06], we have $c \neq c'$, $\mathcal{A}$ wins, and $\mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$ is again the crucial query. In this case, $\mathcal{B}'$ obtains another tuple of responses $\tilde{\mathbf{z}}'$, $\tilde{\mathbf{r}}'$, and $\tilde{\mathbf{e}}'$ satisfying

$$\mathbf{A}\tilde{\mathbf{z}}' + \mathbf{B}\tilde{\mathbf{r}}' + \tilde{\mathbf{e}}' = \tilde{\mathbf{w}} + c'\tilde{\mathbf{t}}^*.$$

Combine the two equations, and then we have

$$\mathbf{A}\hat{\mathbf{z}} + \mathbf{B}\hat{\mathbf{r}} + \hat{\mathbf{e}} = \hat{c}\tilde{\mathbf{t}}^*,$$

where $\hat{\mathbf{z}} = \tilde{\mathbf{z}} - \tilde{\mathbf{z}}'$, $\hat{\mathbf{r}} = \tilde{\mathbf{r}} - \tilde{\mathbf{r}}'$, $\hat{\mathbf{e}} = \tilde{\mathbf{e}} - \tilde{\mathbf{e}}'$ and $\hat{c} = c - c' \neq 0$.

Now we consider the outer forking algorithm $\mathcal{B}$ that runs $\mathcal{B}'$ twice. Note that $\tilde{\mathbf{t}}^* = \mathsf{KAgg}(L^*) = n^* a_1 \mathbf{t}_1^* + \sum_{\mathbf{t}_i \in L^* \wedge \mathbf{t}_i \neq \mathbf{t}_1^*} a_i \mathbf{t}_i$, where n^* is the number of times that $\mathbf{t}_1^*$ occurs in L^*. Thus, the earlier equation becomes

$$\mathbf{A}\hat{\mathbf{z}} + \mathbf{B}\hat{\mathbf{r}} + \hat{\mathbf{e}} = \hat{c}(n^* a_1 \mathbf{t}_1^* + \sum_{\mathbf{t}_i \in L^* \wedge \mathbf{t}_i \neq \mathbf{t}_1^*} a_i \mathbf{t}_i).$$

This time we regard $a_1 = \mathsf{H}_{\mathrm{agg}}(L^*, \mathbf{t}_1^*)$ as the crucial query. Algorithm $\mathcal{B}$ runs $\mathcal{B}'$ another time and assigns another value a_1' to $\mathsf{H}_{\mathrm{agg}}(L^*, \mathbf{t}_1^*)$. With the probability given by the forking lemma, we have $a_1 \neq a_1'$, again $\mathcal{B}'$ succeeds, and $\mathsf{H}_{\mathrm{agg}}(L^*, \mathbf{t}_1^*)$ is the crucial query. Then $\mathcal{B}$ obtains $\hat{c}'$, $\hat{\mathbf{z}}'$, $\hat{\mathbf{r}}'$, and $\hat{\mathbf{e}}'$ satisfying

$$\mathbf{A}\hat{\mathbf{z}}' + \mathbf{B}\hat{\mathbf{r}}' + \hat{\mathbf{e}}' = \hat{c}'(n^* a_1' \mathbf{t}_1^* + \sum_{\mathbf{t}_i \in L^* \wedge \mathbf{t}_i \neq \mathbf{t}_1^*} a_i \mathbf{t}_i).$$

Multiply the first equation by $\hat{c}'$ and the second by $\hat{c}$ and subtract the second from the first, and then we have

$$\mathbf{A}(\hat{c}'\hat{\mathbf{z}} - \hat{c}\hat{\mathbf{z}}') + \mathbf{B}(\hat{c}'\hat{\mathbf{r}} - \hat{c}\hat{\mathbf{r}}') + \hat{c}'\hat{\mathbf{e}} - \hat{c}\hat{\mathbf{e}}' = n^*\hat{c}\hat{c}'(a_1 - a_1')\mathbf{t}_1^*.$$

Rearrange the equation, and we have

$$[\mathbf{t}_1^* | \mathbf{A} | \mathbf{B} | \mathbf{I}] \begin{bmatrix} n^*\hat{c}\hat{c}'(a_1 - a_1') \\ \hat{c}'\hat{\mathbf{z}} - \hat{c}\hat{\mathbf{z}}' \\ \hat{c}'\hat{\mathbf{r}} - \hat{c}\hat{\mathbf{r}}' \\ \hat{c}'\hat{\mathbf{e}} - \hat{c}\hat{\mathbf{e}}' \end{bmatrix} = 0.$$

By Lemma 1, none of $\hat{c}$, $\hat{c}'$, and $a_1 - a'_1$ are zero-divisors. Hence, $n^*\hat{c}\hat{c}'(a_1 - a'_1) \neq 0$, and $\mathcal{B}$ obtains a MSIS solution with respect to the matrix $[\mathbf{t}_1^*|\mathbf{A}|\mathbf{B}]$. Here we replace $\mathbf{t}_1^* = \bar{\mathbf{A}}\mathbf{s}_1$ with a random $\mathbf{t}_1^*$ independent of $\mathbf{A}$ by MLWE.

We need to be careful with the possibility that $\mathcal{B}'$ obtains different public-key sets from the two executions of $\mathcal{A}$. The consequence is that there will not be a unique crucial query $\mathsf{H}_{\mathrm{agg}}(L^*, \mathbf{t}_1^*)$ for $\mathcal{B}$ to fork. Fortunately, such an event is unlikely, because KAgg is somehow "collision resistant". We will formally define the relative bad events and bound their probabilities in Appendix A.

5 Concrete Parameters and Comparison

In this section, we provide two sample concrete parameter settings for DualMS, aiming at about 128 classical bits of security. We consider the number of signers $n = 32$ and $n = 1024$. We also provide parameters for $\mathsf{MuSig\text{-}L}$ for comparison.

Choosing parameters. For both schemes, we fix $N = 256$. Then we let $\kappa = 60$ for 256 bits of entropy. We set $t = 13.5$ aiming at 128-bit security and $\gamma = 1.1$ to let the bound given by Lemma 3 small enough.

For DualMS, we fix $l' = l + 1$, $\eta' = \eta$ and set $\alpha = 8.5n$ for an expected repetition number ≈ 5. It remains for us to choose main parameters q, k, l, and η. We consider the following security criteria:

- The hardness of forging a multi-signature on a new message. Based on the security of hash functions, this is conjectured to be as hard as choosing the inputs to hash functions to fix a target $\mathbf{t}' = \tilde{\mathbf{w}} + c\tilde{\mathbf{t}}$ and finding $\mathbf{z}$, $\mathbf{r}$, and $\mathbf{e}$ satisfying $\mathbf{Az} + \mathbf{Br} + \mathbf{e} = \mathbf{t}'$ [DKL+21]. This implies solving $\mathsf{MSIS}_{q,k,l+l'+1,\beta} = \mathsf{MSIS}_{q,k,2l+2,\beta}$, where $\beta \approx \sqrt{B_n^2 + B_2'^2 + B_2''^2}$.
- The pseudorandomness of the public key, which is related to the hardness of $\mathsf{MLWE}_{q,k,l,\eta}$.
- The "zero-knowledge" of signatures, i.e., how well the signatures hide useful information from the adversary, which is related to $\mathsf{MLWE}_{q,k,l'-1,\eta'} = \mathsf{MLWE}_{q,k,l,\eta}$ required by the signature simulation.

