Generic Ring-Signature Transforms for Fiat-Shamir with Aborts and Hash-and-Sign with Retry

Haruhisa Kosuge[1][0009−0001−9976−3344] and Koutarou Suzuki[2][0000−0003−3464−1933]

[1] NTT Social Informatics Laboratories, Japan
hrhs.kosuge@ntt.com
[2] Toyohashi University of Technology, Japan
suzuki@cs.tut.ac.jp

Abstract. Ring signatures provide signer anonymity for ad hoc sets of public keys. Generic Abe-Ohkubo-Suzuki (AOS) transforms are well understood for plain Fiat–Shamir and hash-and-sign signatures, but not for their rejection-sampling variants: Fiat–Shamir with aborts (FSwA) and hash-and-sign with retry (HSwR). We formalize AOS ring transforms for FSwA and HSwR and analyze their security in the quantum random-oracle model. For unforgeability, we reduce security under adaptive ring-signing queries to security without signing queries using essentially the same assumptions as in security proofs for the corresponding ordinary signatures; handling adaptive corruption additionally requires the simulation property to remain valid after key exposure. Strong unforgeability additionally requires response uniqueness and special soundness for FSwA, or second-preimage resistance and non-invertibility for HSwR. The residual no-signing-query requirement admits generic reductions, but with loss exponential in the number of honest keys. Finally, an ML-DSA case study shows that augmented public keys achieve the required key-exposure simulation while identifying its limitations; we also identify conditions for candidate HSwR instantiations.

Keywords: ring signature, Fiat–Shamir with aborts, hash-and-sign with retry

Table of Contents

1 Introduction 2
2 Preliminaries 5
3 Proof Techniques in QROM 11
4 Ring Transforms for Fiat-Shamir with Aborts and Hash-and-Sign with Retry 12
5 Security Proof of the Ring Transform for Fiat-Shamir with Aborts 13
6 Security Proof of the Ring Transform for Hash-and-Sign with Retry 18
7 Case Study of ML-DSA 21
8 Case Study of Hash-and-Sign Ring Signatures 23
A Algorithmic Templates 26
B Detailed ML-DSA Specification 27
C Simulation and Entropy Properties of Ring-ML-DSA 32
D Concrete Bounds for Hardness Assumptions for Ring-ML-DSA 34
E Detailed HSwR Instantiation Guidance 39
F Missing Proofs 40

1 Introduction

Post-quantum ring signatures. Ring signatures allow a user to sign with respect to an ad hoc set of public keys (a ring) without revealing the signer [RST01], providing anonymous authentication without trusted setup or a group manager. Quantum algorithms, however, threaten ring signatures based on factoring and discrete logarithms. Post-quantum ring signatures have been constructed from lattice assumptions [BK10, ABB$^+$13, LLNW16, LAZ19b, ESS$^+$19, DAZ19, EZS$^+$19, LNS21, CGH$^+$21, YEL$^+$21, GJK24b], multivariate assumptions [MP17, FR25], symmetric-key primitives and accumulators [DRS17, KKW18, BLSS22], and isogeny-based assumptions [BKP20, CHH$^+$21].

Most such constructions are tailored to a particular algebraic setting. Abe, Ohkubo, and Suzuki (AOS), in contrast, gave a generic cyclic transform from Fiat–Shamir and hash-and-sign signatures to ring signatures [AOS02].

Fiat-Shamir with aborts and hash-and-sign with retry. Many practical post-quantum signatures, however, do not fit the plain FS or HS paradigms exactly because signing may require repeated trials. In Fiat-Shamir with aborts (FSwA), introduced for lattice-based signatures [Lyu09], the signer computes an identification-protocol commitment and response, restarting upon rejection. Hash-and-sign with retry (HSwR) captures schemes in which trapdoor inversion may fail: the signer resamples the public salt until inversion succeeds [SSH11, KX24].

These two paradigms capture important classes of post-quantum signatures. FSwA underlies ML-DSA [Nat24] and other lattice-based signatures, while HSwR captures the public-salt retrying inversion structure used by schemes such as salt-UOV and QR-UOV [BCD$^+$24, AFI$^+$24]. A common treatment of the corresponding AOS transforms is therefore important for understanding when these signature designs can be lifted to ring signatures. Because the intended instantiations must resist quantum adversaries, the corresponding AOS transforms must be analyzed in the quantum random oracle model (QROM).

Security notions. Our aim is to formulate each security goal under its strongest natural attack model and then weaken it as needed. We therefore follow Bender, Katz, and Morselli [BKM06], allowing insider corruption for unforgeability and full key exposure for signer anonymity. Strong existential unforgeability under a chosen-ring attack (sEUF-CRA) allows adaptive ring-signing queries and treats a new signature on a previously signed ring-message pair as a forgery; relaxing freshness gives ordinary EUF-CRA, while removing the signing oracle gives existential unforgeability under a no-ring attack (EUF-NRA). For privacy, key-exposure anonymity (ke-ANON) allows exposure of the challenge keys, while ordinary anonymity (ANON) omits this exposure. The NRA/CRA organization follows the recent QROM analysis of AOS by Beckmann and Majenz [BM26].

Previous studies and the remaining gap. Abe, Ohkubo, and Suzuki [AOS02] introduced the original AOS construction and gave instantiations based on Schnorr and RSA signatures. Their security analysis is in the classical ROM. Das [Das20] gave an RLWE-based linkable ring signature following the FSwA paradigm. For the AOS Type-H transform, Lu, Au, and Zhang proved unconditional anonymity and ordinary unforgeability in the classical ROM [LAZ19a]. For the plain FS AOS transform, Beckmann and Majenz [BM26] gave a QROM security analysis of the FS ring transform. Their analysis first replaces signing answers with simulated ones and then reduces forgery to the security of the underlying identification scheme. Neither work gives a QROM analysis of the HS ring transform or treats retry after failed inversion; no corresponding QROM analysis is known for the FSwA or HSwR ring transforms.

The existing analysis does not extend directly to these transforms. Their signing procedures require different random-oracle programming strategies, particularly when an abort or an inversion failure causes a trial to be discarded. Moreover, we seek to base their security on weaker simulation assumptions that reproduce only the distributions exposed by successful signing executions. Such assumptions do not themselves explain how discarded trials should be simulated. A separate analysis is therefore required. We therefore ask:

Once the FSwA and HSwR ring transforms are formalized, which assumptions of the underlying primitives suffice for each security notion in the QROM?

In this paper, we provide generic transforms from FSwA and HSwR to ring signature based on AOS construction and answer the above question as in Table 1.

Table 1. QROM reduction landscape for AOS ring transforms of FS/FSwA and HS/HSwR. Entries without a citation are established in this work. Here, W-REC denotes witness-recovery security [BM26, Definition 13]. Section 2.2 defines the notions of (ke-)acHVZK ((key-exposure) accepting honest-verifier zero knowledge), IMP (impersonation resistance), CUR (computational unique response), and CSS (computational special soundness) for FS/FSwA, as well as (ke-)PS ((key-exposure) preimage simulatability), INV (non-invertibility), and SPR (second-preimage resistance) for HS/HSwR.

Input	EUF-NRA	EUF-CRA	sEUF-CRA	ANON	ke-ANON
FS	IMP [BM26]	EUF-NRA, HVZK [BM26]	EUF-NRA, HVZK, CUR, W-REC [BM26]	HVZK [BM26]	ke-HVZK [BM26]
FSwA	IMP [BM26]	EUF-NRA, ke-acHVZK	EUF-NRA, ke-acHVZK, CUR, CSS	acHVZK	ke-acHVZK
HS	INV	EUF-NRA, ke-PS	EUF-NRA, ke-PS, SPR, INV	PS	ke-PS
HSwR	INV	EUF-NRA, ke-PS	EUF-NRA, ke-PS, SPR, INV	PS	ke-PS

1.1 Contribution

We formalize AOS ring transforms assuming FSwA/HSwR and analyze their security in the QROM. Our results show that the EUF-CRA-to-EUF-NRA reduction relies on ordinary-signature assumptions, except that adaptive corruption requires the assumptions to remain valid under key exposure. Residual EUF-NRA admits generic reductions with exponential loss in the number of honest keys. Furthermore, by applying the FSwA ring transform to ML-DSA, we demonstrate its feasibility and identify its limitations. Our contributions are as follows.

- *Generic transforms.* We formalize ring transforms for Fiat–Shamir with aborts (FSwA) and hash-and-sign with retry (HSwR), covering rejection after response generation and retry after failed inversion.
- *QROM security reductions.* We give rejection-aware QROM reductions for unforgeability and anonymity, summarized in Table 1. Our reductions build on the same accepted-output simulation properties used for the corresponding ordinary signatures: accepting honest-verifier zero knowledge (acHVZK) for FSwA and preimage simulatability (PS) for HSwR. To handle adaptive corruption, including corruption after signing, we require their key-exposure variants, ke-acHVZK and ke-PS. Under these properties, we reduce EUF-CRA to EUF-NRA. For anonymity, the ordinary variants yield ANON, while their key-exposure variants yield ke-ANON. The reduction of sEUF-CRA to EUF-NRA additionally uses computational unique response (CUR) and computational special soundness (CSS) for FSwA, or second-preimage resistance (SPR) and non-invertibility (INV) for HSwR. These additional local properties are likewise those used for the corresponding ordinary signatures. The ring-specific residual problem is EUF-NRA. Beckmann and Majenz reduce it to impersonation (IMP) for FSwA [BM26]; for HSwR, we extend their approach to obtain a reduction to INV. Both generic reductions lose exponentially in the number of honest keys, and tightening them remains open.
- *Post-quantum applicability.* We clarify the applicability of the FSwA ring transform to ML-DSA [Nat24] and its current limitations. Public simulation is enabled by augmenting the public key with $\boldsymbol{t}_0$, and we relate the resulting Ring-ML-DSA security conditions to concrete module-lattice problems. However, publishing $\boldsymbol{t}_0$ changes the standard ML-DSA public-key relation, and its security implications, the required adversarial-key conditions, and a tight EUF-NRA reduction remain open. We similarly identify the conditions still missing for complete HSwR instantiations of Falcon [PFH+22], salt-UOV [BCD+24], QR-UOV [AFI+24], and MAYO [BCC+24].

Generic ring transforms in brief. Let $R = (pk_0, \ldots, pk_{n-1})$ be a ring, and let j denote the signer position. Both transforms form a cyclic challenge chain from local values (a_i, c_i, z_i): $c_{i+1} = \mathsf{H}(i+1, R, m, a_i)$ and $c_n = c_0$. For FSwA, a_i, c_i, and z_i are a commitment, challenge, and response, respectively. The signer first generates its real commitment a_j and retains the state needed to answer a challenge. This commitment determines the challenge at the next position $c_{j+1} = \mathsf{H}(j+1, R, m, a_j)$. The chain then proceeds cyclically through all non-signer positions. Once c_i is fixed, an accepting pair (a_i, z_i) for pk_i and c_i is generated publicly, and a_i determines the next challenge c_{i+1}. After the chain returns to position j, it fixes c_j, and the signer uses the retained state to compute the real response z_j. If z_j is rejected, the signer restarts the entire chain. Given the signature $(c_0, z_0, \ldots, z_{n-1})$, verification reconstructs each a_i from (pk_i, c_i, z_i), recomputes c_{i+1}, and accepts if the chain returns to c_0.

For HSwR, let $\mathsf{F}(pk, \cdot)$ denote the public map of the underlying trapdoor function. We require its target space, which is also the challenge space, to be a finite group with an efficiently computable operation $\circ$ and inverse. This is a restriction of our transform: trapdoor functions without such a target-group structure are not covered. At a non-signer position, one publicly samples z_i, computes $y_i = \mathsf{F}(pk_i, z_i)$, and sets $a_i = y_i \circ c_i^{-1}$. Thus, $\mathsf{F}(pk_i, z_i) = a_i \circ c_i$. At the signer position, it samples a_j and inverts $a_j \circ c_j$ using the trapdoor after the cyclic chain; signing restarts from taking a_j if inversion fails.

Technical overview of QROM security reductions. Our reductions first replace the signing oracle by a keyless full-chain simulator. Unlike the plain AOS analysis of Beckmann and Majenz [BM26, Theorem 7], one signing query may involve several trials while revealing only the accepted one. For each trial, the simulator samples a complete cyclic chain $(c_0, \ldots, c_{n-1})$ and reprograms a random oracle H as $\mathsf{H}(i+1, R, m, a_i) = c_{i+1}$ at every position. Next, it cancels this reprogramming for the rejected trials using the technique of Fallahpour, Fehr, and Huang [FFH25]; thus, only the accepted chain remains reprogrammed in H. Finally, the accepted signer component (a_j, c_j, z_j) is then replaced using ke-acHVZK for FSwA [KLS18, BBD+23] or ke-PS for HSwR [KX24]. These notions are weaker than simulation notions covering rejected transcripts: they simulate only the accepted component and impose no simulation requirement on rejected trials. Their key-exposure variants allow the simulator to answer corruption queries even after signing.

Our full-chain reprogramming strategy is essential for relying on weaker distributional assumptions of underlying primitives. For FSwA, this matches the random-challenge experiment of (ke-)acHVZK and avoids simulation for any challenge value. For HSwR, we require challenges and function outputs to lie in the same finite group and valid preimages to be publicly sampleable for every well-formed key. Sampling z_i independently of c_i and setting $a_i = \mathsf{F}(pk_i, z_i) \circ c_i^{-1}$ makes a_i uniform, since the map from c_i to a_i is a permutation. As a_i is part of the random-oracle input, this uniformity provides the entropy needed to make the programmed point hard to predict, without requiring uniform outputs from F. Likewise, uniform a_j makes the signer target uniform, allowing (ke-)PS to replace its accepted preimage.

After these hops, the signing oracle no longer uses a secret key. EUF-CRA then reduces directly to EUF-NRA because freshness excludes previously signed ring-message pairs. For sEUF-CRA, the forged pair may have been signed before, so we use the properties of the underlying FSwA or HSwR primitive to show that the forgery does not depend on points programmed for a signing query. This requires ruling out three types of overlap between the forged chain and an accepted signing chain for the same ring-message pair. At position i, write (a_i^*, c_i^*, z_i^*) for the forged local component and (a_i', c_i', z_i') for the component of a selected accepted signing chain.

(i) $(a_i^*, c_i^*) = (a_i', c_i')$ *but* $z_i^* \neq z_i'$. For FSwA, CUR requires that producing two distinct accepting responses for the same public key, commitment, and challenge be computationally infeasible, and therefore rules out this case. For HSwR, SPR requires that, given one preimage, finding a distinct preimage of the same target be computationally infeasible. Since both z_i^* and z_i' are preimages of $a_i^* \circ c_i^*$, this case violates SPR.

(ii) $a_i^* = a_i'$ *but* $c_i^* \neq c_i'$. For FSwA, CSS requires that producing two accepting transcripts with the same public key and commitment but distinct challenges be computationally infeasible [3], and therefore rules out this case. For HSwR, this case gives $\mathsf{F}(pk_i, z_i^*) \circ (c_i^*)^{-1} = \mathsf{F}(pk_i, z_i') \circ (c_i')^{-1}$ with $c_i^* \neq c_i'$, but there is

[3] Standard special soundness requires an extractor that recovers a witness. Our CSS formulation instead requires that producing such a pair be computationally infeasible.

no natural CSS-style hardness assumption for this relation.[4] We instead reduce this event to INV using measure-and-reprogramming [DFM20, Theorem 2]. For a uniform INV target y, the reduction reprograms the forged challenge as $c_i^* = (a_i^*)^{-1} \circ y$, so the forged response z_i^* yields a preimage of y.

(iii) *Remaining cycle overlap.* If neither local-reuse case occurs, intersecting chains must split and later rejoin. At the first rejoining position k, $a_{k-1}^* \neq a'_{k-1}$, $c_k^* = c_k$, and hence $\mathsf{H}(k, R, m, a_{k-1}^*) = \mathsf{H}(k, R, m, a'_{k-1})$. Thus, two distinct random-oracle inputs have the same output, yielding a collision.

Once these cases are excluded, the signing-specific programming can be disabled without invalidating the forgery, yielding the reduction to EUF-NRA.

The above reductions leave EUF-NRA as a separate, scheme-specific task. For FSwA, Beckmann and Majenz reduce EUF-NRA to IMP [BM26, Theorem 8]. For HSwR, we apply the same reduction to the local relation $\mathsf{F}(pk_i, z_i) = a_i \circ c_i$ and then give a tight reduction from IMP to INV. The EUF-NRA-to-IMP step loses exponentially in the number of keys, making both complete reductions non-tight.

For anonymity, we apply the same full-chain simulation to both the signing oracle and the challenge signature. In the anonymity game, the adversary must distinguish which of two secret keys produced the challenge signature. Replacing (a_j, c_j, z_j) using acHVZK for FSwA or PS for HSwR makes the signer component distributed like every non-signer component, rendering the chain independent of j and yielding ANON. The key-exposure variants preserve this replacement after exposure of the challenge users' key-generation coins, yielding ke-ANON.

1.2 Open Problem and Deployment Caveat

The generic EUF-NRA reductions incur a loss exponential in the number of honest keys. Obtaining a tighter QROM reduction for EUF-NRA remains open. As a deployment caveat, our ML-DSA case study is not directly compatible with the standard public keys specified by FIPS 204 [Nat24]: simulation at non-signer positions requires publishing $\boldsymbol{t}_0$, which is secret in standard ML-DSA. Using our ring construction therefore requires an augmented public key, and our conditional analysis does not establish that publishing $\boldsymbol{t}_0$ preserves all security properties of standard ML-DSA.

1.3 Organization

Section 2 introduces the required definitions, and Section 3 presents the QROM tools. We define the FSwA and HSwR ring transforms in Section 4 and analyze their security in Sections 5 and 6. Finally, Sections 7 and 8 discuss the ML-DSA case study and candidate HSwR instantiations. The appendices provide detailed specifications, analyses, and missing proofs.