For $\mathsf{MuSig\text{-}L}$, we consider an optimized scheme with computational instead of statistical trapdoor indistinguishability. Similarly, we estimate the hardness of forging a multi-signature related to MSIS, the pseudorandomness of the public key related to MLWE, and the trapdoor indistinguishability required by signature simulation. Remarkably, the simulation of $\mathsf{MuSig\text{-}L}$ requires the Gaussian width s to be so large that the protocol almost always succeeds. See more details about the parameters of $\mathsf{MuSig\text{-}L}$ in Appendix B.

We estimate the hardness of MSIS and MLWE using the security estimator of CRYSTALS.[4] Size of the challenge in the signature is estimated as 32 bytes as in [DKL+21]. Sizes of Gaussian responses in the signature are simply estimated based on B_n, B'_n, and B''_n and their dimensions based on Lemma 2 of [BB13]. The main parameters are chosen to minimize public-key size + signature size while satisfying the required security level.

Comparison. Table 2 summarizes our sample parameters. It shows that, aiming at approximately 128-bit security, public-key size + signature size of DualMS is about 4.7x times smaller than $\mathsf{MuSig\text{-}L}$ when $n = 32$ and about 3.4x smaller when $n = 1024$.

The comparison in Gaussian width s explains such differences. Relatively large Gaussian width of $\mathsf{MuSig\text{-}L}$ is somehow inherent in its construction and simulation technique. The large Gaussian width is directly related to the signature size. Moreover, it makes the underlying MSIS problem easier and thus requires to raise the main parameters.

[4] https://github.com/pq-crystals/security-estimates

Params	MuSig-L	DualMS	MuSig-L	DualMS
n	32	32	1024	1024
N	256	256	256	256
$\lceil \log(q) \rceil$	104	37	104	48
k	8	6	10	7
l	42	7	38	9
η	1	1	1	1
PK	26	6.94	32.5	10.5
Sig	98.47	19.72	106.53	30.91
PK+Sig	124.47	26.66	139.03	41.41
$\log(s)$	60.04	27.13	65.54	32.27
MSIS	125	130	126	132
MLWE	> 700	130	> 500	131

Table 2: Summary of the concrete parameters. "PK" refers to public-key size, and "Sig" refers to signature size, both measured in kB. "MSIS" refers to the hardness of MSIS related to signature forgery. Variable s refers to Gaussian width of the response in the signature. For DualMS, it actually refers to s' which is slightly larger than s. "MLWE" refers to the hardness of MLWE related to the pseudorandomness of public keys (and the indistinguishability of simulation for DualMS). We do not show the hardness of MLWE related to the indistinguishability of simulation of MuSig-L, since it can be set to sufficiently hard with a very small effect on the efficiency (see Appendix B).

Let us give a more detailed discussion. In DualMS, like typical FSwA signature schemes, the Gaussian width s of responses is decided by the requirement of hiding secret keys using rejection sampling. However, the construction of MuSig-L introduces another dominant requirement on the width that grows with $q^{k/(l+k)}$. The consequences include unusually large s and almost no rejection, large l and unnecessarily hard MLWE. Our understanding is that, while MuSig-L enables a clever straight-line simulation, it goes toward the opposite direction of Lyubashevsky's FSwA signatures [Lyu09, Lyu12], which introduce rejection sampling and LWE for a smaller Gaussian width and narrower matrix. On the other hand, the extra lower-bound of s in MuSig-L is independent of $\alpha \propto n$, so the growth of n affects MuSig-L less than DualMS when n is not too large.

We point out that we do not apply practical optimizations to DualMS and MuSig-L. We believe there is a lot of room for improvement of the concrete efficiency of both schemes. Our sample parameters provide more a proof of concept than practical meanings. Especially, parameter setting for MuSig-L is more involved. The authors of [BTT22b] did not provide concrete parameters, and our parameters could be non-optimal. We leave further optimizations and more comparison with different settings, such as security levels and number of signers, as future work.

Despite the advantages in efficiency of DualMS, we stress again that MuSig-L is irreplaceable so far if one wants the first round to be offline. Our estimation shows that the online-offline property could be expensive in lattice setting.

Communication and round complexity. Note that despite being two-round, the signing protocol succeeds in returning a valid signature only with some probability. Hence, the actual round complexity to produce a signature is larger. The naive way to produce a valid signature is to sequentially repeat the signing protocol until getting one. In this case, the protocol round complexity (2 for our scheme) should be multiplied by the expected times of repetitions. In fact, we can also repeat the protocol in parallel, which increases the communication while reduce the actual round number. That is, outside the signing protocol, there is always a trade-off between communication and round complexity.

We provide some examples for our parameter setting. We set the success probability to be around

1/5, so sequential repetitions make the expected round number to be about $5 \times 2 = 10$. If we repeat the protocol for 5 times in parallel, then the success probability will be amplified to more than 2/3, which yields an expected round number of about $1.5 \times 2 = 3$. If we want the success probability to be about 0.99 so that the actual round number is virtually 2, then it will suffice to run 20 parallel repetitions.

Let us roughly compare the communication cost of DualMS and MuSig-L. In MuSig-L, each signer broadcasts in the first round more than $\log q$ many commitments, each of equal size to a public key, and in the second round roughly a signature. In DualMS, each signer broadcasts only one commitment in the first round. The second-round message is k dimensions longer than a signature but is only output with probability about 1/5. Suppose DualMS is executed in parallel 20 times, and MuSig-L is run only once since it almost never rejects. Then in our parameter settings with $n = 32$ for DualMS and MuSig-L, respectively, each signer broadcasts about $<$300 kB vs. $>$2800 kB to produce one valid signature. When $n = 1024$, the number is $<$400 kB vs. $>$3500 kB.

Here we also mention a possible trick to reduce the second round communication of DualMS: generate a pseudorandom $\mathbf{r}_1$ and only broadcast the seed in the second round.

Comparison with DOTT. DualMS has a closer structure to DOTT than MuSig-L. They are likely to have similar public size and communication, while the main difference is the signature size. Compared to DOTT, DualMS basically replaces the opening of a commitment scheme with a partial response of Dilithium (the vector $\mathbf{r}$) in the signature.

We do not make concrete comparison with DOTT. As it is a generic construction using homomorphic equivocable trapdoor commitments as a building block, the concrete efficiency will vary according to different instantiations of the commitment scheme. On the other hand, we argued in Section 1 that the instantiations from MSIS and MLWE of such commitment schemes so far give a significantly larger opening size compared to a Dilithium-G signature.