2 Preliminaries

The security parameter is denoted by $\lambda \in \mathbb{Z}^+$. PPT and QPT stand for probabilistic and quantum polynomial time, respectively. For $n \in \mathbb{Z}^+$, we let $[n] \coloneqq \{0, \ldots, n-1\}$. For a distribution D, the notation $x \leftarrow D$ means that x is sampled according to D. If $\mathcal{S}$ is a finite set, then $x \leftarrow \mathcal{S}$ denotes uniform sampling from $\mathcal{S}$. $H_\infty(x)$ denotes min-entropy $H_\infty(x) = -\log \max_{\bar{x}} \Pr\big[x = \bar{x} : x \leftarrow D\big]$ of $x \leftarrow D$. For an algorithm A with access to oracles O, we write $\mathsf{out} \leftarrow \mathsf{A}^O(\mathsf{inp})$ for its output on input inp. We write $\mathsf{out} \coloneqq \mathsf{A}(\mathsf{inp}; r)$ when making its randomness r explicit. For a function $f\colon \{0,1\}^n \to \{0,1\}^m$, *quantum access* to f is modeled by the unitary oracle $O_f\colon |x\rangle\,|y\rangle \longmapsto |x\rangle\,|y \oplus f(x)\rangle$. We write $\mathsf{A}^{|f\rangle, g}$ to distinguish quantum access to f from classical access to g. For a function $f\colon \mathcal{X} \to \mathcal{Y}$, the notation $f[x \mapsto h]$ denotes the function obtained by reprogramming $f(x)$ to h.

[4] The relation is easy to satisfy if the challenge can be chosen. Its hardness here comes from c_i^* being fixed by the random oracle.

Game $\mathsf{Expt}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$ / $\mathsf{Expt}^{\mathsf{(s)EUF\text{-}CRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$	$\textsc{RingSign}(R, m, s)$
1 : $\mathcal{Q} \coloneqq \emptyset$	1 : **if** $s \notin [N]$ **then return** $\perp$
2 : $\mathcal{C}_{pk} \coloneqq \emptyset$	2 : **if** $pk_s \notin R$ **then return** $\perp$
3 : **for** $s \in [N]$ **do**	3 : $\sigma \leftarrow \mathsf{Sign}(R, m, sk_s)$
4 : $\quad \omega_s \leftarrow \mathcal{W}(1^\lambda), (pk_s, sk_s) \leftarrow \mathsf{Gen}(1^\lambda; \omega_s)$	4 : $\mathcal{Q} \coloneqq \mathcal{Q} \cup \{(R, m, \sigma)\}$
5 : $\mathcal{R} \coloneqq \{pk_0, \ldots, pk_{N-1}\}$	5 : **return** σ
6 : $(R^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\textsc{Corr}}(\mathcal{R})$ $\quad$ // EUF-NRA	
7 : $(R^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\textsc{RingSign},\textsc{Corr}}(\mathcal{R})$ $\quad$ // (s)EUF-CRA	$\textsc{Corr}(s)$
8 : **if** $R^* \not\subseteq \mathcal{R}$ **then return** 0	1 : **if** $s \notin [N]$ **then return** $\perp$
9 : **if** $R^* \cap \mathcal{C}_{pk} \neq \emptyset$ **then return** 0	2 : $\mathcal{C}_{pk} \coloneqq \mathcal{C}_{pk} \cup \{pk_s\}$
10 : **if** $\exists \sigma : (R^*, m^*, \sigma) \in \mathcal{Q}$ **then return** 0 $\quad$ // EUF-CRA	3 : **return** sk_s
11 : **if** $(R^*, m^*, \sigma^*) \in \mathcal{Q}$ **then return** 0 $\quad$ // sEUF-CRA	
12 : **return** $\mathsf{Verify}(R^*, m^*, \sigma^*)$	

Fig. 1. Games for EUF-NRA, EUF-CRA, and sEUF-CRA security of ring signatures (Definition 2.2).

2.1 Ring Signatures and the AOS Generic Transform

A ring is a nonempty ordered tuple of distinct, well-formed public keys with an injective, prefix-free encoding.

Definition 2.1. *A ring signature scheme* RS *consists of PPT algorithms* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$*:*

- $\mathsf{Gen}(1^\lambda) \to (pk, sk)$*: outputs a key pair* (pk, sk)*.*
- $\mathsf{Sign}(R, m, sk_j) \to \sigma$*: outputs a ring signature* $\sigma \in \mathcal{S}$ *for ring* $R = (pk_0, \ldots, pk_{n-1})$*, message* $m \in \mathcal{M}$*, and signing key* sk_j *corresponding to some* $pk_j \in R$*.*
- $\mathsf{Verify}(R, m, \sigma) \to 0/1$*: outputs* 1 *or* 0 *for ring* $R = (pk_0, \ldots, pk_{n-1})$*, message* $m \in \mathcal{M}$*, and signature* σ*.*

We require statistical correctness. For any message $m \in \mathcal{M}$*, any integer* $n \in \mathbb{Z}^+$*, and any signer index* $j \in [n]$*, we require*

$$\Pr\left[\begin{array}{l}(pk_0, sk_0), \ldots, (pk_{n-1}, sk_{n-1}) \leftarrow \mathsf{Gen}(1^\lambda), \\ R \coloneqq (pk_0, \ldots, pk_{n-1}), \sigma \leftarrow \mathsf{Sign}(R, m, sk_j)\end{array} : \mathsf{Verify}(R, m, \sigma) = 1\right] \geq 1 - \delta(\lambda)$$

for some negligible function δ*.*

Security notions. We consider existential and strong existential unforgeability under chosen-ring attacks (EUF-CRA and sEUF-CRA), existential unforgeability under no-ring attacks (EUF-NRA), and ordinary and key-exposure anonymity (ANON and ke-ANON). Our unforgeability notions follow the insider-corruption model of Bender, Katz, and Morselli, while ke-ANON follows their full-key-exposure model [BKM06]. EUF-CRA and sEUF-CRA signing queries may contain adversarial keys, and anonymity challenge rings may do so as well, provided both challenge members are honest; a forgery ring must contain only honest, uncorrupted keys. Corruption and ring-signing remain available in either order. Both anonymity notions allow ring-signing queries before and after the challenge. ANON has no corruption oracle, whereas ke-ANON allows corruption before and after the challenge, reveals key-generation coins, and permits corruption of both challenge members.

Definition 2.2 (Unforgeability notions for ring signatures). *Let* $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *be a ring signature scheme. For any adversary* $\mathcal{A}$*, define*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{RS},\mathcal{A}}(\lambda) \coloneqq \Pr[\mathsf{Expt}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda) = 1],$$
$$\mathsf{Adv}^{\mathsf{EUF\text{-}CRA}}_{\mathsf{RS},\mathcal{A}}(\lambda) \coloneqq \Pr[\mathsf{Expt}^{\mathsf{EUF\text{-}CRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda) = 1],$$
$$\mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{RS},\mathcal{A}}(\lambda) \coloneqq \Pr[\mathsf{Expt}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda) = 1],$$

Game $\mathsf{Expt}^{\mathsf{ANON}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$ / $\mathsf{Expt}^{\mathsf{ke\text{-}ANON}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$	$\textsc{RingSign}(R, m, s)$
1 : **for** $s \in [N]$ **do**	1 : **if** $s \notin [N]$ **then return** $\perp$
2 : $\quad \omega_s \leftarrow \mathcal{W}(1^\lambda), (pk_s, sk_s) \leftarrow \mathsf{Gen}(1^\lambda; \omega_s)$	2 : **if** $pk_s \notin R$ **then return** $\perp$
3 : $\mathcal{R} \coloneqq \{pk_0, \ldots, pk_{N-1}\}$	3 : $\sigma \leftarrow \mathsf{Sign}(R, m, sk_s)$
4 : $(R, m, s_0, s_1, st) \leftarrow \mathcal{A}^{\textsc{RingSign}}(\mathcal{R})$ $\quad$ // ANON	4 : **return** σ
5 : $(R, m, s_0, s_1, st) \leftarrow \mathcal{A}^{\textsc{RingSign},\textsc{Corr}}(\mathcal{R})$ $\quad$ // ke-ANON	
6 : **if** $s_0 \notin [N]$ **or** $s_1 \notin [N]$ **or** $s_0 = s_1$	$\textsc{Corr}(s)$
7 : $\quad$ **then return** 0	1 : **if** $s \notin [N]$ **then return** $\perp$
8 : **if** $pk_{s_0} \notin R$ **or** $pk_{s_1} \notin R$	2 : **return** ω_s
9 : $\quad$ **then return** 0	
10 : $b \leftarrow \{0, 1\}$	
11 : $\sigma \leftarrow \mathsf{Sign}(R, m, sk_{s_b})$	
12 : $b' \leftarrow \mathcal{A}^{\textsc{RingSign}}(st, \sigma)$ $\quad$ // ANON	
13 : $b' \leftarrow \mathcal{A}^{\textsc{RingSign},\textsc{Corr}}(st, \sigma)$ $\quad$ // ke-ANON	
14 : **return** $(b = b')$	

Fig. 2. Games for ring-signature anonymity (Definition 2.3).

where the experiments $\mathsf{Expt}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$, $\mathsf{Expt}^{\mathsf{EUF\text{-}CRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$, *and* $\mathsf{Expt}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$ *are described in Figure 1. Within these experiments, a query* $\textsc{Corr}(s)$ *is called a* post-signing corruption *if the adversary previously obtained a successful response to* $\textsc{RingSign}(R, m, s)$ *for some* (R, m). *We say that* RS *satisfies EUF-NRA, EUF-CRA, or sEUF-CRA if the corresponding advantage above is negligible in* λ *for every QPT adversary* $\mathcal{A}$.

Definition 2.3 (Anonymity notions for ring signatures). *Let* $\mathsf{RS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *be a ring signature scheme. For any adversary* $\mathcal{A}$, *define*

$$\mathsf{Adv}^{\mathsf{ANON}}_{\mathsf{RS},\mathcal{A}}(\lambda) \coloneqq \left| \Pr[\mathsf{Expt}^{\mathsf{ANON}}_{\mathsf{RS},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2} \right|,$$

$$\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{RS},\mathcal{A}}(\lambda) \coloneqq \left| \Pr[\mathsf{Expt}^{\mathsf{ke\text{-}ANON}}_{\mathsf{RS},\mathcal{A}}(1^\lambda) = 1] - \frac{1}{2} \right|,$$

where the experiments $\mathsf{Expt}^{\mathsf{ANON}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$ *and* $\mathsf{Expt}^{\mathsf{ke\text{-}ANON}}_{\mathsf{RS},\mathcal{A}}(1^\lambda)$ *are described in Figure 2. We say that* RS *satisfies ANON (resp., ke-ANON) if* $\mathsf{Adv}^{\mathsf{ANON}}_{\mathsf{RS},\mathcal{A}}(\lambda)$ *(resp.,* $\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{RS},\mathcal{A}}(\lambda)$*) is negligible in* λ *for every QPT adversary* $\mathcal{A}$.

AOS generic transform. The Abe–Ohkubo–Suzuki (AOS) transform [AOS02] converts a canonical identification scheme into a ring signature scheme. It links the signer's real transcript and the other simulated transcripts in a cyclic hash chain, choosing the real branch to close the chain. Verification reconstructs the commitments and accepts when the final challenge equals c_0; the cycle hides the real branch. See Appendix A.2 for the detailed specification.

2.2 Fiat-Shamir with Aborts and Hash-and-Sign with Retry

We define the underlying primitives and signature paradigms, followed by the correctness and security properties used in our proofs.

Underlying primitives and common conventions. We formulate the two paradigms from the following primitives.

Definition 2.4 (Three-round identification scheme). *An identification scheme* ID *consists of the following tuple of PPT algorithms* $(\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$*:*

- $\mathsf{Gen}(1^\lambda) \to (pk, sk)$*: outputs a key pair* (pk, sk)*.*
- $\mathsf{P}_1(sk) \to (a, st)$*: outputs a commitment* a *and state* st*.*
- $\mathsf{P}_2(sk, st, c) \to z/\perp$*: deterministically outputs a response* z *or* $\perp$ *for challenge* $c \in \mathcal{C}$*.*
- $\mathsf{V}(pk, a, c, z) \to 0/1$*: deterministically verifies transcript* (a, c, z)*.*

We assume perfect correctness. Namely, for any honestly generated (pk, sk) *and any* $c \in \mathcal{C}$*, if* $(a, st) \leftarrow \mathsf{P}_1(sk)$ *and* $z \leftarrow \mathsf{P}_2(sk, st, c)$ *with* $z \neq \perp$*, then* $\mathsf{V}(pk, a, c, z) = 1$*. We also say that* ID *has* (α, δ)*-min-entropy commitments if, except with probability at most* δ *over* $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$*, the commitment* a *output by* $(a, st) \leftarrow \mathsf{P}_1(sk)$ *satisfies* $H_\infty(a) \geq \alpha$*.*

Definition 2.5 (Trapdoor function). *A trapdoor function* TDF *consists of a tuple of PPT algorithms* $(\mathsf{Gen}, \mathsf{F}, \mathsf{Inv})$*. For each well-formed public key* pk*, it has a public domain* $\mathcal{X}_{pk}$ *and a public target space* $\mathcal{Y}$*. The algorithms are as follows.*

- $\mathsf{Gen}(1^\lambda) \to (pk, sk)$*: outputs a key pair* (pk, sk)*.*
- $\mathsf{F}(pk, x) \to y \in \mathcal{Y}$*: deterministically evaluates the function at* $x \in \mathcal{X}_{pk}$*.*
- $\mathsf{Inv}(sk, y) \to x/\perp$*: outputs a preimage* x *or* $\perp$*.*

We assume perfect correctness: for any honestly generated (pk, sk) *and any* y*, if* $x \leftarrow \mathsf{Inv}(sk, y)$ *with* $x \neq \perp$*, then* $\mathsf{F}(pk, x) = y$*.*

To model strong key exposure, we write $\omega \leftarrow \mathcal{W}(1^\lambda)$ and $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda; \omega)$. Our key-exposure notions reveal ω, which determines both keys and all values derived during key generation, rather than only the final secret-key encoding. This allows reductions to answer post-signing or post-challenge corruption queries consistently and may constitute strictly stronger exposure.

Signature paradigms. Fiat-Shamir with aborts (FSwA) and hash-and-sign with retry (HSwR) share an outer retry loop but use an identification scheme and a trapdoor function, respectively. The FSwA signer computes a commitment and a response (a, z) and restarts when z is rejected, while the HSwR signer samples a salt r and restarts until $\mathsf{H}(m, r)$ is invertible under the trapdoor. For FSwA, we assume commitment recoverability: a deterministic procedure $\mathsf{Rec}(pk, c, z)$ recovers the unique commitment a for which $\mathsf{V}(pk, a, c, z) = 1$, and returns $\perp$ if no such commitment exists. This is the compressed syntax of Dilithium/ML-DSA, whose signatures contain (c, z) rather than a; without recovery, the signature may include a. See the detailed algorithms in Appendix A.1.

Correctness on adversarial keys. As in the key-exposure models of Definitions 2.2 and 2.3, adversaries may place adversarial keys at non-signing positions but use an honest key. Public non-signer algorithms must therefore remain correct on adversarial keys, independently of the distributional notions below.

Definition 2.6 (Universal simulation correctness and simulated commitment min-entropy). *Let* $\mathsf{ID} = (\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ *be an identification scheme with challenge space* $\mathcal{C}$ *and commitment-recovery procedure* Rec*, and let* Sim *be a public algorithm that takes* (pk, c) *as input and outputs* (a, z)*. We regard* Sim *as stateless and assume that each invocation uses independent fresh randomness. For every well-formed public key* pk*, including an adversarially generated one, and every* $c \in \mathcal{C}$*, we consider the following two requirements. First,* $\mathsf{Sim}(pk, c)$ *terminates in expected polynomial time and*

$$\Pr\big[\mathsf{V}(pk, a, c, z) = 1 \ \wedge \ \mathsf{Rec}(pk, c, z) = a : (a, z) \leftarrow \mathsf{Sim}(pk, c)\big] = 1.$$

We then say that Sim *satisfies universal simulation correctness. Second, the marginal distribution of* a *in* $(a, z) \leftarrow \mathsf{Sim}(pk, c)$ *satisfies*

$$-\log \max_{\bar{a}} \Pr\big[a = \bar{a} : (a, z) \leftarrow \mathsf{Sim}(pk, c)\big] \geq \alpha'.$$

We then say that Sim *has* α'*-min-entropy.*

Game $\mathsf{Expt}^{\mathsf{acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(1^\lambda)$ / $\mathsf{Expt}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(1^\lambda)$

$\omega \leftarrow \mathcal{W}(1^\lambda), (pk, sk) \leftarrow \mathsf{Gen}(1^\lambda; \omega)$
$b \leftarrow \{0,1\}$
if $b = 0$ **then**
  $c \leftarrow \mathcal{C}$
  $(a, z) \leftarrow \mathsf{Sim}(pk, c)$
else
  repeat
    $(a, st) \leftarrow \mathsf{P}_1(sk)$
    $c \leftarrow \mathcal{C}$
    $z \leftarrow \mathsf{P}_2(sk, st, c)$
  until $z \neq \bot$
$b' \leftarrow \mathcal{D}(pk, c, a, z)$ // acHVZK
$b' \leftarrow \mathcal{D}(pk, \omega, c, a, z)$ // ke-acHVZK
return $(b = b')$

Fig. 3. Games for acHVZK and ke-acHVZK (Definition 2.8)

Definition 2.7 (Universal sampling correctness). *Let* $\mathsf{TDF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Inv})$ *be a trapdoor function and let* Samp *be a public algorithm that takes* pk *as input. For every well-formed public key* pk*, let* $\mathcal{X}_{pk}$ *denote the domain of* $\mathsf{F}(pk, \cdot)$*. We say that* Samp *satisfies universal sampling correctness if, for every well-formed public key* pk*, including an adversarially generated one,* $\mathsf{Samp}(pk)$ *terminates in expected polynomial time and*

$$\Pr\big[z \in \mathcal{X}_{pk} \ \wedge \ \mathsf{F}(pk, z) \in \mathcal{Y} : z \leftarrow \mathsf{Samp}(pk)\big] = 1.$$

Security properties of the underlying primitives. Existing QROM proofs for FSwA use two related simulation notions. Devevey et al. [DFPS23] analyze statistical HVZK, which also covers aborting transcripts. Barbosa et al. [BBD+23] prove security for FSwA under accepting HVZK (acHVZK), which requires only the simulation of accepting transcripts. Their simulator takes only pk as input and outputs an accepting transcript (a, c, z) whose distribution matches that of an honest execution conditioned on $z \neq \bot$. Instead, we use a challenge-input formulation of acHVZK: the simulator takes (pk, c) and outputs (a, z). In addition, our definition is distributional on average over this uniformly sampled challenge; it does not assert pointwise indistinguishability for any c. We do not claim that a conventional jointly sampled acHVZK definition automatically implies this challenge-input formulation without an additional argument.

Definition 2.8 (Accepting honest-verifier zero-knowledge (acHVZK)). *Let* $\mathsf{ID} = (\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ *be an identification scheme with challenge space* $\mathcal{C}$*. Let* Sim *be a PPT simulator that takes* (pk, c) *as input and outputs* (a, z)*.*

For any QPT distinguisher $\mathcal{D}$*, define*

$$\mathsf{Adv}^{\mathsf{acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(\lambda) \coloneqq 2\left|\Pr[\mathsf{Expt}^{\mathsf{acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(1^\lambda) = 1] - \frac{1}{2}\right|,$$

$$\mathsf{Adv}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(\lambda) \coloneqq 2\left|\Pr[\mathsf{Expt}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(1^\lambda) = 1] - \frac{1}{2}\right|,$$

where the experiments $\mathsf{Expt}^{\mathsf{acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(1^\lambda)$ *and* $\mathsf{Expt}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(1^\lambda)$ *are described in Figure 3. We say that* ID *satisfies acHVZK (resp., ke-acHVZK) if there exists a PPT simulator* Sim *such that* $\mathsf{Adv}^{\mathsf{acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(\lambda)$ *(resp.,* $\mathsf{Adv}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{D}}(\lambda)$*) is negligible in* λ *for every QPT distinguisher* $\mathcal{D}$*.*

For the EUF-NRA security, we consider the reduction to the standard impersonation security of the underlying identification scheme.

Game $\mathsf{Expt}^{\mathsf{PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(1^\lambda)$ / $\mathsf{Expt}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(1^\lambda)$	
1 :	$\omega \leftarrow \mathcal{W}(1^\lambda), (pk, sk) \leftarrow \mathsf{Gen}(1^\lambda; \omega)$
2 :	$b \leftarrow \{0,1\}$
3 :	**if** $b = 0$ **then**
4 :	$\quad x \leftarrow \mathsf{Samp}(pk)$
5 :	**else**
6 :	$\quad$ **repeat**
7 :	$\quad\quad y \leftarrow \mathcal{Y}$
8 :	$\quad\quad x \leftarrow \mathsf{Inv}(sk, y)$
9 :	$\quad$ **until** $x \neq \perp$
10 :	$b' \leftarrow \mathcal{D}(pk, x)$ $\quad$ //PS
11 :	$b' \leftarrow \mathcal{D}(pk, \omega, x)$ $\quad$ //ke-PS
12 :	**return** $(b = b')$

Fig. 4. Games for PS and ke-PS (Definition 2.10)

Definition 2.9 (Impersonation (IMP) [BM26, Definition 10]). *Let* $\mathsf{ID} = (\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ *be an identification scheme with challenge space* $\mathcal{C}$*. For any QPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, define*

$$\mathsf{Adv}^{\mathsf{IMP}}_{\mathsf{ID},\mathcal{A}}(\lambda) \coloneqq \Pr\left[\mathsf{V}(pk, a, c, z) = 1 : \begin{array}{c}(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda), (a, st) \leftarrow \mathcal{A}_1(pk), \\ c \leftarrow \mathcal{C}, z \leftarrow \mathcal{A}_2(st, c)\end{array}\right].$$

We say that ID *satisfies IMP if this advantage is negligible for every QPT adversary* $\mathcal{A}$*.*

For HSwR, preimage simulatability (PS) lets the proof replace a retrying inversion output with a trapdoor-free sample.

Definition 2.10 (Preimage simulatability (PS)). *For any QPT distinguisher* $\mathcal{D}$*, define*

$$\mathsf{Adv}^{\mathsf{PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(\lambda) \coloneqq 2\left|\Pr[\mathsf{Expt}^{\mathsf{PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(1^\lambda) = 1] - \frac{1}{2}\right|,$$

$$\mathsf{Adv}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(\lambda) \coloneqq 2\left|\Pr[\mathsf{Expt}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(1^\lambda) = 1] - \frac{1}{2}\right|,$$

where the experiments $\mathsf{Expt}^{\mathsf{PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(1^\lambda)$ *and* $\mathsf{Expt}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(1^\lambda)$ *are described in Figure 4. We say that* TDF *satisfies PS (resp., ke-PS) if there exists a PPT sampler* Samp *such that* $\mathsf{Adv}^{\mathsf{PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(\lambda)$ *(resp.,* $\mathsf{Adv}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{D}}(\lambda)$*) is negligible in* λ *for every QPT distinguisher* $\mathcal{D}$*.*

For the EUF-NRA security, the proof uses non-invertibility (INV) [HY18].

Definition 2.11 (Non-invertibility (INV)). *Let* $\mathsf{TDF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Inv})$ *be a trapdoor function whose public target space is* $\mathcal{Y}$*. For any QPT adversary* $\mathcal{A}$*, define its non-invertibility advantage by*

$$\mathsf{Adv}^{\mathsf{INV}}_{\mathsf{TDF},\mathcal{A}}(\lambda) \coloneqq \Pr\left[\mathsf{F}(pk, z) = y : (pk, sk) \leftarrow \mathsf{Gen}(1^\lambda), y \leftarrow \mathcal{Y}, z \leftarrow \mathcal{A}(pk, y)\right].$$

We say that TDF *satisfies INV if this advantage is negligible for every QPT adversary* $\mathcal{A}$*.*

Security properties required for sEUF-CRA. The following assumptions are needed only for the strong unforgeability part of the generic ring-signature security proofs. For FSwA, we use computational unique response (CUR) and computational special soundness (CSS).

Definition 2.12 (Computational unique response (CUR) [KLS18, Definition 2.7]). *Let* ID *be an identification scheme. For any QPT adversary* $\mathcal{A}$*, define its CUR advantage by*

$$\mathsf{Adv}^{\mathsf{CUR}}_{\mathsf{ID},\mathcal{A}}(\lambda) \coloneqq \Pr\left[\begin{array}{c} c \in \mathcal{C} \ \wedge \ z \neq z' \\ \wedge \ \mathsf{V}(pk,a,c,z)=1 \\ \wedge \ \mathsf{V}(pk,a,c,z')=1 \end{array} : \begin{array}{c} (pk,sk) \leftarrow \mathsf{Gen}(1^\lambda), \\ (a,c,z,z') \leftarrow \mathcal{A}(pk) \end{array}\right].$$

We say that ID *satisfies CUR if this advantage is negligible for every QPT adversary* $\mathcal{A}$*.*

Definition 2.13 (Computational special soundness (CSS)). *Let* ID *be an identification scheme. For any QPT adversary* $\mathcal{A}$*, define its CSS advantage by*

$$\mathsf{Adv}^{\mathsf{CSS}}_{\mathsf{ID},\mathcal{A}}(\lambda) \coloneqq \Pr\left[\begin{array}{c} c,c' \in \mathcal{C} \ \wedge \ c \neq c' \\ \wedge \ \mathsf{V}(pk,a,c,z)=1 \\ \wedge \ \mathsf{V}(pk,a,c',z')=1 \end{array} : \begin{array}{c} (pk,sk) \leftarrow \mathsf{Gen}(1^\lambda), \\ (a,c,z,c',z') \leftarrow \mathcal{A}(pk) \end{array}\right].$$

We say that ID *satisfies CSS if this advantage is negligible for every QPT adversary* $\mathcal{A}$*.*

For HSwR, we use INV as well as the second-preimage resistance (SPR).

Definition 2.14 (Second-preimage resistance (SPR)). *Let* $\mathsf{TDF} = (\mathsf{Gen},\mathsf{F},\mathsf{Inv})$ *be a trapdoor function and let* Samp *be a sampler over the domain of* F*. For any QPT adversary* $\mathcal{A}$*, define its second-preimage-resistance advantage by*

$$\mathsf{Adv}^{\mathsf{SPR}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}}(\lambda) \coloneqq \Pr\left[\begin{array}{c} z' \neq z \\ \wedge \ \mathsf{F}(pk,z') = \mathsf{F}(pk,z) \end{array} : \begin{array}{c} (pk,sk) \leftarrow \mathsf{Gen}(1^\lambda), \\ z \leftarrow \mathsf{Samp}(pk), z' \leftarrow \mathcal{A}(pk,z) \end{array}\right].$$

We say that TDF *satisfies SPR with respect to* Samp *if this advantage is negligible for every QPT adversary* $\mathcal{A}$*.*

3 Proof Techniques in QROM

Adaptive reprogramming: Adaptive reprogramming replaces random-oracle values at high-entropy points. Such changes are hard to detect [GHHM21]; we use the refinement of Fallahpour, Fehr, and Huang [FFH25].

Lemma 3.1 (Adaptive reprogramming with classical write queries [FFH25, Lemma 1]). *Let* $\mathcal{X}$ *and* $\mathcal{Y}$ *be finite sets, and let* H *be a uniformly random function from* $\mathcal{X}$ *to* $\mathcal{Y}$*. Let* $\epsilon > 0$*, and let* $\{D_i\}_{i\in\mathcal{I}}$ *be a family of distributions over* $\mathcal{X}$ *such that* $\max_{\hat{x}\in\mathcal{X}} \Pr_{x\leftarrow D_i}[x=\hat{x}] \leq \epsilon$ *for every* $i \in \mathcal{I}$*. Define two oracles* $\textsc{Repro}_0$ *and* $\textsc{Repro}_1$ *as follows. On input* $i \in \mathcal{I}$*, the oracle samples* $x \leftarrow D_i$*.* $\textsc{Repro}_0$ *returns* $(x,\mathsf{H}(x))$*, whereas* $\textsc{Repro}_1$ *samples* $y \leftarrow \mathcal{Y}$ *uniformly at random, reprograms* H *at* x *as* $\mathsf{H}[x\mapsto y]$*, and returns* (x,y)*.*

Let $\mathcal{A}$ *be an oracle algorithm that makes one query to* $\textsc{Repro}_b$*, at most* Q_r *queries to* $|\mathsf{H}\rangle$ *before that query, and at most* Q_w *classical reprogramming queries to* H *in expectation before that query. Then*

$$\left|\Pr\left[1 \leftarrow \mathcal{A}^{|\mathsf{H}\rangle,\textsc{Repro}_0}()\right] - \Pr\left[1 \leftarrow \mathcal{A}^{|\mathsf{H}\rangle,\textsc{Repro}_1}()\right]\right| \leq \left(2Q_w + \frac{Q_r}{2}\right)\epsilon + \sqrt{Q_r\epsilon}.$$

Measure-and-reprogram: Measure-and-reprogram converts a random-oracle-dependent forgery into a reduction receiving a fresh challenge. We use the search-game formulation of [KX24] of the technique from [DFM20].

Lemma 3.2 (Measure-and-reprogram [DFM20, Theorem 2]). *Let* $\mathsf{H}\colon \mathcal{X}\to\mathcal{Y}$ *be a random oracle, and let* $\mathcal{A}$ *be a QPT algorithm that makes at most* Q *quantum queries to* H *and outputs* $(x,z)\in\mathcal{X}\times\mathcal{Z}$*. For every predicate* V *over* $\mathcal{X}\times\mathcal{Y}\times\mathcal{Z}$*, there exists a two-stage QPT simulator* $\mathcal{S}=(\mathcal{S}_1,\mathcal{S}_2)$ *such that*

$$\Pr\left[V(x,\mathsf{H}(x),z) : (x,z)\leftarrow\mathcal{A}^{|\mathsf{H}\rangle}()\right] \leq (2Q+1)^2 \Pr\left[\mathsf{V}(x',\theta,z') : \theta\leftarrow\mathcal{Y}, x'\leftarrow\mathcal{S}_1^{|\mathsf{H}\rangle}(), z'\leftarrow\mathcal{S}_2^{|\mathsf{H}[x'\mapsto\theta]\rangle}(\theta)\right].$$

The running time of $\mathcal{S}$ *is polynomial in the running time of* $\mathcal{A}$ *and in* Q*.*

$\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}].\mathsf{Sign}(R, m, sk_j)$

1 : **repeat**
2 : $\quad(a_j, st_j) \leftarrow \mathsf{P}_1(sk_j)$
3 : $\quad c_{j+1} \coloneqq \mathsf{H}(j+1, R, m, a_j)$
4 : $\quad$**for** $i = j+1, \ldots, n-1, 0, \ldots, j-1$ **do**
5 : $\quad\quad(a_i, z_i) \leftarrow \mathsf{Sim}(pk_i, c_i)$
6 : $\quad\quad c_{i+1} \coloneqq \mathsf{H}(i+1, R, m, a_i)$
7 : $\quad z_j \leftarrow \mathsf{P}_2(sk_j, st_j, c_j)$
8 : **until** $z_j \neq \bot$
9 : $\sigma \coloneqq (c_0, z_0, \ldots, z_{n-1})$
10 : **return** σ

$\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}].\mathsf{Verify}(R, m, \sigma)$

1 : $(c_0, z_0, \ldots, z_{n-1}) \coloneqq \mathsf{Parse}(\sigma)$
2 : **for** $i = 0, \ldots, n-1$ **do**
3 : $\quad a_i \coloneqq \mathsf{Rec}(pk_i, c_i, z_i)$
4 : $\quad$**if** $a_i = \bot$ **then return** $\bot$
5 : $\quad c_{i+1} \coloneqq \mathsf{H}(i+1, R, m, a_i)$
6 : **if** $c_n = c_0$ **then return** $\top$
7 : **else return** $\bot$

Fig. 5. FSwA ring transform.

$\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}].\mathsf{Sign}(R, m, sk_j)$

1 : **repeat**
2 : $\quad a_j \leftarrow \mathcal{C}$
3 : $\quad c_{j+1} \coloneqq \mathsf{H}(j+1, R, m, a_j)$
4 : $\quad$**for** $i = j+1, \ldots, n-1, 0, \ldots, j-1$ **do**
5 : $\quad\quad z_i \leftarrow \mathsf{Samp}(pk_i)$
6 : $\quad\quad a_i \coloneqq \mathsf{F}(pk_i, z_i) \circ c_i^{-1}$
7 : $\quad\quad c_{i+1} \coloneqq \mathsf{H}(i+1, R, m, a_i)$
8 : $\quad z_j \leftarrow \mathsf{Inv}(sk_j, a_j \circ c_j)$
9 : **until** $z_j \neq \bot$
10 : $\sigma \coloneqq (c_0, z_0, \ldots, z_{n-1})$
11 : **return** σ

$\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}].\mathsf{Verify}(R, m, \sigma)$

1 : $(c_0, z_0, \ldots, z_{n-1}) \coloneqq \mathsf{Parse}(\sigma)$
2 : **for** $i = 0, \ldots, n-1$ **do**
3 : $\quad a_i \coloneqq \mathsf{F}(pk_i, z_i) \circ c_i^{-1}$
4 : $\quad c_{i+1} \coloneqq \mathsf{H}(i+1, R, m, a_i)$
5 : **if** $c_n = c_0$ **then return** $\top$
6 : **else return** $\bot$

Fig. 6. HSwR ring transform.

Collision resistance: We use Zhandry's collision bound in the QROM [Zha15].

Lemma 3.3 (Random-oracle collision bound [Zha15, Theorem 3.1][Zha12, Corollary 7.5]). *Let* $\mathsf{H}\colon \mathcal{X} \to \mathcal{Y}$ *be a random oracle. Let a QPT algorithm make at most* Q *oracle queries to* H*, where classical oracle queries are counted as quantum queries. Then, it outputs distinct* $x, x' \in \mathcal{X}$ *such that* $\mathsf{H}(x) = \mathsf{H}(x')$ *with probability at most* $632(Q+1)^3/|\mathcal{Y}|$*.*

4 Ring Transforms for Fiat-Shamir with Aborts and Hash-and-Sign with Retry

We give the FSwA and HSwR ring transforms in Figures 5 and 6. For the HSwR transform, we specialize the target space of Definition 2.5 to a finite group $(\mathcal{C}, \circ)$ with efficiently computable operation and inverse, and require $\mathsf{F}(pk, z) \in \mathcal{C}$ for every well-formed public key and domain element accepted by the public sampler. Consequently $(u \circ c^{-1}) \circ c = u$ and, for every $a, y \in \mathcal{C}$, the unique solution of $a \circ c = y$ is $c = a^{-1} \circ y$. This group-compatible typing is part of the HSwR transform; a general TDF whose target space has no such action does not instantiate this construction.

For FSwA, we restate Das's RLWE construction [Das20] at the paradigm level, separating the transform from its original RLWE representation. For HSwR, we generalize the RSA-based AOS construction of [AOS02, Section 5.2] from permutations to trapdoor functions. The AOS cycle shifts the ordinary HSwR retry target $\mathsf{H}(m, r)$ to $a_j \circ c_j$, where the signer chooses a_j and the hash chain determines c_j. Verification recomputes the other components as $a_i = \mathsf{F}(pk_i, z_i) \circ c_i^{-1}$. In both transforms, the outgoing ring position is part of every

challenge-hash input. We use the injective encoding of the integer $i+1 \in \{1,\dots,n\}$ in $\mathsf{H}(i+1, R, m, a_i)$; challenge indices are cyclic, so $c_n = c_0$, but the position tag n is not reduced modulo n. The tag prevents an oracle value programmed for one ring position from being reused at another.

5 Security Proof of the Ring Transform for Fiat-Shamir with Aborts

This section reduces EUF-CRA and sEUF-CRA to EUF-NRA (Section 5.1), EUF-NRA to IMP (Section 5.2), and proves ANON and ke-ANON (Section 5.3) in the FSwA ring transform.