Here, we provide some evidence of the advantage of DualMS over DOTT from their constructions: DualMS can be seen as an instantiation of DOTT, with a much weaker "commitment". In other words, DOTT overstrikes the target of a multi-signature scheme using an unnecessarily powerful building block.

Specifically, if we view $\mathbf{w} = \bar{\mathbf{A}}\mathbf{y} + \bar{\mathbf{B}}\mathbf{r}$ as a commitment to $\bar{\mathbf{A}}\mathbf{y}$ with commitment key $\bar{\mathbf{B}}$ and randomness $\mathbf{r}$, then DualMS is an instantiation of the generic construction of DOTT. However, the properties of such a "commitment" scheme have two main differences compared to the requirements of the security analysis of DOTT.

- The commitment is not binding, while the security of DOTT is reduced to the binding of the commitment scheme. Clearly, one can open $\mathbf{w}$ to some random vector by sampling random $\mathbf{r}$.
- The equivocability is much weaker. An equivocable commitment scheme generally allows the trapdoor-holder to open a commitment to any vector. In our commitment scheme, the trapdoor is a short vector $\bar{\mathbf{u}}$ satisfying $\bar{\mathbf{B}}\bar{\mathbf{u}} = \mathbf{t}$. It only allows the trapdoor-holder to commit to a vector $\bar{\mathbf{A}}\mathbf{y}$ and then equivocate it to some related vector, namely $\bar{\mathbf{A}}\mathbf{y} - c\mathbf{t}$, where c is from a much smaller set than the whole vector space. Moreover, rejection sampling is necessary to keep the trapdoor secret, so the equivocation can fail. It was also shown in Schnorr-based constructions that weaker equivocability is enough for multi-signatures [BCJ08, PW23].

The security proof of DOTT cannot work with our weaker equivocable commitment. Thus, it is necessary for us to provide a new proof for DualMS.

Acknowledgements

We thank Andrej Bogdanov for his helpful advice throughout this work. We thank Yunlei Zhao for letting us know this topic and the trick that reduces communication and Zhichuang Liang for his help in parameter setting. We thank the anonymous reviewers of PKC 2023 and Crypto 2023 for their constructive comments and suggestions.

References

[AB21] Handan Kilinç Alper and Jeffrey Burdges. Two-round trip schnorr multi-signatures via delinearized witnesses. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 157–188, Virtual Event, August 2021. Springer, Heidelberg. 1, 4, 5

[BB13] Rachid El Bansarkhani and Johannes Buchmann. Improvement and efficient implementation of a lattice-based signature scheme. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*, pages 48–67. Springer, 2013. 21

[BCJ08] Ali Bagherzandi, Jung Hee Cheon, and Stanislaw Jarecki. Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008*, pages 449–458. ACM Press, October 2008. 1, 3, 5, 23

[BD21] Mihir Bellare and Wei Dai. Chain reductions for multi-signatures and the HBMS scheme. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 650–678. Springer, Heidelberg, December 2021. 1, 5, 13

[BK20] Dan Boneh and Sam Kim. One-time and interactive aggregate signatures from lattices. https://crypto.stanford.edu/~skim13/agg_ots.pdf, 2020. 2

[BLL+21] Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 33–53. Springer, Heidelberg, October 2021. 1, 13

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 390–399. ACM Press, October / November 2006. 1, 20, 30

[BTT22a] Cecilia Boschini, Akira Takahashi, and Mehdi Tibouchi. MuSig-L: Lattice-based multi-signature with single-round online phase. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 276–305. Springer, Heidelberg, August 2022. 2

[BTT22b] Cecilia Boschini, Akira Takahashi, and Mehdi Tibouchi. MuSig-L: Lattice-based multi-signature with single-round online phase. Cryptology ePrint Archive, Report 2022/1036, 2022. https://eprint.iacr.org/2022/1036. 2, 4, 13, 20, 22, 30, 31

[DEF+19] Manu Drijvers, Kasra Edalatnejad, Bryan Ford, Eike Kiltz, Julian Loss, Gregory Neven, and Igors Stepanovs. On the security of two-round multi-signatures. In *2019 IEEE Symposium on Security and Privacy*, pages 1084–1101. IEEE Computer Society Press, May 2019. 1, 3, 5, 13

[DKL+21] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium c algorithm specifications and supporting documentation (version 3.1), 2021. 21

[DLL+17] Léo Ducas, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehle. CRYSTALS – Dilithium: Digital signatures from module lattices. Cryptology ePrint Archive, Report 2017/633, 2017. https://eprint.iacr.org/2017/633. 2

[DOTT21] Ivan Damgård, Claudio Orlandi, Akira Takahashi, and Mehdi Tibouchi. Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 99–130. Springer, Heidelberg, May 2021. 2

[DOTT22] Ivan Damgård, Claudio Orlandi, Akira Takahashi, and Mehdi Tibouchi. Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. *J. Cryptol.*, 35(2):14, 2022. 2, 4, 6, 13

[ES16] Rachid El Bansarkhani and Jan Sturm. An efficient lattice-based multisignature scheme with applications to bitcoins. In Sara Foresti and Giuseppe Persiano, editors, *CANS 16*, volume 10052 of *LNCS*, pages 140–155. Springer, Heidelberg, November 2016. 2

[FH19] Masayuki Fukumitsu and Shingo Hasegawa. A tightly-secure lattice-based multisignature. In Keita Emura and Takaaki Mizuki, editors, *Proceedings of the 6th on ASIA Public-Key Cryptography Workshop, APKC@AsiaCCS 2019, Auckland, New Zealand, July 8, 2019*, pages 3–11. ACM, 2019. 2

[FH20] Masayuki Fukumitsu and Shingo Hasegawa. A lattice-based provably secure multisignature scheme in quantum random oracle model. In Khoa Nguyen, Wenling Wu, Kwok-Yan Lam, and Huaxiong Wang, editors, *ProvSec 2020*, volume 12505 of *LNCS*, pages 45–64. Springer, Heidelberg, November / December 2020. 2

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987. 3

[FSZ22] Nils Fleischhacker, Mark Simkin, and Zhenfei Zhang. Squirrel: Efficient synchronized multi-signatures from lattices. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 1109–1123. ACM Press, November 2022. 2

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 3

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 469–477. ACM Press, June 2015. 2, 4

[IN83] Kazuharu Itakura and Katsuhiro Nakamura. A public-key cryptosystem suitable for digital multisignatures. *NEC Research & Development*, (71):1–8, 1983. 1

[LN17] Vadim Lyubashevsky and Gregory Neven. One-shot verifiable encryption from lattices. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 293–323. Springer, Heidelberg, April / May 2017. 5