Standing assumptions and notation. Throughout this section, we assume that Sim satisfies universal simulation correctness and has α'-min-entropy simulated commitments on adversarial keys as in Definition 2.6. For honestly generated keys, real signer commitments have (α, δ)-min-entropy as in Definition 2.4. Conditioned on any preceding execution, each honest signing trial aborts with probability at most $\varepsilon < 1$, where $(1-\varepsilon)^{-1}$ is polynomially bounded.

Let N be the number of honestly generated keys, let n bound the ring size, let Q_{H} bound the number of quantum random-oracle queries, and let Q denote the number of signatures to be generated. For any $\theta > 1$, define

$$T(Q) \coloneqq \left\lceil \frac{\theta Q}{1-\varepsilon} \right\rceil, \qquad \tau(Q) \coloneqq \begin{cases} 0, & Q = 0, \\ \exp\left(-\dfrac{(\theta-1)^2 Q}{2\theta}\right), & Q > 0. \end{cases} \tag{1}$$

Define the shared FSwA loss term by

$$\begin{aligned} \Gamma_{\mathrm{FSwA}}(Q) \coloneqq{}& 3T(Q)\left(\sqrt{Q_{\mathsf{H}} 2^{-\alpha}} + (n-1)\sqrt{Q_{\mathsf{H}} 2^{-\alpha'}}\right) + 3T(Q) Q_{\mathsf{H}}\left(2^{-\alpha-1} + (n-1)2^{-\alpha'-1}\right) \\ &+ 7n(T(Q))^2\left(2^{-\alpha} + (n-1)2^{-\alpha'}\right) + 2N\delta + 2\tau(Q). \end{aligned} \tag{2}$$

The unforgeability statements use $Q = Q_{\mathsf{Sign}}$, whereas the anonymity statements use $Q = Q_{\mathsf{Sign}} + 1$.

Remark 5.1 (On universal simulation guarantees). Since the EUF-CRA and sEUF-CRA signing oracle permits adversarially generated public keys, the proofs require termination, simulation correctness, and commitment min-entropy for every adversarial key.

If queried rings are restricted to honestly generated keys, these guarantees are needed only on the honest-key distribution, but the result applies only to the correspondingly restricted security models, not to Definition 2.2 as stated. Certified-key or proof-of-possession models may offer a solution, but require a modified experiment. The same issue applies to (ke-)ANON.

5.1 Unforgeability

The following theorem reduces EUF-CRA to EUF-NRA and ke-acHVZK, and sEUF-CRA additionally to CUR and CSS.

Theorem 5.1 (EUF-NRA to EUF-CRA/sEUF-CRA of the Fiat-Shamir-with-aborts ring transform). *Let $\mathsf{ID} = (\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ be an identification scheme, and let $\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}]$ be the ring signature scheme obtained from the FSwA ring transform with random oracle H. For any QPT adversary $\mathcal{A}$ making at most Q_{Sign} ring-signing queries for rings of size at most n and at most Q_{H} quantum queries to H, let N be the number of honestly generated public keys in the security experiment. There exist QPT adversaries $\mathcal{A}_{\mathrm{nra}}$ and $\mathcal{A}_{\mathrm{hvzk}}$ such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CRA}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}_{\mathrm{nra}}}(\lambda) + N Q_{\mathsf{Sign}}\, \mathsf{Adv}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{A}_{\mathrm{hvzk}}}(\lambda) + \Gamma_{\mathrm{FSwA}}(Q_{\mathsf{Sign}}).$$

For sEUF-CRA, there exist additional QPT adversaries $\mathcal{A}_{\mathrm{cur}}$ and $\mathcal{A}_{\mathrm{css}}$ such that

$$\mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}_{\mathrm{nra}}}(\lambda) + NQ_{\mathsf{Sign}}\,\mathsf{Adv}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{A}_{\mathrm{hvzk}}}(\lambda) + N\,\mathsf{Adv}^{\mathsf{CUR}}_{\mathsf{ID},\mathcal{A}_{\mathrm{cur}}}(\lambda) + N\,\mathsf{Adv}^{\mathsf{CSS}}_{\mathsf{ID},\mathcal{A}_{\mathrm{css}}}(\lambda) + \frac{632(Q_{\mathsf{H}} + nQ_{\mathsf{Sign}} + n + 1)^3}{|\mathcal{C}|} + \Gamma_{\mathrm{FSwA}}(Q_{\mathsf{Sign}}).$$

In both bounds, Γ_{FSwA} is defined in (2). The running times of $\mathcal{A}_{\mathrm{nra}}$, $\mathcal{A}_{\mathrm{hvzk}}$, $\mathcal{A}_{\mathrm{cur}}$, and $\mathcal{A}_{\mathrm{css}}$ are polynomial in the running time of $\mathcal{A}$ and include at most $O(nT(Q_{\mathsf{Sign}}))$ transcript simulations and oracle-programming operations.

Proof sketch. We modify the ring-signing oracle so that it can be simulated in three steps: first, sample the entire challenge chain $(c_{i+1})_i$ and, for every index i, reprogram $\mathsf{H} \coloneqq \mathsf{H}[(i+1, R, m, a_i) \mapsto c_{i+1}]$; second, for each aborted trial, cancel all reprogramming performed for the entire chain; and third, invoke ke-acHVZK to replace each accepted signer transcript with Sim. Commitment min-entropy, adaptive reprogramming, and abort cancellation bound these changes by $\Gamma_{\mathrm{FSwA}}(Q_{\mathsf{Sign}})$ plus the stated ke-acHVZK term.

The resulting signing oracle uses no secret key, so an EUF-CRA forgery is also an EUF-NRA forgery. For sEUF-CRA, however, (R^*, m^*) may have been queried to the signing oracle; freshness only guarantees that the forged signature is different. To obtain an EUF-NRA forgery, we must therefore show that its verification chain does not depend on transcripts or oracle values introduced only to answer signing queries. If a forged local transcript reuses a simulated commitment, then the same challenge with a different response violates CUR, while a different challenge violates CSS. Once these cases are excluded, any remaining use of a programmed signing input forces the distinct forged and simulated cycles to split and later rejoin, yielding a random-oracle collision. Except for these bounded events, verification of the forgery uses no oracle value introduced to answer a signing query, so the same output wins the EUF-NRA experiment.

Proof. We use the games in Figure 7. Let W_i be the event that $\mathcal{A}$ outputs a valid fresh forgery in Game i, denoted by G_i, and write $Q'_{\mathsf{Sign}} = T(Q_{\mathsf{Sign}})$. We first truncate the combined signing loops after Q'_{Sign} trials. The statistical distance from the original game is at most $\tau(Q_{\mathsf{Sign}})$; the conservative term $2\tau(Q_{\mathsf{Sign}})$ in Γ_{FSwA} covers introducing and, where needed, removing this cutoff.

Common game hops. Games 0–3 are common to ordinary and strong unforgeability, differing only in the final winning predicate.

Game 0. This is the truncated EUF-CRA game for $\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}]$.

Game 1. Game 1 samples full-chain challenges uniformly and programs the corresponding H points. The signer position uses a real commitment, whose guessing probability is at most $2^{-\alpha}$ except with probability δ; the other positions use simulated commitments, each with guessing probability at most $2^{-\alpha'}$. Applying Lemma 3.1 position by position gives

$$|\Pr[\mathsf{G}_0 : W_0] - \Pr[\mathsf{G}_1 : W_1]| \le 2n(Q'_{\mathsf{Sign}})^2 2^{-\alpha} + \frac{Q'_{\mathsf{Sign}} Q_{\mathsf{H}}}{2} 2^{-\alpha} + Q'_{\mathsf{Sign}}\sqrt{Q_{\mathsf{H}} 2^{-\alpha}} + (n-1)\left(2n(Q'_{\mathsf{Sign}})^2 2^{-\alpha'} + \frac{Q'_{\mathsf{Sign}} Q_{\mathsf{H}}}{2} 2^{-\alpha'} + Q'_{\mathsf{Sign}}\sqrt{Q_{\mathsf{H}} 2^{-\alpha'}}\right) + N\delta.$$

Game 2. Game 2 retains full-chain programming only for successful signing iterations, discarding the chain when $z_j = \perp$. The abort-cancellation technique of Fallahpour, Fehr, and Huang [FFH25], applied position by position, yields

$$|\Pr[\mathsf{G}_1 : W_1] - \Pr[\mathsf{G}_2 : W_2]| \le 5n(Q'_{\mathsf{Sign}})^2 2^{-\alpha} + Q'_{\mathsf{Sign}} Q_{\mathsf{H}} 2^{-\alpha} + 2Q'_{\mathsf{Sign}}\sqrt{Q_{\mathsf{H}} 2^{-\alpha}} + (n-1)\left(5n(Q'_{\mathsf{Sign}})^2 2^{-\alpha'} + Q'_{\mathsf{Sign}} Q_{\mathsf{H}} 2^{-\alpha'} + 2Q'_{\mathsf{Sign}}\sqrt{Q_{\mathsf{H}} 2^{-\alpha'}}\right) + N\delta.$$

See the detailed proof in Appendix F.1.

G_0-G_6

$\mathcal{Q} \coloneqq \emptyset$
$\mathcal{C}_{pk} \coloneqq \emptyset$
$\mathcal{T} \coloneqq \emptyset$ // G_4-G_6
for $s \in [N]$ **do**
  $\omega_s \leftarrow \mathcal{W}(1^\lambda), (pk_s, sk_s) \leftarrow \mathsf{Gen}(1^\lambda; \omega_s)$
$\mathcal{R} \coloneqq \{pk_0, \ldots, pk_{N-1}\}$
$(R^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\textsc{RingSign},\textsc{Corr},|\mathsf{H}\rangle}(\mathcal{R})$
if $R^* \not\subseteq \mathcal{R}$ **then return** 0
if $R^* \cap \mathcal{C}_{pk} \neq \emptyset$ **then return** 0
if $\exists \sigma : (R^*, m^*, \sigma) \in \mathcal{Q}$ **then** // EUF-CRA
  return 0 // EUF-CRA
if $(R^*, m^*, \sigma^*) \in \mathcal{Q}$ **then** // sEUF-CRA
  return 0 // sEUF-CRA
if $\mathsf{Verify}(R^*, m^*, \sigma^*) = 0$ **then return** 0
if $\mathsf{CollCheck}(R^*, m^*, \sigma^*) = 0$ **then** // G_4-G_6
return 0 // G_4-G_6
return 1

$\textsc{Corr}(s)$

if $s \notin [N]$ **then return** $\bot$
$\mathcal{C}_{pk} \coloneqq \mathcal{C}_{pk} \cup \{pk_s\}$
return sk_s

$\textsc{RingSign}(R, m, s)$ in G_0

if $s \notin [N]$ **then return** $\bot$
if $pk_s \notin R$ **then return** $\bot$
$j \coloneqq$ unique index such that $R[j] = pk_s$
repeat
  $(a_j, st_j) \leftarrow \mathsf{P}_1(sk_s)$
  $c_{j+1} \coloneqq \mathsf{H}(j+1, R, m, a_j)$
  for $i = j+1, \ldots, n-1, 0, \ldots, j-1$ **do**
    $(a_i, z_i) \leftarrow \mathsf{Sim}(R[i], c_i)$
    $c_{i+1} \coloneqq \mathsf{H}(i+1, R, m, a_i)$
  $z_j \leftarrow \mathsf{P}_2(sk_s, st_j, c_j)$
until $z_j \neq \bot$
$\sigma \coloneqq (c_0, z_0, \ldots, z_{n-1})$
$\mathcal{Q} \coloneqq \mathcal{Q} \cup \{(R, m, \sigma)\}$
return σ

$\textsc{RingSign}(R, m, s)$ in G_1-G_2

if $s \notin [N]$ **then return** $\bot$
if $pk_s \notin R$ **then return** $\bot$
$j \coloneqq$ unique index such that $R[j] = pk_s$
repeat
  $(a_j, st_j) \leftarrow \mathsf{P}_1(sk_s)$
  $c_{j+1} \leftarrow \mathcal{C}$
  $\mathsf{H} \coloneqq \mathsf{H}[(j+1, R, m, a_j) \mapsto c_{j+1}]$ // G_1
  $\mathcal{P} \coloneqq \{(j+1, R, m, a_j, c_{j+1})\}$ // G_2
  for $i = j+1, \ldots, n-1, 0, \ldots, j-1$ **do**
    $(a_i, z_i) \leftarrow \mathsf{Sim}(R[i], c_i)$
    $c_{i+1} \leftarrow \mathcal{C}$
    $\mathsf{H} \coloneqq \mathsf{H}[(i+1, R, m, a_i) \mapsto c_{i+1}]$ // G_1
    $\mathcal{P} \coloneqq \mathcal{P} \cup \{(i+1, R, m, a_i, c_{i+1})\}$ // G_2
  $z_j \leftarrow \mathsf{P}_2(sk_s, st_j, c_j)$
  if $z_j \neq \bot$ **then** // G_2
    for $(k, R, m, a, c) \in \mathcal{P}$ **do** // G_2
      $\mathsf{H} \coloneqq \mathsf{H}[(k, R, m, a) \mapsto c]$ // G_2
until $z_j \neq \bot$
$\sigma \coloneqq (c_0, z_0, \ldots, z_{n-1})$
$\mathcal{Q} \coloneqq \mathcal{Q} \cup \{(R, m, \sigma)\}$
return σ

$\textsc{RingSign}(R, m, s)$ in G_3-G_6

if $s \notin [N]$ **then return** $\bot$
if $pk_s \notin R$ **then return** $\bot$
$j \coloneqq$ unique index such that $R[j] = pk_s$
$c_0, \ldots, c_{n-1} \leftarrow \mathcal{C}$
for $i = 0, \ldots, n-1$ **do**
  $(a_i, z_i) \leftarrow \mathsf{Sim}(R[i], c_i)$
  $\mathsf{H} \coloneqq \mathsf{H}[(i+1, R, m, a_i) \mapsto c_{i+1}]$
  $\mathcal{T} \coloneqq \mathcal{T} \cup \{(R, m, i, a_i, c_i, z_i)\}$ // G_4-G_6
$\sigma \coloneqq (c_0, z_0, \ldots, z_{n-1})$
$\mathcal{Q} \coloneqq \mathcal{Q} \cup \{(R, m, \sigma)\}$
return σ

$\mathsf{CollCheck}(R^*, m^*, \sigma^*)$

$(c_0^*, z_0^*, \ldots, z_{n-1}^*) \coloneqq \mathsf{Parse}(\sigma^*)$
for $i = 0, \ldots, n-1$ **do**
  $a_i^* \coloneqq \mathsf{Rec}(R^*[i], c_i^*, z_i^*)$
  $c_{i+1}^* \coloneqq \mathsf{H}(i+1, R^*, m^*, a_i^*)$
if $\exists (R^*, m^*, i, a_i, c_i, z_i) \in \mathcal{T}$:
  $a_i = a_i^* \wedge \big((c_i = c_i^* \wedge z_i \neq z_i^*)$
    $\vee\ (c_i \neq c_i^*)$ // G_5-G_6
    $\vee\ (c_i = c_i^* \wedge z_i = z_i^*)$ // G_6
  $\big)$ **then return** 0
return 1

Fig. 7. Game hops for the FSwA unforgeability proof of $\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}]$ (Definition 2.2); indices are modulo n.

Game 3. Game 3 replaces the signer component by $(a_j, z_j) \leftarrow \mathsf{Sim}(R[j], c_j)$ through a hybrid over signing answers and honest signer keys. Game 2 samples c_j uniformly and retains the first non-aborting trial, matching the real ke-acHVZK branch; Game 3 matches its simulated branch. The experiment's key-generation coins answer later corruption queries for the embedded signer. Applying the hybrid argument over the at most NQ_{Sign} transitions incurs a factor of at most NQ_{Sign}. Hence,

$$|\Pr[\mathsf{G}_2 : W_2] - \Pr[\mathsf{G}_3 : W_3]| \leq NQ_{\mathsf{Sign}}\, \mathsf{Adv}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{A}_{\mathrm{hvzk}}}(\lambda).$$

EUF-CRA conclusion. The EUF-NRA adversary answers signing queries as in Game 3 without signing keys. A fresh forgery can avoid reprogramming points of H and therefore constitutes a valid EUF-NRA forgery. Hence, $\Pr[\mathsf{G}_3 : W_3] \leq \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}_{\mathrm{nra}}}(\lambda)$. Combining the bounds from the game hops proves the EUF-CRA bound.

sEUF-CRA extension. Games 0–3 now use sEUF-CRA freshness. The lines marked G_4–G_6 in Figure 7 add $\mathsf{CollCheck}$ and rejection rules for the forgery. For each signing answer and ring position, $\mathcal{T}$ stores (R, m, i, a_i, c_i, z_i), with $\mathsf{H}(i+1, R, m, a_i)$ programmed to the next challenge. If the forged signature is $\sigma^* = (c_0^*, z_0^*, \ldots, z_{n-1}^*)$, write $a_i^* = \mathsf{Rec}(pk_i^*, c_i^*, z_i^*)$ for the local commitments recomputed by verification. Because the programmed input contains the position index, any reuse of a signing-query oracle input by the forged chain must occur at the same ring position recorded in $\mathcal{T}$.

Game 4. Game 4 rejects a valid forgery if, for some ring position i, the table $\mathcal{T}$ contains a tuple $(R^*, m^*, i, \bar{a}_i, \bar{c}_i, \bar{z}_i)$ such that $a_i^* = \bar{a}_i$, $c_i^* = \bar{c}_i$, and $z_i^* \neq \bar{z}_i$.