[LNTW19] Benoît Libert, Khoa Nguyen, Benjamin Hong Meng Tan, and Huaxiong Wang. Zero-knowledge elementary databases with more expressive queries. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part I*, volume 11442 of *LNCS*, pages 255–285. Springer, Heidelberg, April 2019. 2, 4

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-LWE cryptography. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 35–54. Springer, Heidelberg, May 2013. 6

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009. 2, 9, 22

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Heidelberg, April 2012. 2, 6, 9, 22

[MJ19] Changshe Ma and Mei Jiang. Practical lattice-based multisignature schemes for blockchains. *IEEE Access*, 7:179765–179778, 2019. 2

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012. 3, 4, 31

[MP13] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with small parameters. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 21–39. Springer, Heidelberg, August 2013. 6

[MPSW19] Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple schnorr multi-signatures with applications to bitcoin. *Des. Codes Cryptogr.*, 87(9):2139–2164, 2019. 1, 7, 20

[MR04] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. In *45th FOCS*, pages 372–381. IEEE Computer Society Press, October 2004. 6

[MWLD10] Changshe Ma, Jian Weng, Yingjiu Li, and Robert H. Deng. Efficient discrete logarithm based multi-signature scheme in the plain public key model. *Des. Codes Cryptogr.*, 54(2):121–133, 2010. 1

[NRS21] Jonas Nick, Tim Ruffing, and Yannick Seurin. MuSig2: Simple two-round Schnorr multi-signatures. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 189–221, Virtual Event, August 2021. Springer, Heidelberg. 1, 4, 5, 7, 20

[NRSW20] Jonas Nick, Tim Ruffing, Yannick Seurin, and Pieter Wuille. MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1717–1731. ACM Press, November 2020. 1

[Oka93] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 31–53. Springer, Heidelberg, August 1993. 4

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398. Springer, Heidelberg, May 1996. 20

[PW23] Jiaxin Pan and Benedikt Wagner. Chopsticks: Fork-free two-round multi-signatures from non-interactive assumptions. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 597–627. Springer, Heidelberg, April 2023. 1, 23

$\mathsf{H}_{\mathrm{agg}}(L, \mathbf{t}_j)$	$\mathsf{H}_{\mathrm{com}}(\tilde{\mathbf{t}}, \mu)$	$\mathsf{H}_{\mathrm{sig}}(\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}})$

if $\mathcal{T}_{\mathrm{agg}}[L, \mathbf{t}_j] = \perp$ **then**
  if $\mathbf{t}_1^* \notin L$ **then**
    $\mathcal{T}_{\mathrm{agg}}[L, \mathbf{t}_j] \leftarrow\$ C$
  else
    for $\mathbf{t}_i \in L \wedge \mathbf{t}_i \neq \mathbf{t}_1^*$ **do**
      $\mathcal{T}_{\mathrm{agg}}[L, \mathbf{t}_i] \leftarrow\$ C$
    $\mathsf{ctr}_{\mathrm{agg}} := \mathsf{ctr}_{\mathrm{agg}} + 1$
    $\mathcal{T}_{\mathrm{agg}}[L, \mathbf{t}_1^*] := a^{(\mathsf{ctr}_{\mathrm{agg}})}$
return $\mathcal{T}_{\mathrm{agg}}[L, \mathbf{t}_j]$
	if $\mathcal{T}_{\mathrm{com}}[\tilde{\mathbf{t}}, \mu] = \perp$ **then**
	  if $\lvert\mathcal{T}_{\mathrm{com}}\rvert \neq i^* - 1$ **then**
	    $\mathcal{T}_{\mathrm{com}}[\tilde{\mathbf{t}}, \mu] \leftarrow\$ R_q^{k \times l'}$
	  else $\mathcal{T}_{\mathrm{com}}[\tilde{\mathbf{t}}, \mu] := \mathbf{B}$
	return $\mathcal{T}_{\mathrm{com}}[\tilde{\mathbf{t}}, \mu]$
		if $\mathcal{T}_{\mathrm{sig}}[\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}}] = \perp$ **then**
		  $\mathsf{ctr}_{\mathrm{sig}} := \mathsf{ctr}_{\mathrm{sig}} + 1$
		  $\mathcal{T}_{\mathrm{sig}}[\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}}] := c^{(\mathsf{ctr}_{\mathrm{sig}})}$
		return $\mathcal{T}_{\mathrm{sig}}[\tilde{\mathbf{t}}, \mu, \tilde{\mathbf{w}}]$

Figure 6: How $\mathcal{W}$ answers hash queries.

[RY07] Thomas Ristenpart and Scott Yilek. The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks. In Moni Naor, editor, *EUROCRYPT 2007*, volume 4515 of *LNCS*, pages 228–245. Springer, Heidelberg, May 2007. 1

[STV+16] Ewa Syta, Iulia Tamas, Dylan Visher, David Isaac Wolinsky, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ismail Khoffi, and Bryan Ford. Keeping authorities "honest or bust" with decentralized witness cosigning. In *2016 IEEE Symposium on Security and Privacy*, pages 526–545. IEEE Computer Society Press, May 2016. 1

[TZ23] Stefano Tessaro and Chenzhi Zhu. Threshold and multi-signature schemes from linear hash functions. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 628–658. Springer, Heidelberg, April 2023. 1

A Proof of Lemma 10

Proof. We first construct a wrapper $\mathcal{W}$. It takes as inputs $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{t}_1^*$ which are uniformly distributed over $R_q^{k \times l}$, $R_q^{k \times l'}$, and R_q^k, respectively, and $a^{(1)}, \ldots, a^{(Q_\mathrm{h})}$, $c^{(1)}, \ldots, c^{(Q_\mathrm{h})}$ which are uniformly distributed over C. It initializes three empty tables $\mathcal{T}_{\mathrm{agg}}$, $\mathcal{T}_{\mathrm{com}}$, and $\mathcal{T}_{\mathrm{sig}}$ and sets two counters $\mathsf{ctr}_{\mathrm{agg}}$ and $\mathsf{ctr}_{\mathrm{sig}}$ to 0. It runs $\mathcal{A}$ on inputs $\bar{\mathbf{A}} = [\mathbf{A}|\mathbf{I}]$ and $\mathbf{t}$. In the first stage, $\mathcal{A}$ outputs an index i^*.

In the second stage, $\mathcal{W}$ answers hash queries from $\mathcal{A}$ as described in Fig. 6. We underline several points that are worth noting. First, to answer a fresh query $\mathsf{H}_{\mathrm{agg}}(L, \mathbf{t}_j)$ with $\mathbf{t}_1^* \in L$, $\mathcal{W}$ programs $\mathcal{T}_{\mathrm{agg}}[L, \mathbf{t}_1^*]$ with the next unused value among $\{a^{(1)}, \ldots, a^{(Q_\mathrm{h})}\}$ after setting $\mathcal{T}_{\mathrm{agg}}[L, \mathbf{t}_i] \leftarrow\$ C$ for all other $\mathbf{t}_i \in L$. Second, $\mathcal{W}$ answers the i^*-th fresh $\mathsf{H}_{\mathrm{com}}$ query with $\mathbf{B}$. Third, $\mathcal{W}$ answers every fresh $\mathsf{H}_{\mathrm{sig}}$ with the next unused value among $\{c^{(1)}, \ldots, c^{(Q_\mathrm{h})}\}$.

At the end, adversary $\mathcal{A}$ outputs a list of public key L^* and a forged multi-signature $\tilde{\sigma}^* = (c, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{e}})$. Let $\mathbf{t}^*$ and μ^* be the inputs to the i^*-th fresh $\mathsf{H}_{\mathrm{com}}$ query. The wrapper $\mathcal{W}$ checks the following conditions and outputs $I = 0$ if any of them does not hold:

- $\mathbf{t}_1^* \in L^*$, $\mathsf{KAgg}(L^*) = \mathbf{t}^*$, and $\mathsf{Vf}(\mathbf{t}^*, \mu^*, \tilde{\sigma}^*) = 1$.
- Event AggOrd does not occur, where AggOrd is defined as that there exists a set of public key L that contains $\mathbf{t}_1^*$, was queried to $\mathsf{H}_{\mathrm{agg}}$ later than the i^*-th fresh $\mathsf{H}_{\mathrm{com}}$ query, and satisfies $\mathsf{KAgg}(L) = \mathbf{t}^*$.

- Event AggCol does not occur, where AggCol is defined as that there exists two different sets of public key L and L' that have been queried to H_{agg}, both contain $\mathbf{t}_1^*$, and satisfy $\mathsf{KAgg}(L) = \mathsf{KAgg}(L')$.

If all the three conditions hold, then $\mathcal{W}$ outputs

$$I, J, \tilde{\mathbf{w}}, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{e}}, \tilde{\mathbf{t}}^*, c, L^*, a_1, \ldots, a_n,$$

where $\tilde{\mathbf{w}} = \mathbf{A}\tilde{\mathbf{z}} + \mathbf{B}\tilde{\mathbf{r}} + \tilde{\mathbf{e}} - c\tilde{\mathbf{t}}^*$, $a_i = \mathcal{T}_{\text{agg}}[L^*, \mathbf{t}_i]$ for every $\mathbf{t}_i \in L^*$, I is the index such that $c^{(I)} = c$ was assigned to $\mathcal{T}_{\text{sig}}[\tilde{\mathbf{t}}^*, \mu^*, \tilde{\mathbf{w}}]$, and J is the index such that $a^{(J)} = a_1$ was assigned to $\mathcal{T}_{\text{agg}}[L^*, \mathbf{t}_1]$.

The first condition that $\mathcal{W}$ checks is exactly the success condition of $\mathcal{A}$ in $\mathsf{sel\text{-}UF\text{-}KOA}_{\mathsf{DualMS}}$. The game simulated by $\mathcal{W}$ differs from $\mathsf{sel\text{-}UF\text{-}KOA}_{\mathsf{DualMS}}$ only in the distribution of target public key. In particular, in $\mathsf{sel\text{-}UF\text{-}KOA}_{\mathsf{DualMS}}$, $\mathbf{t}_1^* = \bar{\mathbf{A}}\mathbf{s}_1$ for a secret key $\mathbf{s}_1$ uniformly distributed over S_η^l, while $\mathcal{W}$ gives $\mathbf{t}_1^*$ uniformly distributed over R_q^k to $\mathcal{A}$. Apparently there exists a distinguisher $\mathcal{D}$, which essentially runs $\mathcal{W}$, such that the first condition holds with probability at least $\mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}KOA}}_{\mathsf{DualMS}}(\mathcal{A}) - \mathbf{Adv}_{\mathsf{MLWE}_{q,k,l,\eta}}(\mathcal{D})$. Lemma 11 and Lemma 12 bound the probability of AggOrd and AggCol by $Q_{\text{h}}/|C| + (2/q^{N/2})^k$ and $Q_{\text{h}}(Q_{\text{h}} - 1)/|C| + (2/q^{N/2})^k$, respectively. Let $\varepsilon_{\mathcal{W}}$ denote the probability that $\mathcal{W}$ outputs $I \neq 0$. Then we have

$$\mathbf{Adv}^{\mathsf{sel\text{-}UF\text{-}KOA}}_{\mathsf{DualMS}}(\mathcal{A}) \leq \varepsilon_{\mathcal{W}} + \mathbf{Adv}_{\mathsf{MLWE}_{q,k,l,\eta}}(\mathcal{D}) + \frac{Q_{\text{h}}^2}{|C|} + 2(\frac{2}{q^{N/2}})^k. \tag{5}$$

Inner forking algorithm. Then we construct the inner forking algorithm $\mathcal{B}'$. It takes as inputs $\mathbf{A}$, $\mathbf{B}$, $\mathbf{t}_1^*$ which are uniformly distributed over $R_q^{k\times l}$, $R_q^{k\times l'}$, and R_q^k, respectively, and $a^{(1)}, \ldots, a^{(Q_{\text{h}})}$ which are uniformly distributed over C. It runs $\mathsf{Fork}_{\mathcal{W}}(X)$, where the input X consists of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{t}_1^*$, and $a^{(1)}, \ldots, a^{(Q_{\text{h}})}$, and the inputs $c^{(1)}, \ldots, c^{(Q_{\text{h}})}$ of $\mathcal{W}$ are interpreted as $h_1, \ldots, h_Q$ in Lemma 13. All the outputs of $\mathcal{W}$ other than I are interpreted as the side output Y in Lemma 13.

If $\mathsf{Fork}_{\mathcal{W}}$ outputs $\perp$, then $\mathcal{B}'$ outputs $J = 0$. If $\mathsf{Fork}_{\mathcal{W}}$ gives a non-$\perp$ output, then suppose the side outputs Y and Y' of $\mathcal{W}$ in the two executions consist of

$$J, \tilde{\mathbf{w}}, \tilde{\mathbf{z}}, \tilde{\mathbf{r}}, \tilde{\mathbf{e}}, \tilde{\mathbf{t}}^*, c, L^*, a_1, \ldots, a_n, \text{ and } J', \tilde{\mathbf{w}}', \tilde{\mathbf{z}}', \tilde{\mathbf{r}}', \tilde{\mathbf{e}}', \tilde{\mathbf{t}}^{*\prime}, c', L^{*\prime}, a_1', \ldots, a_n',$$

respectively. We argue that:

$$J = J', \tilde{\mathbf{w}} = \tilde{\mathbf{w}}', \tilde{\mathbf{t}}^* = \tilde{\mathbf{t}}^{*\prime}, L^* = L^{*\prime}, a_1 = a_1', \ldots, a_n = a_n'.$$