The difference $|\Pr[\mathsf{G}_3 : W_3] - \Pr[\mathsf{G}_4 : W_4]|$ is at most the rejection probability, which we bound via a reduction to CUR. The CUR adversary guesses its honest-key index uniformly from $[N]$, embeds the CUR public key as the corresponding key in $\mathcal{R}$, generates the other honest keys itself, and simulates signing as in Game 3. It aborts if the embedded key is corrupted; this cannot happen on a correct guess because the selected key belongs to the forgery ring. On a correct guess, the recorded and forged responses share a commitment and challenge. Thus,

$$|\Pr[\mathsf{G}_3 : W_3] - \Pr[\mathsf{G}_4 : W_4]| \leq N\, \mathsf{Adv}^{\mathsf{CUR}}_{\mathsf{ID},\mathcal{A}_{\mathrm{cur}}}(\lambda).$$

Game 5. Game 5 rejects a valid forgery if, for some ring position i, the table $\mathcal{T}$ contains a tuple $(R^*, m^*, i, \bar{a}_i, \bar{c}_i, \bar{z}_i)$ such that $a_i^* = \bar{a}_i$, $c_i^* \neq \bar{c}_i$.

The difference $|\Pr[\mathsf{G}_4 : W_4] - \Pr[\mathsf{G}_5 : W_5]|$ is at most the rejection probability, which we bound via a reduction to CSS. The CSS reduction uses the same guessing-and-embedding strategy as the CUR reduction above. On a correct guess, it outputs $(\bar{a}_i, \bar{c}_i, \bar{z}_i, c_i^*, z_i^*)$ to the CSS game. Thus,

$$|\Pr[\mathsf{G}_4 : W_4] - \Pr[\mathsf{G}_5 : W_5]| \leq N\, \mathsf{Adv}^{\mathsf{CSS}}_{\mathsf{ID},\mathcal{A}_{\mathrm{css}}}(\lambda).$$

Game 6. Game 6 rejects a valid forgery if, for some ring position i, the table $\mathcal{T}$ contains a tuple $(R^*, m^*, i, \bar{a}_i, \bar{c}_i, \bar{z}_i)$ such that $a_i^* = \bar{a}_i$. After Games 4 and 5, such a reuse can only occur with $a_i^* = \bar{a}_i$, $c_i^* = \bar{c}_i$, and $z_i^* = \bar{z}_i$.

If the chains never split, the forgery repeats a returned signature, violating sEUF-CRA freshness. Otherwise, immediately before their first rejoining point, distinct commitments $a \neq a'$ produce the same next challenge: $\mathsf{H}(i+1, R^*, m^*, a) = \mathsf{H}(i+1, R^*, m^*, a')$. Thus, after Games 4 and 5, every Game 6 rejection gives a random-oracle collision. The finder evaluates the final chain's n inputs and retains at most nQ_{Sign} programmed signing pairs. Counting these classical evaluations together with the adversary's oracle queries, the collision finder makes at most $Q_{\mathsf{H}} + nQ_{\mathsf{Sign}} + n$ queries. Hence Lemma 3.3 gives

$$|\Pr[\mathsf{G}_5 : W_5] - \Pr[\mathsf{G}_6 : W_6]| \leq \frac{632(Q_{\mathsf{H}} + nQ_{\mathsf{Sign}} + n + 1)^3}{|\mathcal{C}|}.$$

sEUF-CRA conclusion. The EUF-NRA reduction simulates the keyless ring-signing oracle by reprogramming H and retaining the programmed entries throughout the adversary's execution. By Game 6, none of the oracle inputs used to verify the resulting forgery belongs to these signing-specific entries; hence, after the adversary terminates, the reduction can disable them without affecting verification. The forgery is therefore valid in the EUF-NRA experiment, yielding $\Pr[\mathsf{G}_6 : W_6] \leq \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}_{\mathrm{nra}}}(\lambda)$. Combining the bounds for Games 0–6 proves the claimed inequality. □

Remark 5.2 (Replacing ke-acHVZK with acHVZK). The use of ke-acHVZK in Theorem 5.1 is needed only to handle post-signing corruption. If the security experiment disallows every post-signing corruption, the same proof applies under ordinary acHVZK.

5.2 Reduction to Impersonation

The following lemma reduces EUF-NRA to impersonation with loss exponential in the number N of honest keys.

Lemma 5.1 (EUF-NRA of the FSwA ring transform from IMP [BM26, Theorem 8]). *Let $\mathsf{ID} = (\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ be an identification scheme with challenge space $\mathcal{C}$, and let $\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}]$ be the FSwA ring transform with random oracle H. In the EUF-NRA experiment with N honestly generated public keys, for any QPT adversary $\mathcal{A}$ making at most Q_{H} quantum queries to H, there exists a QPT impersonation adversary $\mathcal{A}_{\mathrm{imp}}$ against ID such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}}(\lambda) \leq \frac{1}{2}\big(2(Q_{\mathsf{H}} + N) + 1\big)^{2N}\, \mathsf{Adv}^{\mathsf{IMP}}_{\mathsf{ID},\mathcal{A}_{\mathrm{imp}}}(\lambda).$$

The running time of $\mathcal{A}_{\mathrm{imp}}$ is polynomial in the running time of $\mathcal{A}$.

Proof. We apply the EUF-NRA-to-IMP reduction of Beckmann and Majenz [BM26, Theorem 8]. Our formulation additionally provides a corruption oracle and uses rejection sampling.

Their reduction extends to corruption queries without additional loss. It embeds the IMP instance into one honestly generated public key and aborts if that key is corrupted. This abort cannot occur on the successful extraction branch: the embedded key belongs to the final forgery ring, whereas our winning predicate requires that the final ring be disjoint from the corruption set. All other corruption queries can be answered using the corresponding known secret keys. Hence, the adversary's view on the successful extraction branch is unchanged, and the corruption oracle causes no additional loss. □

Remark 5.3 (Exponential loss in the EUF-NRA reduction). The reduction in Lemma 5.1 is non-tight: its loss is exponential in the number N of honestly generated keys. Consequently, it yields the usual asymptotic security implication only when N is fixed. This loss belongs only to the generic IMP route of Lemma 5.1; it is not incurred by the EUF-NRA-to-EUF-CRA lifting of Theorem 5.1.

5.3 Anonymity

The following theorem gives ANON from acHVZK and ke-ANON from ke-acHVZK.

Theorem 5.2 (ANON and ke-ANON of the Fiat-Shamir-with-aborts ring transform). *Let $\mathsf{ID} = (\mathsf{Gen}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ be an identification scheme, and let $\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}]$ be the ring signature scheme obtained from the FSwA ring transform with random oracle H. For any QPT adversary $\mathcal{A}$ making at most Q_{Sign} queries to* RingSign *for rings of size at most n and at most Q_{H} quantum queries to H, let N be the number of honestly generated public keys in the security experiment. There exist QPT distinguishers $\mathcal{A}_{\mathrm{hvzk}}$ and $\mathcal{A}'_{\mathrm{hvzk}}$ such that, for any simulator Sim,*

$$\mathsf{Adv}^{\mathsf{ANON}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}}(\lambda) \leq N(Q_{\mathsf{Sign}} + 1)\, \mathsf{Adv}^{\mathsf{acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{A}_{\mathrm{hvzk}}}(\lambda) + \Gamma_{\mathrm{FSwA}}(Q_{\mathsf{Sign}} + 1),$$

$$\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{AOS}^{\mathsf{FSwA}}[\mathsf{ID}],\mathcal{A}}(\lambda) \le N(Q_{\mathsf{Sign}}+1)\,\mathsf{Adv}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID},\mathsf{Sim},\mathcal{A}'_{\mathrm{hvzk}}}(\lambda) + \Gamma_{\mathrm{FSwA}}(Q_{\mathsf{Sign}}+1).$$

Here, Γ_{FSwA} is defined in (2)*. Consequently, acHVZK implies ANON, whereas ke-acHVZK implies ke-ANON. The running times of $\mathcal{A}_{\mathrm{hvzk}}$ and $\mathcal{A}'_{\mathrm{hvzk}}$ are polynomial in the running time of $\mathcal{A}$ and include at most $O(nT(Q_{\mathsf{Sign}}+1))$ transcript simulations and oracle-programming operations.*

Proof. We replace the at most $Q_{\mathsf{Sign}}+1$ signing and challenge chains by public full-chain simulations, then replace each real signer transcript using acHVZK in the ANON experiment and ke-acHVZK in the ke-ANON experiment. The resulting challenge signature is independent of the challenge bit.

The full-chain replacement is identical to that in the unforgeability proof and costs $\Gamma_{\mathrm{FSwA}}(Q_{\mathsf{Sign}}+1)$. In the resulting games, each challenge c is sampled uniformly before replacing a real accepting transcript, generated by $(a, st) \leftarrow \mathsf{P}_1(sk)$ and $z \leftarrow \mathsf{P}_2(sk, st, c)$ conditioned on $z \neq \perp$, with $(a, z) \leftarrow \mathsf{Sim}(pk, c)$. These are precisely the two branches of acHVZK. Since the ANON adversary has no corruption oracle, ordinary acHVZK suffices in that experiment. For ke-ANON, we instead embed the ke-acHVZK experiment; its key-generation coins answer every CORR query, including post-signing and post-challenge corruptions. The hybrid over the signer keys and chains gives the stated $N(Q_{\mathsf{Sign}}+1)$ simulation term.

The final challenge signature uses the same public simulator for either signer, so its advantage is zero. Summing the hybrid losses proves the claim. □

6 Security Proof of the Ring Transform for Hash-and-Sign with Retry

This section extends Section 5 to HSwR. As above, the EUF-NRA-to-EUF-CRA/sEUF-CRA and anonymity theorems are the rejection-aware results, whereas the subsequent EUF-NRA-to-INV reduction is one replaceable way to establish the residual EUF-NRA term.

Standing assumptions and notation. Throughout this section, we assume the public target space is $\mathcal{C}$ and Samp satisfies universal sampling correctness as in Definition 2.7. Conditioned on any preceding execution, each honest inversion trial fails with probability at most $\varepsilon < 1$, where $(1-\varepsilon)^{-1}$ is polynomially bounded.

For rings of size at most n, at most Q_{H} quantum random-oracle queries, and Q signature chains, and for $\theta > 1$, define $T(Q)$ and $\tau(Q)$ as in Equation (1) and define the shared HSwR loss term by

$$\Gamma_{\mathrm{HSwR}}(Q) \coloneqq 3nT(Q)\sqrt{\frac{Q_{\mathsf{H}}}{|\mathcal{C}|}} + \frac{3nT(Q)Q_{\mathsf{H}}}{2|\mathcal{C}|} + \frac{7n^2(T(Q))^2}{|\mathcal{C}|} + 2\tau(Q). \tag{3}$$

The unforgeability statements use $Q = Q_{\mathsf{Sign}}$, whereas the anonymity statements use $Q = Q_{\mathsf{Sign}}+1$.

Remark 6.1 (Mitigating universal sampling correctness). As in the FSwA case discussed in Remark 5.1, universal sampling correctness is needed to handle adversarially generated public keys in EUF-CRA and sEUF-CRA signing queries. The full theorem requires sampler termination and validity for every well-formed adversarial key admitted by those queries. If these properties hold only for honestly generated keys, the security results hold only in the corresponding honest-key signing-ring restrictions.

6.1 Unforgeability

The following theorem reduces EUF-CRA to EUF-NRA and ke-PS, and sEUF-CRA additionally to SPR and INV.

Theorem 6.1 (EUF-NRA to EUF-CRA/sEUF-CRA of the hash-and-sign-with-retry ring transform). *Let $\mathsf{TDF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Inv})$ be a trapdoor function, and let $\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}]$ be the ring signature scheme obtained from the HSwR ring transform with random oracle H. Assume that $(\mathcal{C}, \circ)$ is a finite efficiently computable group. For any QPT adversary $\mathcal{A}$ making at most Q_{Sign} ring-signing queries for rings of size at*

most n and at most Q_{H} quantum queries to H, let N be the number of honestly generated public keys in the security experiment. There exist QPT adversaries $\mathcal{A}_{\mathrm{nra}}$ and $\mathcal{A}_{\mathrm{ps}}$ such that

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CRA}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}_{\mathrm{nra}}}(\lambda) + NQ_{\mathsf{Sign}}\,\mathsf{Adv}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}_{\mathrm{ps}}}(\lambda) + \Gamma_{\mathrm{HSwR}}(Q_{\mathsf{Sign}}).$$

For sEUF-CRA, there exist additional QPT adversaries $\mathcal{A}_{\mathrm{spr}}$ and $\mathcal{A}_{\mathrm{inv}}$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}}(\lambda) \le{}& \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}_{\mathrm{nra}}}(\lambda) + NQ_{\mathsf{Sign}}\,\mathsf{Adv}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}_{\mathrm{ps}}}(\lambda) + NQ_{\mathsf{Sign}}\,\mathsf{Adv}^{\mathsf{SPR}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}_{\mathrm{spr}}}(\lambda)\\ &+ NQ_{\mathsf{Sign}}\left(\big(2(Q_{\mathsf{H}}+n)+1\big)^2 + Q_{\mathsf{Sign}} - 1\right)\mathsf{Adv}^{\mathsf{INV}}_{\mathsf{TDF},\mathcal{A}_{\mathrm{inv}}}(\lambda)\\ &+ \frac{632(Q_{\mathsf{H}}+nQ_{\mathsf{Sign}}+n+1)^3}{|\mathcal{C}|} + \Gamma_{\mathrm{HSwR}}(Q_{\mathsf{Sign}}).\end{aligned}$$

In both bounds, Γ_{HSwR} is defined in (3)*. The running times of $\mathcal{A}_{\mathrm{nra}}$, $\mathcal{A}_{\mathrm{ps}}$, $\mathcal{A}_{\mathrm{spr}}$, and $\mathcal{A}_{\mathrm{inv}}$ are polynomial in the running time of $\mathcal{A}$ and include at most $O(nT(Q_{\mathsf{Sign}}))$ public simulations and oracle-programming operations.*

Proof sketch. We adapt Theorem 5.1 to HSwR. At a non-signer position, we sample $z_i \leftarrow \mathsf{Samp}(pk_i)$ and set $a_i := \mathsf{F}(pk_i, z_i) \circ c_i^{-1}$, which gives an accepting local equation with uniform a_i. Full-chain simulation and removal of failed-trial programming proceed as in Theorem 5.1. ke-PS replaces the successful signer inversion by Samp, contributing the stated ke-PS term, while reprogramming and abort cancellation contribute $\Gamma_{\mathrm{HSwR}}(Q_{\mathsf{Sign}})$. The resulting keyless signing oracle yields the EUF-NRA reduction.

For sEUF-CRA, SPR plays the role of CUR by ruling out two openings of the same target. However, the trapdoor-function setting has no natural CSS analogue, so we reduce the different-challenge case to INV via the measure-and-reprogramming technique shown in Lemma 3.2. Any remaining overlap between the simulated and forged cycles yields a random-oracle collision. Outside these bad events, the terminal EUF-NRA argument of Theorem 5.1 applies unchanged.

Proof. The proof follows Theorem 5.1, replacing its transcript simulator with the HSwR public simulation above. Let W_i be the event that $\mathcal{A}$ outputs a valid fresh forgery in Game i, denoted by G_i, and write $Q'_{\mathsf{Sign}} = T(Q_{\mathsf{Sign}})$. We truncate the combined retry loops after this many trials; the term $2\tau(Q_{\mathsf{Sign}})$ in Γ_{HSwR} covers the cutoff transitions.

Games 0–2. In Game 1, each signing attempt first samples all full-chain challenges $c_1, \ldots, c_n$, and then programs the corresponding oracle points in chain order. At the signer position, a_j is uniform in $\mathcal{C}$. At every non-signer position, $a_i = \mathsf{F}(pk_i, z_i) \circ c_i^{-1}$ is also uniform, since, for every fixed pk_i, z_i, the map $c \mapsto \mathsf{F}(pk_i, z_i) \circ c^{-1}$ is a permutation of $\mathcal{C}$. Hence, every programmed point has conditional guessing probability at most $1/|\mathcal{C}|$, and Lemma 3.1 gives the bound.

Game 2 removes the programming for each unsuccessful attempt, starting from the end of the chain. When the programming at position i is removed, the earlier part of the chain is therefore still unchanged, and the later part is no longer relevant. Since the point at position i is randomized by the independently sampled c_i, Lemma 3.1 applies at each step. Combining this with the FSwA abort-cancellation argument and the Game 0-to-Game 1 bound gives

$$|\Pr[\mathsf{G}_0 : W_0] - \Pr[\mathsf{G}_2 : W_2]| \le \Gamma_{\mathrm{HSwR}}(Q_{\mathsf{Sign}}).$$

Game 3. For the signer-component replacement, condition real signing on success: $a_j \leftarrow \mathcal{C}$, $c_j \leftarrow \mathcal{C}$, and $z_j \leftarrow \mathsf{Inv}(sk_s, a_j \circ c_j)$. The target $a_j \circ c_j$ is uniform, so z_j has the successful preimage distribution. A hybrid over signing answers and honest signer keys embeds ke-PS in one component: it receives $z^\star$, sets $a_j = \mathsf{F}(R[j], z^\star) \circ c_j^{-1}$, and uses this as the signer component. A real ke-PS sample gives Game 2; a $z^\star \leftarrow \mathsf{Samp}(R[j])$ sample gives Game 3. Summing the hybrids,

$$|\Pr[\mathsf{G}_2 : W_2] - \Pr[\mathsf{G}_3 : W_3]| \le NQ_{\mathsf{Sign}}\,\mathsf{Adv}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}_{\mathrm{ps}}}(\lambda).$$

Game 4. Game 4 rejects if, for some position i, the table $\mathcal{T}$ contains a tuple $(R^*, m^*, i, pk_i^*, \bar{c}_i, \bar{z}_i, \bar{a}_i)$ such that $a_i^* = \bar{a}_i$, $c_i^* = \bar{c}_i$, and $z_i^* \neq \bar{z}_i$. The recorded and forged equations then share a public key, challenge, and hash input, while $z_i^* \neq \bar{z}_i$ and $\mathsf{F}(pk_i^*, z_i^*) = \mathsf{F}(pk_i^*, \bar{z}_i)$ violate SPR. The reduction guesses the signing query and honest-key index; its position is then fixed by the ordered ring. It uses the SPR challenge preimage for that local component and simulates the rest as in Game 3. Thus,

$$|\Pr[\mathsf{G}_3 : W_3] - \Pr[\mathsf{G}_4 : W_4]| \leq NQ_{\mathsf{Sign}}\, \mathsf{Adv}^{\mathsf{SPR}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}_{\mathrm{spr}}}(\lambda).$$

Game 5. Game 5 rejects if, for some ring position i, the table $\mathcal{T}$ contains a tuple $(R^*, m^*, i, pk_i^*, \bar{c}_i, \bar{z}_i, \bar{a}_i)$ such that $a_i^* = \bar{a}_i$ and $c_i^* \neq \bar{c}_i$. For the first tuple satisfying this condition, consider the predecessor input used to compute c_i^*. If it is not in the signing-programming table, measure-and-reprogramming embeds a uniform INV target at that input and incurs the factor $NQ_{\mathsf{Sign}}(2(Q_{\mathsf{H}}+n)+1)^2$. If it is in the table, the entry must come from a different signing answer; a direct embedding then incurs the factor $NQ_{\mathsf{Sign}}(Q_{\mathsf{Sign}}-1)$. The detailed reduction is given in Lemma F.2. Combining the two cases gives

$$|\Pr[\mathsf{G}_4 : W_4] - \Pr[\mathsf{G}_5 : W_5]| \leq NQ_{\mathsf{Sign}} \left(\big(2(Q_{\mathsf{H}}+n)+1\big)^2 + Q_{\mathsf{Sign}} - 1 \right) \mathsf{Adv}^{\mathsf{INV}}_{\mathsf{TDF},\mathcal{A}_{\mathrm{inv}}}(\lambda).$$

Game 6 and conclusion. The transition to Game 6 and the concluding argument are identical to those in the proof of Theorem 5.1. In particular, Lemma 3.3 bounds this hop by

$$|\Pr[\mathsf{G}_5 : W_5] - \Pr[\mathsf{G}_6 : W_6]| \leq \frac{632(Q_{\mathsf{H}} + nQ_{\mathsf{Sign}} + n + 1)^3}{|\mathcal{C}|}.$$

As in the proof of Theorem 5.1, every winning forgery in Game 6 yields a valid EUF-NRA forgery. Consequently,

$$\Pr[\mathsf{G}_6 : W_6] \leq \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}_{\mathrm{nra}}}(\lambda).$$

Summing Games 0-6, together with the cutoff loss already included in Γ_{HSwR}, proves the theorem. □

6.2 Reduction to Non-invertibility

The following corollary reduces EUF-NRA to INV with the same exponential-in-N loss as Lemma 5.1.

Corollary 6.1 (EUF-NRA of the HSwR ring transform from INV). *Let* $\mathsf{TDF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Inv})$ *be a trapdoor function whose public target space is* $\mathcal{C}$*, and let* $\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}]$ *be the HSwR ring transform with random oracle* H*. Assume that* $(\mathcal{C}, \circ)$ *is a finite efficiently computable group. In the EUF-NRA experiment with* N *honestly generated public keys, for any QPT adversary* $\mathcal{A}$ *making at most* Q_{H} *quantum queries to* H*, there exists a QPT adversary* $\mathcal{A}_{\mathrm{inv}}$ *such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}}(\lambda) \leq \frac{1}{2}\big(2(Q_{\mathsf{H}}+N)+1\big)^{2N}\, \mathsf{Adv}^{\mathsf{INV}}_{\mathsf{TDF},\mathcal{A}_{\mathrm{inv}}}(\lambda).$$

The running time of $\mathcal{A}_{\mathrm{inv}}$ *is polynomial in the running time of* $\mathcal{A}$*.*

Proof. We compose the EUF-NRA-to-IMP reduction of Beckmann and Majenz [BM26, Theorem 8] for the local relation $\mathsf{V}(pk, a, c, z) = 1 \Leftrightarrow \mathsf{F}(pk, z) = a \circ c$ with a tight IMP-to-INV reduction. The first step gives the displayed exponential loss and is compatible with corruption for the same reason as Lemma 5.1; the second maps a uniform INV target to a uniform local challenge using the invertible action. For the second step, let $\mathcal{A}_{\mathrm{imp}} = (\mathcal{A}_1, \mathcal{A}_2)$ be the impersonation adversary. Given an INV challenge (pk, y) with $y \leftarrow \mathcal{C}$, the reduction runs $(a, st) \leftarrow \mathcal{A}_1(pk)$, computes $c = a^{-1} \circ y$, and gives c to $\mathcal{A}_2$. By assumption, c is uniform in $\mathcal{C}$, so this is distributed as the verifier challenge in the local IMP experiment. If $\mathcal{A}_2$ returns z such that $\mathsf{F}(pk, z) = a \circ c$, then z is a valid preimage of the INV target. Combining both reductions gives the bound. □

Remark 6.2 (Non-tightness of the EUF-NRA reduction). The same qualification as in Remark 5.3 applies to Corollary 6.1: the reduction is non-tight, with a loss exponential in the number N of honestly generated keys. The loss is confined to this generic EUF-NRA-to-INV route and disappears when a direct or tighter EUF-NRA proof is supplied.

6.3 Anonymity

The following theorem gives ANON from PS and ke-ANON from ke-PS.

Theorem 6.2 (ANON and ke-ANON of the hash-and-sign-with-retry ring transform). *Let* $\mathsf{TDF} = (\mathsf{Gen}, \mathsf{F}, \mathsf{Inv})$ *be a trapdoor function, and let* $\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}]$ *be the ring signature scheme obtained from the HSwR ring transform with random oracle* H*. Assume that* $(\mathcal{C}, \circ)$ *is a finite efficiently computable group. For any QPT adversary* $\mathcal{A}$ *making at most* Q_{Sign} *queries to* RingSign *for rings of size at most* n *and at most* Q_{H} *quantum queries to* H*, let* N *be the number of honestly generated public keys in the security experiment. There exist QPT distinguishers* $\mathcal{A}_{\mathrm{ps}}$ *and* $\mathcal{A}'_{\mathrm{ps}}$ *such that*

$$\mathsf{Adv}^{\mathsf{ANON}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}}(\lambda) \le N(Q_{\mathsf{Sign}}+1)\,\mathsf{Adv}^{\mathsf{PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}_{\mathrm{ps}}}(\lambda) + \Gamma_{\mathrm{HSwR}}(Q_{\mathsf{Sign}}+1),$$
$$\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{AOS}^{\mathsf{HSwR}}[\mathsf{TDF}],\mathcal{A}}(\lambda) \le N(Q_{\mathsf{Sign}}+1)\,\mathsf{Adv}^{\mathsf{ke\text{-}PS}}_{\mathsf{TDF},\mathsf{Samp},\mathcal{A}'_{\mathrm{ps}}}(\lambda) + \Gamma_{\mathrm{HSwR}}(Q_{\mathsf{Sign}}+1).$$

Here Γ_{HSwR} *is defined in* (3)*. Consequently, PS implies ANON, whereas ke-PS implies ke-ANON. The running times of* $\mathcal{A}_{\mathrm{ps}}$ *and* $\mathcal{A}'_{\mathrm{ps}}$ *are polynomial in the running time of* $\mathcal{A}$ *and include at most* $O(nT(Q_{\mathsf{Sign}}+1))$ *public simulations and oracle-programming operations.*

Proof. Apply the full-chain replacement from Theorem 6.1 to the signing and challenge chains, at cost $\Gamma_{\mathrm{HSwR}}(Q_{\mathsf{Sign}}+1)$. A hybrid over these chains replaces each successful inversion output by Samp, using PS for ANON and ke-PS for ke-ANON; in the latter case, the exposed key-generation coins answer the corruption queries. The resulting challenge signature is independent of the signer, and summing the hybrid losses gives the claimed bound. □

7 Case Study of ML-DSA

We clarify the applicability of the FSwA ring transform to ML-DSA [Nat24] and its current limitations through a case study of a scheme we call Ring-ML-DSA. We specify the construction, in which public simulation is enabled by augmenting the public key with $\boldsymbol{t}_0$, reduce it to a simplified variant, and apply the generic FSwA security theorem to make explicit the assumptions required for its security proof. Appendix D further relates these requirements to concrete module-lattice problems and evaluates the resulting parameters. However, publishing $\boldsymbol{t}_0$ changes the standard ML-DSA public-key relation, and its security implications, the required adversarial-key conditions, and a tight EUF-NRA reduction remain open.

7.1 Specification

Ring-ML-DSA runs the key-generation of ML-DSA but publishes an augmented ring public key $pk' := \mathsf{PkEncode}'(pk, \boldsymbol{t}_0)$, where pk is the standard public key and $\boldsymbol{t}_0$ is the low-order component kept secret in ordinary ML-DSA. The generic simulator for non-signing ring positions needs this component as public input; the theorems below are therefore stated for the augmented-public-key identification protocol. In addition, the ring interface admits an augmented key only when the matrix expanded from seed ρ satisfies the parameter-set-dependent rank condition. This prevents an adversary from inserting a sufficiently rank-deficient key that would invalidate the commitment min-entropy.

We also define simplified Ring-ML-DSA (sRing-ML-DSA), which replaces the pseudorandom generation of $\boldsymbol{y}_j$ for the signer component and $\boldsymbol{z}_i$ for the non-signer components with fresh sampling. This simplification is needed because we cannot apply Section 5 to Ring-ML-DSA directly. Appendix B gives the detailed specifications of Ring-ML-DSA and its sRing-ML-DSA.

Remark 7.1 (Augmented public keys). ML-DSA omits $\boldsymbol{t}_0$ from the public key and retains it in the secret-key encoding to reduce size [Nat24], although the underlying relation is $\boldsymbol{t} = 2^d\boldsymbol{t}_1 + \boldsymbol{t}_0 = \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2$, where $\boldsymbol{A} = \mathsf{ExpandA}(\rho)$. Ring-ML-DSA instead publishes $pk' = (pk, \boldsymbol{t}_0)$ to enable public simulation at non-signing positions; the verifier can then reconstruct $\boldsymbol{t}$ and recover the commitment as $\mathsf{HighBits}(\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t})$ without a hint $\boldsymbol{h}$. Our results concern only this augmented-key relation and do not establish equivalence with standardized ML-DSA or preservation of all ML-DSA hardness assumptions; uncompressing the public key may also affect the equations available to side-channel attacks [OVCG25], whose concrete analysis is outside our scope.

7.2 Reduction from Original ML-DSA to Simplified Ring-ML-DSA

We adapt the fault-free randomization hops of Kosuge and Xagawa [KX25, Theorem 2] to the augmented-key algorithms above. We give the adapted proof explicitly in Appendix F.3.

Corollary 7.1 (Original-to-simplified Ring-ML-DSA reduction in the QROM). *For $Q \geq 0$, define*

$$\Lambda_{\mathrm{rnd}}(Q) \coloneqq \frac{Q(Q-1)}{2^{512}} + \frac{Q(Q-1)}{2^{256}} + \frac{\sqrt{Q_{\mathsf{H}}(Q_{\mathsf{H}}+1)Q}}{2^{127}} + \frac{\sqrt{Q_{\mathsf{H}}(Q_{\mathsf{H}}+1)Q}}{2^{255}}.$$

For any QPT adversary $\mathcal{A}$ making at most Q_{Sign} ring-signing queries, suppose that every signing execution samples nonce rnd independently and uniformly from $\{0,1\}^{256}$ and that the complete experiment makes at most Q_{H} reads to the single oracle H across all call sites. Then, there is a QPT adversary $\mathcal{A}'$ for the corresponding experiment such that

$$\mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{Ring\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}'}(\lambda) + \Lambda_{\mathrm{rnd}}(Q_{\mathsf{Sign}}),$$
$$\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{Ring\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}'}(\lambda) + \Lambda_{\mathrm{rnd}}(Q_{\mathsf{Sign}}+1).$$

The running time of $\mathcal{A}'$ is polynomial in the running time of $\mathcal{A}$ and in the cost of simulating at most $Q_{\mathsf{Sign}}+1$ signatures and answering at most Q_{H} random-oracle queries.

Remark 7.2 (Fresh signing randomness under key exposure). In Ring-ML-DSA, the j-th signer holds the signing-key component $K_j \in \{0,1\}^{256}$ used for nonce derivation. For each signing, it independently samples $rnd \leftarrow \{0,1\}^{256}$ and derives the 64-byte masking seed $\rho'' = \mathsf{H}(K_j \| rnd \| \mu, 64)$, where μ is the message representative. The response masks are generated from ρ'' using $\mathsf{ExpandMask}$, whereas sRing-ML-DSA replaces them by fresh independent samples.

The reduction in Corollary 7.1 remains valid if K_j is exposed before or after signer j produces the signature, including when the key-generation coins are exposed in the ke-ANON experiment. For every signing execution, *rnd* is sampled independently and uniformly from $\{0,1\}^{256}$. Thus, even conditioned on the preceding execution history and on known K_j and μ, the hidden value *rnd* supplies 256 bits of entropy to the oracle input (K_j, rnd, μ). Collisions among the internal nonces used in at most Q signing executions are covered by the $Q(Q-1)/2^{256}$ term in $\Lambda_{\mathrm{rnd}}(Q)$. If *rnd* is not assumed to contain fresh hidden entropy, the unpredictability of ρ'' must instead rely on the secrecy of K_j. Under that weaker randomness assumption, the reduction applies only to the restricted notions that do not expose K_j.

7.3 Conditional Security of Simplified Ring-ML-DSA

The argument specializes the generic FSwA transform. As in the generic theorem, n bounds the ring size and N is the number of honestly generated keys in the security experiment. The parameters $n, N, Q_{\mathsf{H}}, \alpha, \alpha', \delta, \varepsilon_{\mathrm{ML}}$, and θ are inherited from that theorem, and the argument Q only counts the number of simulated ring signatures. The bound Q_{H} counts all queries to H. Set $T_{\mathrm{ML}}(Q) \coloneqq \lceil \theta Q/(1-\varepsilon_{\mathrm{ML}}) \rceil$ and $\tau_{\mathrm{ML}}(0) \coloneqq 0$, while for $Q > 0$ set $\tau_{\mathrm{ML}}(Q) \coloneqq \exp(-(\theta-1)^2 Q/(2\theta))$. Define the ML-DSA specialization of the reprogramming and abort-cancellation loss by

$$\begin{aligned} \Gamma_{\mathrm{ML}}(Q) \coloneqq{}& 3T_{\mathrm{ML}}(Q)\sqrt{Q_{\mathsf{H}}2^{-\alpha}} + 3(n-1)T_{\mathrm{ML}}(Q)\sqrt{Q_{\mathsf{H}}2^{-\alpha'}} + 7n(T_{\mathrm{ML}}(Q))^2 2^{-\alpha} + 3T_{\mathrm{ML}}(Q)Q_{\mathsf{H}}2^{-\alpha-1} \\ &+ 7n(n-1)(T_{\mathrm{ML}}(Q))^2 2^{-\alpha'} + 3(n-1)T_{\mathrm{ML}}(Q)Q_{\mathsf{H}}2^{-\alpha'-1} + 2N\delta + 2\tau_{\mathrm{ML}}(Q). \end{aligned} \tag{4}$$

In addition, the concrete chain shares the same random function H across all call sites. The proof of the corollary below accounts for this difference from the generic syntax by the additive context-binding term

$$\Delta_{\mathrm{bind}} \coloneqq \frac{632(Q_{\mathsf{H}}+1)^3}{2^{512}}. \tag{5}$$

We can apply Theorems 5.1 and 5.2 using the simulator properties established in Appendix C, including universal simulation correctness and $(1-\varepsilon_{\mathrm{ML}})^{-1} = \mathsf{poly}(\lambda)$ under the heuristic assumption that $\mathsf{LowBits}(\boldsymbol{A}\boldsymbol{z} -$

Table 2. HSwR instantiation notes.

Candidate	Public map $\mathsf{F}(pk, z)$	Group	(ke-)PS	SPR
Falcon [PFH+22]	NTRU map on a short pair: $pk = h$, $z = (s_1, s_2) \in \mathcal{R}_q^2$, and $\mathsf{F}(pk, z) = s_1 + s_2 h$, where $\mathcal{R}_q = \mathbb{Z}_q[x]/(\phi)$.	$(\mathcal{R}_q, +)$	ke-PS with Rényi-divergence loss [PFH+22, Sec. 2.6]	Not established under standard assumptions of Falcon [GJK24a]
salt-UOV [BCD+24]	Public system of quadratic polynomials: $pk = P : \mathbb{F}_q^n \to \mathbb{F}_q^m$ or seed to generate P, $z = \boldsymbol{s} \in \mathbb{F}_q^n$, and $\mathsf{F}(pk, z) = P(\boldsymbol{s})$.	$(\mathbb{F}_q^m, +)$	Perfect ke-PS [KX24, Prop. 5.3]	Not established
QR-UOV [AFI+24]	Public system of quadratic polynomials with quotient-ring structure: pk is seed to generate $P : \mathbb{F}_q^n \to \mathbb{F}_q^m$, $z = \boldsymbol{s} \in \mathbb{F}_q^n$, and $\mathsf{F}(pk, z) = P(\boldsymbol{s})$.	$(\mathbb{F}_q^m, +)$	Perfect ke-PS [KX24, Prop. 5.3]	Not established
MAYO [BCC+24]	Whipped quadratic polynomials: pk is seed to generate $P^* : (\mathbb{F}_{16}^n)^k \to \mathbb{F}_{16}^m$, $z = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k) \in (\mathbb{F}_{16}^n)^k$, and $\mathsf{F}(pk, z) = P^*(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k)$.	$(\mathbb{F}_{16}^m, +)$	Perfect ke-PS, conditioned on full rank [KX24, Prop. 5.5]	Not established

$c\boldsymbol{t}$) is uniformly distributed,[5] the commitment min-entropy parameters (α, δ) and α', and perfect ke-acHVZK. See Appendix F.4 for the proofs.