This is because the two executions of $\mathcal{W}$ are identical until $\mathcal{W}$ assigns $c^{(I)}$ to $\mathcal{T}_{\text{sig}}[\tilde{\mathbf{t}}^*, \mu^*, \tilde{\mathbf{w}}]$. It immediately follows that $\tilde{\mathbf{t}}^* = \tilde{\mathbf{t}}^{*\prime}$ and $\tilde{\mathbf{w}} = \tilde{\mathbf{w}}'$ as index I are the same in the two executions. Note that AggOrd and AggCol did not occur in both executions, and we assumed at the beginning of Section 4 that query $\mathsf{H}_{\text{com}}(\tilde{\mathbf{t}}^*, \mu^*)$ was made earlier than $\mathsf{H}_{\text{sig}}(\tilde{\mathbf{t}}^*, \mu^*, \tilde{\mathbf{w}})$. Hence, L^* was queried to H_{agg} earlier than the assignment to $\mathcal{T}_{\text{sig}}[\tilde{\mathbf{t}}^*, \mu^*, \tilde{\mathbf{w}}]$, and it is the only public-key list ever queried that is aggregated into $\tilde{\mathbf{t}}^*$. All the other equations follow. On the other hand, we have $c \neq c'$ by the description of $\mathsf{Fork}_{\mathcal{W}}$ in Lemma 13.

Then we have

$$\mathbf{A}\tilde{\mathbf{z}} + \mathbf{B}\tilde{\mathbf{r}} + \tilde{\mathbf{e}} = \tilde{\mathbf{w}} + c\tilde{\mathbf{t}}^* \text{ and } \mathbf{A}\tilde{\mathbf{z}}' + \mathbf{B}\tilde{\mathbf{r}}' + \tilde{\mathbf{e}}' = \tilde{\mathbf{w}} + c'\tilde{\mathbf{t}}^*$$

as $\mathcal{W}$ outputs $I \neq 0$ only if the forgery given by $\mathcal{A}$ is valid. Subtract the second equation from the first, and let $\hat{\mathbf{z}} = \tilde{\mathbf{z}} - \tilde{\mathbf{z}}'$, $\hat{\mathbf{r}} = \tilde{\mathbf{r}} - \tilde{\mathbf{r}}'$, $\hat{\mathbf{e}} = \tilde{\mathbf{e}} - \tilde{\mathbf{e}}'$ and $\hat{c} = c - c' \neq 0$. Then we have

$$\mathbf{A}\hat{\mathbf{z}} + \mathbf{B}\hat{\mathbf{r}} + \hat{\mathbf{e}} = \hat{c}\tilde{\mathbf{t}}^* = \hat{c}\sum_{i=1}^{n} a_i \mathbf{t}_i,$$

where $\{\mathbf{t}_1, \ldots, \mathbf{t}_n\} = L^*$ and $\mathbf{t}_1 = \mathbf{t}_1^*$. Algorithm $\mathcal{B}'$ outputs

$$J, \hat{\mathbf{z}}, \hat{\mathbf{r}}, \hat{\mathbf{e}}, \hat{c}, L^*, a_1, \ldots, a_n.$$

Let $\varepsilon_{\mathcal{B}'}$ be the probability that $\mathcal{B}'$ outputs $J \neq 0$. By Lemma 13,

$$\varepsilon_{\mathcal{W}} \leq \frac{Q_{\mathrm{h}}}{|C|} + \sqrt{Q_{\mathrm{h}} \varepsilon_{\mathcal{B}'}}. \tag{6}$$

Outer forking algorithm. Finally, we construct the outer forking algorithm $\mathcal{B}$. It takes as inputs a $k \times (1 + l + l')$ matrix over R_q and partitions it as $[\mathbf{t}_1^*|\mathbf{A}|\mathbf{B}]$. It runs $\mathsf{Fork}_{\mathcal{B}'}(X)$, where the input X consists of $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{t}_1^*$, and the inputs $a^{(1)}, \ldots, a^{(Q_{\mathrm{h}})}$ of $\mathcal{B}'$ are interpreted as $h_1, \ldots, h_Q$ in Lemma 13. The output J of $\mathcal{B}'$ are interpreted as I in Lemma 13, and all other outputs as side output Y.

If $\mathsf{Fork}_{\mathcal{B}'}$ outputs $J \neq 0$, then suppose the side outputs Y and Y' of $\mathcal{B}'$ in the two executions consist of

$$\hat{\mathbf{z}}, \hat{\mathbf{r}}, \hat{\mathbf{e}}, \hat{c}, L^* = \{\mathbf{t}_1, \ldots, \mathbf{t}_n\}, a_1, \ldots, a_n, \text{ and } \hat{\mathbf{z}}', \hat{\mathbf{r}}', \hat{\mathbf{e}}', \hat{c}', L'^* = \{\mathbf{t}_1', \ldots, \mathbf{t}_n'\}, a_1', \ldots, a_n',$$

respectively. We argue that $L^* = L^{*\prime}$, and $a_i = a_i'$ for every $\mathbf{t}_i \neq \mathbf{t}_1^*$. This is because the four executions of $\mathcal{W}$ during the two executions of $\mathcal{B}'$ are identical until $\mathcal{W}$ assigns $a^{(J)}$ to $\mathcal{T}_{\mathrm{agg}}[L^*, \mathbf{t}_1^*]$. It immediately follows that $L^* = L^{*\prime}$ by the fact that index J are the same in the four executions. Other equations follow from the fact that $\mathcal{W}$ assigns $\mathcal{T}_{\mathrm{agg}}[L^*, \mathbf{t}_i]$ for $\mathbf{t}_i \neq \mathbf{t}_1^*$ earlier than $\mathcal{T}_{\mathrm{agg}}[L^*, \mathbf{t}_1^*]$. On the other hand, we have $a_1 \neq a_1'$ again by the definition of $\mathsf{Fork}_{\mathcal{B}'}$.

Suppose that $\mathbf{t}_1^*$ occurs n^* times in L^*. Then we have

$$\mathbf{A}\hat{\mathbf{z}} + \mathbf{B}\hat{\mathbf{r}} + \hat{\mathbf{e}} = \hat{c}(n^* a_1 \mathbf{t}_1^* + \sum_{\mathbf{t}_i \neq \mathbf{t}_1^*} a_i \mathbf{t}_i) \text{ and } \mathbf{A}\hat{\mathbf{z}}' + \mathbf{B}\hat{\mathbf{r}}' + \hat{\mathbf{e}}' = \hat{c}'(n^* a_1' \mathbf{t}_1^* + \sum_{\mathbf{t}_i \neq \mathbf{t}_1^*} a_i \mathbf{t}_i).$$

By multiplying the first equation by $\hat{c}'$ and the second by $\hat{c}$, we have

$$\mathbf{A}(\hat{c}'\hat{\mathbf{z}} - \hat{c}\hat{\mathbf{z}}') + \mathbf{B}(\hat{c}'\hat{\mathbf{r}} - \hat{c}\hat{\mathbf{r}}') + \hat{c}'\hat{\mathbf{e}} - \hat{c}\hat{\mathbf{e}}' = n^* \hat{c}\hat{c}'(a_1 - a_1')\mathbf{t}_1^*.$$

Then we rearrange the equation and have

$$[\mathbf{t}_1^*|\mathbf{A}|\mathbf{B}|\mathbf{I}] \begin{bmatrix} n^* \hat{c}\hat{c}'(a_1 - a_1') \\ \hat{c}'\hat{\mathbf{z}} - \hat{c}\hat{\mathbf{z}}' \\ \hat{c}'\hat{\mathbf{r}} - \hat{c}\hat{\mathbf{r}}' \\ \hat{c}'\hat{\mathbf{e}} - \hat{c}\hat{\mathbf{e}}' \end{bmatrix} = 0.$$

Note that $\hat{c}$, $\hat{c}'$, and $\hat{a}$ are not zeros and also not zero-divisors by Lemma 1, so $n^*\hat{c}\hat{c}'\hat{a} \neq 0$. The L^2-norm of the vector is

$$\begin{aligned} &\sqrt{\|n^*\hat{c}\hat{c}'\hat{a}\|_2^2 + \|\hat{c}'\hat{\mathbf{z}} - \hat{c}\hat{\mathbf{z}}'\|_2^2 + \|\hat{c}'\hat{\mathbf{r}} - \hat{c}\hat{\mathbf{r}}'\|_2^2 + \|\hat{c}'\hat{\mathbf{e}} - \hat{c}\hat{\mathbf{e}}'\|_2^2} \\ &\leq 8\kappa\sqrt{\hat{n}^2\kappa^3 + B_n^2 + B_n'^2 + B_n''^2} = \beta. \end{aligned}$$

To see the inequality, first note that $\|n^*\hat{c}\hat{c}'\hat{a}\|_2 \leq \hat{n}\|\hat{c}\|_1\|\hat{c}'\|_1\|\hat{a}\|_2$, where $\|\hat{c}\|_1$, $\|\hat{c}'\|_1 \leq 2\kappa$ and $\|\hat{a}\|_2 \leq 2\sqrt{\kappa}$. Moreover, we have $\|\hat{c}'\hat{\mathbf{z}} - \hat{c}\hat{\mathbf{z}}'\|_2 \leq \|\hat{c}'\|_1\|\hat{\mathbf{z}}\|_2 + \|\hat{c}\|_1\|\hat{\mathbf{z}}'\|_2$, where $\|\hat{\mathbf{z}}\|_2$, $\|\hat{\mathbf{z}}'\|_2 \leq 2B_n$. Similar bounds hold for $\|\hat{c}'\hat{\mathbf{r}} - \hat{c}\hat{\mathbf{r}}'\|_2$ and $\|\hat{c}'\hat{\mathbf{e}} - \hat{c}\hat{\mathbf{e}}'\|_2$. Thus, $\mathcal{B}$ successfully obtains a solution to $\mathsf{MSIS}_{q,k,1+l+l',\beta}$ with respect to $[\mathbf{t}_1^*|\mathbf{A}|\mathbf{B}]$.

By Lemma 13, we have

$$\varepsilon_{\mathcal{B}'} \leq \frac{Q_{\mathrm{h}}}{|C|} + \sqrt{Q_{\mathrm{h}} \mathbf{Adv}_{\mathsf{MSIS}_{q,k,1+l+l',\beta}}(\mathcal{B})}. \tag{7}$$

Combining Eqs. (5) to (7) we have

$$\begin{aligned} \mathbf{Adv}_{\mathsf{DualMS}}^{\mathsf{sel\text{-}UF\text{-}KOA}}(\mathcal{A}) \leq \sqrt{\frac{Q_{\mathrm{h}}^2}{|C|} + Q_{\mathrm{h}}\sqrt{Q_{\mathrm{h}}\mathbf{Adv}_{\mathsf{MSIS}_{q,k,1+l+l',\beta}}(\mathcal{B})}} \\ + \mathbf{Adv}_{\mathsf{MLWE}_{q,k,l,\eta}}(\mathcal{D}) + \frac{Q_{\mathrm{h}}(Q_{\mathrm{h}}+1)}{|C|} + 2(\frac{2}{q^{N/2}})^k. \end{aligned}$$

□

Lemma 11 ([BTT22b])**.**

$$\Pr[\textsc{AggOrd}] \leq \frac{Q_h}{|C|} + (\frac{2}{q^{N/2}})^k.$$

Proof. Note that $\mathbf{t}_1^* = [t_{1,1}^*, \ldots, t_{1,k}^*]^\intercal$ is a vector that consists of k elements uniformly distributed over R_q. By Lemma 1, except with probability at most $(2/q^{N/2})^k$, one of the k elements is invertible. We condition on this event and assume t_1^* is invertible without loss of generality. We bound the probability that a list of public keys $L = \{\mathbf{t}_1^*, \mathbf{t}_2, \ldots, \mathbf{t}_n\}$ queried to H_{agg} later than the i^*-th fresh H_{com} query, where $\mathbf{t}_1^*$ occurs n^* times, is aggregated into $\tilde{\mathbf{t}}^*$. This event occurs only if the first element of $\mathbf{t}_1^*$, …, $\mathbf{t}_n$ aggregate into the first element of $\tilde{\mathbf{t}}^*$. More specifically, let $t_{i,1}$ be the first element of $\mathbf{t}_i$ for $i \in [n]$ and $\tilde{t}_1$ the first element of $\tilde{\mathbf{t}}^*$. Then L is aggregated into $\tilde{\mathbf{t}}^*$ only if

$$n^* a_1 t_{1,1}^* + \sum_{\mathbf{t}_i \neq \mathbf{t}_1^*} a_i t_{i,1} = \tilde{t}_1.$$

As an integer, n^* is invertible, so it follows that

$$a_1 = n^{*-1} t_1^{*-1}(\tilde{t}_1 - \sum_{\mathbf{t}_i \neq \mathbf{t}_1^*} a_i t_{i,1}).$$

Since a_1 is uniformly distributed over C, it occurs with probability at most $1/|C|$. We can conclude the proof by the union bound. □

Lemma 12 ([BTT22b])**.**

$$\Pr[\textsc{AggCol}] \leq \frac{Q_h(Q_h-1)}{|C|} + (\frac{2}{q^{N/2}})^k.$$

Proof. The reason is basically the same as Lemma 11. Here we can bound the probability that $\mathsf{KAgg}(L) = \mathsf{KAgg}(L')$ with $1/|C|$ for each pair of public-key sets L and L' queried to KAgg, and the lemma follows from the union bound. □