Corollary 7.2 (Conditional security of simplified Ring-ML-DSA in the QROM). *Let* $\mathsf{ID}_{\mathrm{ML}}$ *be the ML-DSA's identification protocol underlying the above construction, where the ring public key is* $pk' = \mathsf{PkEncode}'(pk, \boldsymbol{t}_0)$*, and let* $\mathsf{Sim}_{\mathrm{ML}}$ *be its perfect ke-acHVZK simulator. For any QPT adversary* $\mathcal{A}$ *against* sRing-ML-DSA *making at most* Q_{Sign} *ring-signing queries for rings of size at most* n*, let* N *be the number of honestly generated keys in the security experiment, and suppose that the complete experiment makes at most* Q_{H} *reads to the single oracle* H *across all call sites. There exists a QPT adversary* $\mathcal{A}_{\mathrm{nra}}$ *such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}_{\mathrm{nra}}}(\lambda) + \Gamma_{\mathrm{ML}}(Q_{\mathsf{Sign}}) + \Delta_{\mathrm{bind}}.$$

Here Γ_{ML} *is defined in* (4) *and* Δ_{bind} *in* (5)*. If, in addition,* $\mathsf{ID}_{\mathrm{ML}}$ *satisfies CUR and CSS, then there exist QPT adversaries* $\mathcal{A}_{\mathrm{cur}}$ *and* $\mathcal{A}_{\mathrm{css}}$ *such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \le{}& \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}_{\mathrm{nra}}}(\lambda) + N\,\mathsf{Adv}^{\mathsf{CUR}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{cur}}}(\lambda) + N\,\mathsf{Adv}^{\mathsf{CSS}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{css}}}(\lambda)\\ &+ \frac{632}{|\mathcal{C}|}(Q_{\mathsf{H}} + nQ_{\mathsf{Sign}} + n + 1)^3 + \Gamma_{\mathrm{ML}}(Q_{\mathsf{Sign}}) + \Delta_{\mathrm{bind}}.\end{aligned}$$

Moreover,

$$\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \le \Gamma_{\mathrm{ML}}(Q_{\mathsf{Sign}} + 1) + \Delta_{\mathrm{bind}}.$$

The running times of $\mathcal{A}_{\mathrm{nra}}$*,* $\mathcal{A}_{\mathrm{cur}}$*, and* $\mathcal{A}_{\mathrm{css}}$ *are polynomial in the running time of* $\mathcal{A}$ *and include at most* $O(nT(Q_{\mathsf{Sign}} + 1))$ *transcript simulations and oracle-programming operations.*

8 Case Study of Hash-and-Sign Ring Signatures

This section examines the applicability of the HSwR ring transform to salt-UOV, QR-UOV, Falcon, and MAYO and identifies the conditions that remain to be established for complete instantiations. Table 2 summarizes

[5] The same heuristic assumption is used for honestly generated keys in [KLS18, BBD+23]. Here, we require the stronger version that also applies to adversarially generated keys.

the properties relevant to our security reductions. Universal sampling is immediate from the key-independent domain samplers, while INV is the standard one-wayness requirement for the underlying trapdoor functions. The table therefore focuses on (ke-)PS and the nontrivial SPR status; a complete instantiation additionally requires a concrete EUF-NRA analysis. Salt-UOV [BCD+24][6] and QR-UOV [AFI+24] fit the public-salt retry interface and admit perfect ke-PS. Falcon's public NTRU map still requires a QROM treatment of its Rényi-divergence-based public-Gaussian replacement and SPR. MAYO simulation is conditioned on full rank, making applicability depend on its failure probability and the number of signing samples. Appendix E gives details.

Acknowledgments

A generative-AI tool (GPT 5.6) was used solely for language editing and presentation. All technical content was produced and verified by the authors.

References

ABB+13. Carlos Aguilar-Melchor, Slim Bettaieb, Xavier Boyen, Laurent Fousse, and Philippe Gaborit. Adapting Lyubashevsky's signature schemes to the ring signature setting. In Amr Youssef, Abderrahmane Nitaj, and Aboul Ella Hassanien, editors, *AFRICACRYPT 13*, volume 7918 of *LNCS*, pages 1–25. Springer, Berlin, Heidelberg, June 2013.

AFI+24. Rika Akiyama, Hiroki Furue, Yasuhiko Ikematsu, Fumitaka Hoshino, Koha Kinjo, Haruhisa Kosuge, Satoshi Nakamura, Shingo Orihara, Tsuyoshi Takagi, and Kimihiro Yamakoshi. QR-UOV. Technical report, National Institute of Standards and Technology, 2024. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures`.

AHU19. Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 269–295. Springer, Cham, August 2019.

AOS02. Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n signatures from a variety of keys. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 415–432. Springer, Berlin, Heidelberg, December 2002.

BBD+23. Manuel Barbosa, Gilles Barthe, Christian Doczkal, Jelle Don, Serge Fehr, Benjamin Grégoire, Yu-Hsuan Huang, Andreas Hülsing, Yi Lee, and Xiaodi Wu. Fixing and mechanizing the security proof of Fiat-Shamir with aborts and Dilithium. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 358–389. Springer, Cham, August 2023.

BCC+24. Ward Beullens, Fabio Campos, Sofía Celi, Basil Hess, and Matthias J. Kannwischer. MAYO. Technical report, National Institute of Standards and Technology, 2024. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures`.

BCD+24. Ward Beullens, Ming-Shing Chen, Jintai Ding, Boru Gong, Matthias J. Kannwischer, Jacques Patarin, Bo-Yuan Peng, Dieter Schmidt, Cheng-Jhih Shih, Chengdong Tao, and Bo-Yin Yang. UOV — Unbalanced Oil and Vinegar. Technical report, National Institute of Standards and Technology, 2024. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures`.

BK10. Zvika Brakerski and Yael Tauman Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. Cryptology ePrint Archive, Report 2010/086, 2010.

BKM06. Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 60–79. Springer, Berlin, Heidelberg, March 2006.

BKP20. Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 464–492. Springer, Cham, December 2020.

BLSS22. Maxime Buser, Joseph K. Liu, Ron Steinfeld, and Amin Sakzad. Post-quantum ID-based ring signatures from symmetric-key primitives. In Giuseppe Ateniese and Daniele Venturi, editors, *ACNS 2022*, volume 13269 of *LNCS*, pages 892–912. Springer, Cham, June 2022.

[6] The UOV specification presents salt-UOV as an alternative; we consider it because its public-salt retry structure matches HSwR and supports our simulator.

BM26. Marvin Beckmann and Christian Majenz. Quantum oracle distribution switching and its applications to fully anonymous ring signatures. Cryptology ePrint Archive, Paper 2026/293; arXiv:2602.16268, 2026.

CGH+21. Rohit Chatterjee, Sanjam Garg, Mohammad Hajiabadi, Dakshita Khurana, Xiao Liang, Giulio Malavolta, Omkant Pandey, and Sina Shiehian. Compact ring signatures from learning with errors. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 282–312, Virtual Event, August 2021. Springer, Cham.

CHH+21. Kai-Min Chung, Yao-Ching Hsieh, Mi-Ying Huang, Yu-Hsuan Huang, Tanja Lange, and Bo-Yin Yang. Group signatures and accountable ring signatures from isogeny-based assumptions. Cryptology ePrint Archive, Report 2021/1368, 2021.

Das20. Dipayan Das. Fiat-shamir with aborts: From identification schemes to linkable ring signatures. In Lejla Batina, Stjepan Picek, and Mainack Mondal, editors, *Security, Privacy, and Applied Cryptography Engineering — 10th International Conference, SPACE 2020, Kolkata, India, December 17–21, 2020, Proceedings*, volume 12586 of *Lecture Notes in Computer Science*, pages 167–187, Cham, 2020. Springer.

DAZ19. Dipayan Das, Man Ho Au, and Zhenfei Zhang. Ring signatures based on middle-product learning with errors problems. In Johannes Buchmann, Abderrahmane Nitaj, and Tajje eddine Rachidi, editors, *AFRICACRYPT 19*, volume 11627 of *LNCS*, pages 139–156. Springer, Cham, July 2019.

DFM20. Jelle Don, Serge Fehr, and Christian Majenz. The measure-and-reprogram technique 2.0: Multi-round Fiat-Shamir and more. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 602–631. Springer, Cham, August 2020.

DFPS23. Julien Devevey, Pouria Fallahpour, Alain Passelègue, and Damien Stehlé. A detailed analysis of Fiat-Shamir with aborts. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 327–357. Springer, Cham, August 2023.

DRS17. David Derler, Sebastian Ramacher, and Daniel Slamanig. Post-quantum zero-knowledge proofs for accumulators with applications to ring signatures from symmetric-key primitives. Cryptology ePrint Archive, Report 2017/1154, 2017.

ESS+19. Muhammed F. Esgin, Ron Steinfeld, Amin Sakzad, Joseph K. Liu, and Dongxi Liu. Short lattice-based one-out-of-many proofs and applications to ring signatures. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 2019*, volume 11464 of *LNCS*, pages 67–88. Springer, Cham, June 2019.

EZS+19. Muhammed F. Esgin, Raymond K. Zhao, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. MatRiCT: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 567–584. ACM Press, November 2019.

FFH25. Pouria Fallahpour, Serge Fehr, and Yu-Hsuan Huang. Tighter quantum security for Fiat-Shamir-with-aborts and hash-and-sign-with-retry signatures. Cryptology ePrint Archive, Report 2025/985, 2025.

FR25. Thibauld Feneuil and Matthieu Rivain. Threshold computation in the head: Improved framework for post-quantum signatures and zero-knowledge arguments. *Journal of Cryptology*, 38(3):28, July 2025.

GHHM21. Alex B. Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight adaptive reprogramming in the QROM. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 637–667. Springer, Cham, December 2021.

GJK24a. Phillip Gajland, Jonas Janneck, and Eike Kiltz. A closer look at falcon. Cryptology ePrint Archive, Report 2024/1769, 2024.

GJK24b. Phillip Gajland, Jonas Janneck, and Eike Kiltz. Ring signatures for deniable AKEM: Gandalf's fellowship. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part I*, volume 14920 of *LNCS*, pages 305–338. Springer, Cham, August 2024.

HY18. Akinori Hosoyamada and Kan Yasuda. Building quantum-one-way functions from block ciphers: Davies-Meyer and Merkle-Damgård constructions. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 275–304. Springer, Cham, December 2018.

JMW24. Kelsey A. Jackson, Carl A. Miller, and Daochen Wang. Evaluating the security of CRYSTALS-Dilithium in the quantum random oracle model. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part VI*, volume 14656 of *LNCS*, pages 418–446. Springer, Cham, May 2024.

KKW18. Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 525–537. ACM Press, October 2018.

KLS18. Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 552–586. Springer, Cham, April / May 2018.

KX24. Haruhisa Kosuge and Keita Xagawa. Probabilistic hash-and-sign with retry in the quantum random oracle model. In Qiang Tang and Vanessa Teague, editors, *PKC 2024, Part I*, volume 14601 of *LNCS*, pages 259–288. Springer, Cham, April 2024.

KX25. Haruhisa Kosuge and Keita Xagawa. The security of ML-DSA against fault-injection attacks. In Goichiro Hanaoka and Bo-Yin Yang, editors, *ASIACRYPT 2025, Part II*, volume 16246 of *LNCS*, pages 641–674. Springer, Singapore, December 2025.

LAZ19a. Xingye Lu, Man Ho Au, and Zhenfei Zhang. (Linkable) ring signature from hash-then-one-way signature. Cryptology ePrint Archive, Report 2019/567, 2019.

LAZ19b. Xingye Lu, Man Ho Au, and Zhenfei Zhang. Raptor: A practical lattice-based (linkable) ring signature. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 2019*, volume 11464 of *LNCS*, pages 110–130. Springer, Cham, June 2019.

LDK+20. Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2020. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions`.

LLNW16. Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 1–31. Springer, Berlin, Heidelberg, May 2016.

LNS21. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. SMILE: Set membership from ideal lattices with applications to ring signatures and confidential transactions. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 611–640, Virtual Event, August 2021. Springer, Cham.

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, Berlin, Heidelberg, December 2009.

MP17. Mohamed Saied Emam Mohamed and Albrecht Petzoldt. RingRainbow - an efficient multivariate ring signature scheme. In Marc Joye and Abderrahmane Nitaj, editors, *AFRICACRYPT 17*, volume 10239 of *LNCS*, pages 3–20. Springer, Cham, May 2017.

Nat24. National Institute of Standards and Technology. FIPS 204, module-lattice-based digital signature standard. Standard, U.S. Department of Commerce, Washington, D.C., August 2024.

OVCG25. Paco Azevedo Oliveira, Andersson Calle Viera, Benoît Cogliati, and Louis Goubin. Uncompressing dilithium's public key. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part I*, volume 16000 of *LNCS*, pages 417–443. Springer, Cham, August 2025.

PFH+22. Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at `https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022`.

RST01. Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer, Berlin, Heidelberg, December 2001.

SSH11. Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. On provable security of UOV and HFE signature schemes against chosen-message attack. In Bo-Yin Yang, editor, *PQCrypto 2011*, pages 68–82. Springer, Berlin, Heidelberg, November / December 2011.

YEL+21. Tsz Hon Yuen, Muhammed F. Esgin, Joseph K. Liu, Man Ho Au, and Zhimin Ding. DualRing: Generic construction of ring signatures with efficient instantiations. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 251–281, Virtual Event, August 2021. Springer, Cham.

Zha12. Mark Zhandry. How to construct quantum random functions. In *53rd FOCS*, pages 679–687. IEEE Computer Society Press, October 2012.

Zha15. Mark Zhandry. A note on the quantum collision and set equality problems. *Quantum Info. Comput.*, 15(7–8):557–567, May 2015.

A Algorithmic Templates

A.1 Figures of FSwA and HSwR

Figures 8 and 9 provide generic templates for the signature-generation and verification algorithms of FSwA and HSwR.

$\mathsf{AOS}^{\mathsf{FS}}[\mathsf{ID}].\mathsf{Sign}(R, m, sk_j)$	$\mathsf{AOS}^{\mathsf{FS}}[\mathsf{ID}].\mathsf{Verify}(R, m, \sigma)$
1 : $(a_j, st_j) \leftarrow \mathsf{P}_1(sk_j)$	1 : $(c_0, z_0, \ldots, z_{n-1}) \coloneqq \mathsf{Parse}(\sigma)$
2 : $c_{j+1} \coloneqq \mathsf{H}(j+1, R, m, a_j)$	2 : **for** $i = 0, \ldots, n-1$ **do**
3 : **for** $i = j+1, \ldots, n-1, 0, \ldots, j-1$ **do**	3 : $a_i \coloneqq \mathsf{Rec}(pk_i, c_i, z_i)$
4 : $(a_i, z_i) \leftarrow \mathsf{Sim}(pk_i, c_i)$	4 : **if** $a_i = \bot$ **then return** $\bot$
5 : $c_{i+1} \coloneqq \mathsf{H}(i+1, R, m, a_i)$	5 : $c_{i+1} \coloneqq \mathsf{H}(i+1, R, m, a_i)$
6 : $z_j \leftarrow \mathsf{P}_2(sk_j, st_j, c_j)$	6 : **if** $c_n = c_0$ **then return** $\top$
7 : $\sigma \coloneqq (c_0, z_0, \ldots, z_{n-1})$	7 : **else return** $\bot$
8 : **return** σ	

Fig. 10. Generic construction of ring signatures from Fiat-Shamir (Definition 2.1).

A.2 AOS Transform

Figure 10 provides a generic template for the signature-generation and verification algorithms of AOS generic construction.

$\mathsf{FSwA}[\mathsf{ID}].\mathsf{Sign}(sk, m)$	$\mathsf{FSwA}[\mathsf{ID}].\mathsf{Vrfy}(pk, m, \sigma)$
1 : **repeat**	1 : $(c, z) \coloneqq \mathsf{Parse}(\sigma)$
2 : $(a, st) \leftarrow \mathsf{P}_1(sk)$	2 : $a' \coloneqq \mathsf{Rec}(pk, c, z)$
3 : $c \leftarrow \mathsf{H}(m, a)$	3 : **if** $a' = \bot$ **then**
4 : $z \leftarrow \mathsf{P}_2(sk, st, c)$	4 : **return** $\bot$
5 : **until** $z \neq \bot$	5 : **if** $c = \mathsf{H}(m, a')$ **then**
6 : $\sigma \coloneqq (c, z)$	6 : **return** $\top$
7 : **return** σ	7 : **else return** $\bot$

Fig. 8. Fiat-Shamir with aborts, based on an identification scheme (Definition 2.4).

$\mathsf{HSwR}[\mathsf{TDF}].\mathsf{Sign}(sk, m)$	$\mathsf{HSwR}[\mathsf{TDF}].\mathsf{Vrfy}(pk, m, \sigma)$
1 : **repeat**	1 : $(r, x) \coloneqq \mathsf{Parse}(\sigma)$
2 : $r \leftarrow \mathcal{S}_{\mathsf{salt}}$	2 : **if** $\mathsf{F}(pk, x) = \mathsf{H}(m, r)$ **then**
3 : $y \leftarrow \mathsf{H}(m, r)$	3 : **return** $\top$
4 : $x \leftarrow \mathsf{Inv}(sk, y)$	4 : **else return** $\bot$
5 : **until** $x \neq \bot$	
6 : $\sigma \coloneqq (r, x)$	
7 : **return** σ	

Fig. 9. Hash-and-sign with retry, based on a trapdoor function (Definition 2.5).

B Detailed ML-DSA Specification

This section describes the signature-generation and verification procedures shown in Figures 11 and 12. We first make explicit how these procedures differ from FIPS 204 ML-DSA; the subsequent paragraphs then define their encodings and each algorithm in detail.