Lemma 13 (General forking lemma [BN06])**.** *Fix an interger $Q \geq 1$, a set $\mathcal{H}$ of size $|\mathcal{H}| \geq 2$. Let $\mathcal{A}$ be a randomized algorithm that takes as inputs X, h_1, …, h_Q, takes as random coin tosses from set $\mathcal{R}$, and outputs tuple (I, Y) where $I \in \{0, \ldots, Q\}$ and Y is what we call "side output". Let $\mathcal{D}_X$ be an unspecified distribution. Let*

$$\varepsilon = \Pr[I \geq 1 : X \leftarrow\!\$\, \mathcal{D}_X; h_1, \ldots, h_Q \leftarrow\!\$\, \mathcal{H}; (I, Y) \leftarrow \mathcal{A}(X, h_1, \ldots, h_Q)].$$

Define the forking algorithm $\mathsf{Fork}_\mathcal{A}$ with respect to $\mathcal{A}$ as follows:

$\mathsf{Fork}_\mathcal{A}(X)$

$\rho \leftarrow\!\$\, \mathcal{R}$
$h_1 \ldots, h_Q \leftarrow\!\$\, \mathcal{H}$
$(I, Y) \leftarrow \mathcal{A}(X, h_1, \ldots, h_Q; \rho)$
if $I = 0$ **then return** $\perp$
$h'_I, \ldots, h'_Q \leftarrow\!\$\, \mathcal{H}$
$(I', Y') \leftarrow \mathcal{A}(X, h_1, \ldots, h_{I-1}, h'_I, \ldots, h'_Q; \rho)$
if $I \neq I' \vee h_I = h'_I$ **then return** $\perp$
return (I, Y, Y')

Let

$$\varepsilon' = \Pr[\mathsf{Fork}_{\mathcal{A}}(X) \neq \perp : X \leftarrow\$ \mathcal{D}_X].$$

Then

$$\varepsilon \leq \frac{Q}{|\mathcal{H}|} + \sqrt{Q\varepsilon'}.$$

B Parameter Setting for MuSig-L

In MuSig-L, a signer's pre-commitments in the first round form a matrix that supports trapdoor sampling in simulation. We consider computationally distinguishable trapdoors based on MLWE, so $w_1 + w_2$ pre-commitments form a $k \times (k + w_1 + w_2)$ matrix $\mathbf{W} = [\mathbf{I}|\mathbf{W}_1|\mathbf{W}_2] = [\bar{\mathbf{W}}|\mathbf{W}_2]$. In the simulation, $\mathbf{W}_1$ is uniform distributed, and $\mathbf{W}_2$ is set as $\mathbf{G} - \bar{\mathbf{W}}\mathbf{R}$, where $\mathbf{G}$ is the gadget trapdoor $\mathbf{I} \otimes \mathbf{g}$ with $\mathbf{g}^\intercal = [1, b, b^2, \ldots, b^{\lceil \log(q) \rceil - 1}]$, and $\mathbf{R}$ is a Gaussian distributed trapdoor.

Let $m = k + w_1 + w_2$ be the width of matrix $\mathbf{W}$. According to [BTT22b], we approximate the standard deviation σ_1 of MuSig-L signatures as $\sigma_1 \approx \sigma_y \sigma_b \sqrt{N(w_1 + w_2)(l + k)}$. The simulation requires

$$\sigma_b \approx \bar{\sigma} \cdot s_1(\sqrt{\Sigma_{\mathbf{G}}}) \cdot (\sqrt{N(k + w_1)} + \sqrt{N w_2}),$$

with $\bar{\sigma}$ the standard deviation of $\mathbf{R}$ and $s_1(\sqrt{\Sigma_{\mathbf{G}}})$ a parameter related to the structure of $\mathbf{G}$, and

$$\sigma_y \approx 64\sqrt{2}/\pi \cdot \sigma_b q^{k/(l+k)} \sqrt{N(2 + N + \log(N(l + k)))}.$$

Here we have some parameters to decide about matrix $\mathbf{W}$. First, the base b reduce w_2 by a logarithmic factor but increases $s_1(\sqrt{\Sigma_{\mathbf{G}}})$ by a linear factor [MP12]. Hence, we set $b = 2$, so $w_2 = k\lceil \log(q) \rceil$. Second, there is a trade-off between w_1 and $\bar{\sigma}$. We observe that w_1 is much smaller than w_2, so increasing w_1 does not essentially affect the performance. Thus, we set very small $\bar{\sigma} = 1$, and then decide w_1 according to the trapdoor indistinguishability related to MLWE. In our parameter settings, w_2 is more than 768, while typically $w_1 < 20$ is enough for MLWE hardness. We can easily adjust the hardness with very slight effect on the efficiency. Therefore, trapdoor indistinguishability is not a crucial security criteria and not included in Table 2.

C A Note on The Optimization

In the first version of this paper[5], we did not realize the simple and useful trick of aggregating $\mathbf{z}'_i$ and $\mathbf{r}'_i$. In this updated version, we apply the optimization but only in the very last step SAgg of signature generation. This already makes pronounced improvement in signature size while allows us to make as few changes as possible. In particular, Sign_1 and Sign_2 are unchanged, so we do not need to modify the security proofs until Section 4.3.

Alternatively, we can apply the optimization earlier. Since $\mathbf{z}''_1$ and $\mathbf{r}''_1$ will be eventually aggregated to $\mathbf{e}_1 \in R^k$, we can let the signer directly sample $\mathbf{y}'_1 \leftarrow D_s^l$, $\mathbf{r}'_1 \leftarrow D_{s'}^{l'}$, and a vector $\mathbf{d}_1 \in R^k$ in Sign_1. Then in Sign_2, $\mathbf{d}_1$ can be viewed as split into $\mathbf{y}''_1 \leftarrow D_s^k$ and $\mathbf{r}''_1$ for rejection sampling, while the second-round message is directly $\mathbf{e}_1$.

The benefit is that the Gaussian width of $\mathbf{e}_1$ can be reduced. Recall that the Gaussian width s (resp. s') should be larger than a bound s_1 (resp. s'_1) to hide the secret key (resp. dual key) and another bound s_2 (resp. s'_2) to apply Lemma 5 in the dual signing (resp. normal signing) procedures. Hence, we set $s > \max(s_1, s_2)$ and $s' > \max(s'_1, s'_2)$. However, in the normal signing, it suffices to split $\mathbf{d}_1$ into $\mathbf{y}''_1$ with Gaussian width s_1 for rejection sampling and $\mathbf{r}''_1$ with Gaussian width s'_2 for Lemma 5. Similarly, in the dual signing, $\mathbf{d}_1$ is split into $\mathbf{p}''$ with Gaussian width s'_1 and $\mathbf{z}''_1$ with Gaussian width s_2. Therefore, rather than $\sqrt{s^2 + s'^2}$ where $s > \max(s_1, s_2)$ and $s' > \max(s'_1, s'_2)$ as in our scheme specification, the Gaussian width of $\mathbf{e}_1$ can be reduced to $\max(\sqrt{s_1^2 + s_2'^2}, \sqrt{s_1'^2 + s_2^2})$.

[5] https://eprint.iacr.org/archive/2023/263/20230222:193121