Ring-ML-DSA.Sign(sk_j, R, M') / sRing-ML-DSA.Sign(sk_j, R, M')

$R \coloneqq \mathsf{CanonRing}(R)$
if $R = \perp$ **then return** $\perp$
$R = (pk'_0, \ldots, pk'_{n-1})$
$j \coloneqq$ index in R of public key corresponding to sk_j
if $j = \perp$ **then return** $\perp$
$(\rho_j, K_j, tr_j, \boldsymbol{s}_{1,j}, \boldsymbol{s}_{2,j}, \boldsymbol{t}_{0,j}) \coloneqq \mathsf{skDecode}(sk_j)$
$rnd \leftarrow \{0,1\}^{256}$ ∥Ring-ML-DSA
$\hat{\boldsymbol{s}}_{1,j} \coloneqq \mathsf{NTT}(\boldsymbol{s}_{1,j}), \quad \hat{\boldsymbol{s}}_{2,j} \coloneqq \mathsf{NTT}(\boldsymbol{s}_{2,j}), \quad \hat{\boldsymbol{A}}_j \coloneqq \mathsf{ExpandA}(\rho_j)$
for $i = [n] \backslash \{j\}$ **do**
  $(pk_i, \boldsymbol{t}_{0,i}) \coloneqq \mathsf{PkParse}'(pk'_i), \quad (\rho_i, \boldsymbol{t}_{1,i}) \coloneqq \mathsf{pkDecode}(pk_i)$
  $\hat{\boldsymbol{t}}_i \coloneqq \mathsf{NTT}(\boldsymbol{t}_{1,i} \cdot 2^d + \boldsymbol{t}_{0,i}), \quad \hat{\boldsymbol{A}}_i \coloneqq \mathsf{ExpandA}(\rho_i)$
$tr_R \coloneqq \mathsf{H}(\mathsf{ringEncode}(R), 64), \quad \mu \coloneqq \mathsf{H}(\mathsf{BytesToBits}(tr_R) \| M', 64)$
$\rho'' \coloneqq \mathsf{H}(K_j \| rnd \| \mu, 64)$ ∥Ring-ML-DSA
$\kappa \coloneqq 0$ ∥Ring-ML-DSA
$\boldsymbol{z}_j \coloneqq \perp$
while $\boldsymbol{z}_j = \perp$ **do**
  $\boldsymbol{y}_j \coloneqq \mathsf{ExpandMask}(\rho'', \kappa)$ ∥Ring-ML-DSA
  $\boldsymbol{y}_j \leftarrow \mathcal{Y}$ ∥sRing-ML-DSA
  $\kappa \coloneqq \kappa + \ell$ ∥Ring-ML-DSA
  $\boldsymbol{w}_j \coloneqq \mathsf{NTT}^{-1}(\hat{\boldsymbol{A}}_j \circ \mathsf{NTT}(\boldsymbol{y}_j)), \quad \boldsymbol{w}_{1,j} \coloneqq \mathsf{HighBits}(\boldsymbol{w}_j)$
  $\tilde{c}_{j+1} \coloneqq \mathsf{H}(\mu \| \mathsf{IntegerToBytes}(j+1, 2) \| \mathsf{w1Encode}(\boldsymbol{w}_{1,j}), \lambda/4)$
  for $r = 1, \ldots, n-1$ **do**
    $i \coloneqq j + r \bmod n, \quad c_i \coloneqq \mathsf{SampleInBall}(\tilde{c}_i), \quad \hat{c}_i \coloneqq \mathsf{NTT}(c_i)$
    $\boldsymbol{z}_i \coloneqq \perp$
    while $\boldsymbol{z}_i = \perp$ **do**
      $\boldsymbol{z}_i \coloneqq \mathsf{ExpandMask}(\rho'', \kappa)$ ∥Ring-ML-DSA
      $\boldsymbol{z}_i \leftarrow \mathcal{Y}$ ∥sRing-ML-DSA
      $\kappa \coloneqq \kappa + \ell$ ∥Ring-ML-DSA
      if $\|\boldsymbol{z}_i\|_\infty \geq \gamma_1 - \beta$ **then** $\boldsymbol{z}_i \coloneqq \perp$
      else
        $\boldsymbol{r}_i \coloneqq \mathsf{NTT}^{-1}\big(\hat{\boldsymbol{A}}_i \circ \mathsf{NTT}(\boldsymbol{z}_i) - \hat{c}_i \circ \hat{\boldsymbol{t}}_i\big)$
        if $\|\mathsf{LowBits}(\boldsymbol{r}_i)\|_\infty \geq \gamma_2 - \beta$ **then** $\boldsymbol{z}_i \coloneqq \perp$
        else
          $\boldsymbol{w}_{1,i} \coloneqq \mathsf{HighBits}(\boldsymbol{r}_i)$
          $\tilde{c}_{i+1} \coloneqq \mathsf{H}(\mu \| \mathsf{IntegerToBytes}(i+1, 2) \| \mathsf{w1Encode}(\boldsymbol{w}_{1,i}), \lambda/4)$
  $c_j \coloneqq \mathsf{SampleInBall}(\tilde{c}_j), \quad \hat{c}_j \coloneqq \mathsf{NTT}(c_j)$
  $\langle\langle c_j \boldsymbol{s}_{1,j} \rangle\rangle \coloneqq \mathsf{NTT}^{-1}(\hat{c}_j \circ \hat{\boldsymbol{s}}_{1,j}), \quad \langle\langle c_j \boldsymbol{s}_{2,j} \rangle\rangle \coloneqq \mathsf{NTT}^{-1}(\hat{c}_j \circ \hat{\boldsymbol{s}}_{2,j})$
  $\boldsymbol{z}_j \coloneqq \boldsymbol{y}_j + \langle\langle c_j \boldsymbol{s}_{1,j} \rangle\rangle, \quad \boldsymbol{r}_{0,j} \coloneqq \mathsf{LowBits}(\boldsymbol{w}_j - \langle\langle c_j \boldsymbol{s}_{2,j} \rangle\rangle)$
  if $\|\boldsymbol{z}_j\|_\infty \geq \gamma_1 - \beta$ **or** $\|\boldsymbol{r}_{0,j}\|_\infty \geq \gamma_2 - \beta$ **then** $\boldsymbol{z}_j \coloneqq \perp$
$\sigma \coloneqq \mathsf{ringSigEncode}(\tilde{c}_0, \boldsymbol{z}_0, \ldots, \boldsymbol{z}_{n-1})$
return σ

Fig. 11. Ring-ML-DSA and simplified Ring-ML-DSA signing algorithms (Definition 2.1) with the simulator written inline. Indices are taken modulo n.

$\mathsf{Ring\text{-}ML\text{-}DSA.Verify}(R, M', \sigma)$

$R \coloneqq \mathsf{CanonRing}(R)$ //includes RankCheck
if $R = \perp$ **then return** $\perp$
$R = (pk'_0, \ldots, pk'_{n-1})$
$(\tilde{c}_0, \boldsymbol{z}_0, \ldots, \boldsymbol{z}_{n-1}) \coloneqq \mathsf{ringSigDecode}(\sigma, n)$
$tr_R \coloneqq \mathsf{H}(\mathsf{ringEncode}(R), 64)$
$\mu \coloneqq \mathsf{H}(\mathsf{BytesToBits}(tr_R) \| M', 64)$
for $i = 0, 1, \ldots, n-1$ **do**
  $(pk_i, \boldsymbol{t}_{0,i}) \coloneqq \mathsf{PkParse}'(pk'_i)$
  $(\rho_i, \boldsymbol{t}_{1,i}) \coloneqq \mathsf{pkDecode}(pk_i)$
  $\hat{\boldsymbol{A}}_i \coloneqq \mathsf{ExpandA}(\rho_i)$
  $\hat{\boldsymbol{t}}_i \coloneqq \mathsf{NTT}(\boldsymbol{t}_{1,i} \cdot 2^d + \boldsymbol{t}_{0,i})$
  $c_i \coloneqq \mathsf{SampleInBall}(\tilde{c}_i)$
  $\hat{c}_i \coloneqq \mathsf{NTT}(c_i)$
  if $\|\boldsymbol{z}_i\|_\infty \geq \gamma_1 - \beta$ **then return** $\perp$
  $\boldsymbol{r}_i \coloneqq \mathsf{NTT}^{-1}\left(\hat{\boldsymbol{A}}_i \circ \mathsf{NTT}(\boldsymbol{z}_i) - \hat{c}_i \circ \hat{\boldsymbol{t}}_i\right)$
  $\boldsymbol{w}'_{1,i} \coloneqq \mathsf{HighBits}(\boldsymbol{r}_i)$
  $\tilde{c}_{i+1} \coloneqq \mathsf{H}(\mu \| \mathsf{IntegerToBytes}(i+1, 2) \| \mathsf{w1Encode}(\boldsymbol{w}'_{1,i}), \lambda/4)$
if $\tilde{c}_n = \tilde{c}_0$ **then return** $\top$
else return $\perp$

Fig. 12. ML-DSA-based ring-verification algorithm (Definition 2.1). Indices are taken modulo n.

Differences from FIPS 204 ML-DSA. Ring-ML-DSA retains the ML-DSA module relation, parameter sets, mask and challenge samplers, decomposition functions, NTT arithmetic, and secret-key format. Its departure from FIPS 204 has three major, logically distinct components.

The first is the ring transformation. The canonical ring and the message determine a common context for one cyclic challenge chain. Exactly one position contains the real signer response, the other $n-1$ positions contain publicly simulated responses, and verification accepts only when the last local transcript closes the chain back to its initial challenge.

The second is the augmented public key, which makes the ML-DSA hint unnecessary. A FIPS 204 public key contains $(\rho, \boldsymbol{t}_1)$ but not the low-order component $\boldsymbol{t}_0$. Its verifier can therefore form only the full-key approximation $\boldsymbol{Az} - c2^d\boldsymbol{t}_1$ and uses the hint $\boldsymbol{h}$ to recover the intended high-order commitment. Ring-ML-DSA additionally publishes $\boldsymbol{t}_0$, as is needed to simulate a non-signer transcript from public information. Hence every verifier knows $\boldsymbol{t} = 2^d\boldsymbol{t}_1 + \boldsymbol{t}_0$ and can instead compute the exact full-key value $\boldsymbol{Az} - c(2^d\boldsymbol{t}_1 + \boldsymbol{t}_0) = \boldsymbol{Az} - c\boldsymbol{t}$ and apply $\mathsf{HighBits}$ directly. Consequently, Ring-ML-DSA generates and encodes no hint and has no hint-weight rejection test.

The third component is a public matrix-admissibility rule needed because a ring may contain adversarially chosen keys. For each augmented key, the top $\ell \times \ell$ square submatrix of $\mathsf{ExpandA}(\rho)$ is viewed as an $\mathbb{F}_q$-linear map, and its corank must not exceed the threshold fixed for the selected ML-DSA parameter set. This test turns the rank-dependent min-entropy estimate into a deterministic bound for every key admitted to the ring. It is performed before any challenge-chain randomness is generated. The exact differences are summarized in Table 3.

Encoding conventions. Ring-ML-DSA inherits $\mathsf{pkDecode}$, $\mathsf{skDecode}$, $\mathsf{w1Encode}$, $\mathsf{IntegerToBytes}$, and $\mathsf{skEncode}$ component encodings from FIPS 204 [Nat24]. It adds $\mathsf{PkEncode}'$, $\mathsf{PkParse}'$, $\mathsf{CanonRing}$, $\mathsf{ringEncode}$,

Table 3. Differences between FIPS 204 ML-DSA and Ring-ML-DSA.

Aspect	ML-DSA of FIPS 204	Ring-ML-DSA
Public key	$pk = \mathsf{pkEncode}(\rho, \boldsymbol{t}_1)$; $\boldsymbol{t}_0$ is encoded in the secret key.	$pk' = \mathsf{PkEncode}'(pk, \boldsymbol{t}_0)$; hence the verifier knows $\boldsymbol{t} = 2^d\boldsymbol{t}_1 + \boldsymbol{t}_0$.
Matrix admissibility	No public-key rank test.	Every ring member must satisfy (6); CanonRing rejects the entire ring otherwise, before signing or verification continues.
Message binding	$tr = \mathsf{H}(pk, 64)$ and $\mu = \mathsf{H}(\mathsf{BytesToBits}(tr)\|\|M', 64)$.	$tr_R = \mathsf{H}(\mathsf{ringEncode}(R), 64)$ and $\mu = \mathsf{H}(\mathsf{BytesToBits}(tr_R)\|\|M', 64)$, binding the message to the ring.
Challenges	One challenge seed $\tilde{c}$ is obtained by hashing $\mathsf{H}(\mu\|\|\mathsf{w1Encode}(\boldsymbol{w}_{1,i}), \lambda/4)$, and $c = \mathsf{SampleInBall}(\tilde{c})$.	Each local commitment determines the next seed $\tilde{c}_{i+1} = \mathsf{H}(\mu\|\|\mathsf{IntegerToBytes}(i+1, 2)\|\|\mathsf{w1Encode}(\boldsymbol{w}_{1,i}), \lambda/4)$ and $c_{i+1} = \mathsf{SampleInBall}(\tilde{c}_{i+1})$; acceptance requires $\tilde{c}_n = \tilde{c}_0$.
Signature	$(\tilde{c}, \boldsymbol{z}, \boldsymbol{h})$.	$(\tilde{c}_0, \boldsymbol{z}_0, \ldots, \boldsymbol{z}_{n-1})$.
Hint	Hint is used to reconstruct $\boldsymbol{w}'_1 = \mathsf{UseHint}(\boldsymbol{h}, \boldsymbol{A}\boldsymbol{z} - c2^d\boldsymbol{t}_1)$.	Hint is not used. The verifier directly computes $\boldsymbol{w}'_{1,i} = \mathsf{HighBits}(\boldsymbol{A}_i\boldsymbol{z}_i - c_i\boldsymbol{t}_i)$.
Rejection	Signing restarts after a failed response, LowBits, $c\boldsymbol{t}_0$, or hint-weight test.	The response-range and LowBits tests remain. The $c\boldsymbol{t}_0$ and hint-weight tests are absent; failure at any genuine or simulated position restarts the entire cyclic chain.

ringSigEncode, and ringSigDecode. Define $\mathsf{PkEncode}'(pk, \boldsymbol{t}_0)$ as the FIPS public-key string pk followed by skEncode's canonical $\boldsymbol{t}_0$ encoding, and let $\mathsf{PkParse}'$ be the corresponding ring-level parser. The FIPS decoders are unchanged.

We next define the matrix-admissibility predicate. For a parsed key $pk' = (\rho, \boldsymbol{t}_1, \boldsymbol{t}_0)$, write $\boldsymbol{A}_\rho = \mathsf{ExpandA}(\rho)$ and let $\boldsymbol{A}^\circ_\rho \in R_q^{\ell\times\ell}$ be its top-most square submatrix. Viewing the induced map on R_q^ℓ as an $\mathbb{F}_q$-linear map of dimension 256ℓ, define

$$\mathrm{corank}_q(pk') \coloneqq 256\ell - \mathrm{rank}_{\mathbb{F}_q}(\boldsymbol{A}^\circ_\rho), \qquad \chi \coloneqq \begin{cases} 3 & \text{for ML-DSA-44,} \\ 19 & \text{for ML-DSA-65,} \\ 28 & \text{for ML-DSA-87.} \end{cases} \tag{6}$$

The key-admission procedure $\mathsf{RankCheck}(pk')$ returns $\top$ if $\mathrm{corank}_q(pk') \le \chi$, and returns $\bot$ otherwise. These thresholds are the conservative parameter-set choices in the rank-tail analysis of Barbosa et al. [BBD+23, Appendix A, Fig. 12].

The procedure CanonRing parses a candidate ring, rejects malformed augmented public-key strings, duplicates, and every key for which $\mathsf{RankCheck}(pk')$ is $\bot$, checks that all keys use the same ML-DSA parameter set, rejects $n = 0$ or $n \ge 2^{16}$, and sorts the encoded keys lexicographically. Thus, caller order is outside the scheme syntax and no challenge-chain randomness is sampled unless all ring members pass the rank test. For the ring $R = (pk'_0, \ldots, pk'_{n-1})$ returned by CanonRing, define

$$\mathsf{ringEncode}(R) \coloneqq \mathsf{IntegerToBytes}(n, 2)\|pk'_0\|\cdots\|pk'_{n-1}.$$

Because all ring keys use one parameter set, this encoding is injective and order preserving; it does not re-encode public keys. A signature is encoded as

$$\mathsf{ringSigEncode}\big(\tilde{c}_0, \boldsymbol{z}_0, \ldots, \boldsymbol{z}_{n-1}\big),$$

the initial challenge seed followed by n response encodings in canonical order. Its inverse $\mathsf{ringSigDecode}(\sigma, n)$ returns the tuple using n.

Ring binding and message representative. Ring-ML-DSA retains the separation between public-key digest and message representative, but replaces the former with a ring digest using ringEncode. For the ring $R = (pk'_0, \ldots, pk'_{n-1})$, set $tr_R \coloneqq \mathsf{H}(\mathsf{ringEncode}(R), 64)$ and $\mu \coloneqq \mathsf{H}(\mathsf{BytesToBits}(tr_R)\|M', 64)$ instead of using the ML-DSA public-key digest tr. Here tr_R binds the message to the ring. Although decoding sk_j still yields tr_j, μ does not use it; transcripts cannot move between rings, while permutations of one ring have the same value.

Signing algorithm. Ring-ML-DSA inherits skDecode, pkDecode, NTT, ExpandA, ExpandMask, SampleInBall, HighBits, and LowBits. It additionally defines CanonRing, ringEncode, ringSigEncode, and the indexed challenge hash $\mathsf{H}(\mu\|\mathsf{IntegerToBytes}(\cdot, 2)\|\mathsf{w1Encode}(\cdot), \lambda/4)$; the original scheme samples a fresh $rnd \leftarrow \{0,1\}^{256}$ internally for every call and keeps the ML-DSA nonce derivation $\mathsf{H}(K_j\|rnd\|\mu, 64)$. Within one Ring-ML-DSA signing execution, a two-byte counter input to ExpandMask is never reused under the same seed ρ''; signing aborts before the counter would wrap.

Both Ring-ML-DSA and sRing-ML-DSA first canonicalize R and thereby test RankCheck for every ring public key. If any member fails, signing returns $\perp$ before the signer index is selected and before rnd or any challenge-chain value is sampled. Otherwise, both schemes use j as the signer's canonical index and write components in that order. The signer generates the real component at j and simulates the others. The simplified scheme samples $\boldsymbol{y}_j$ and each simulated candidate $\boldsymbol{z}_i$ freshly instead of deriving them from $\mathsf{H}(K_j\|rnd\|\mu, 64)$ and ExpandMask. At each non-signer position, the signer resamples $\boldsymbol{z}_i$ under the fixed incoming challenge until the response-range and LowBits tests accept, exactly as in $\mathsf{Sim}_{\mathrm{ML}}$. If the genuine response subsequently fails either test, the signer discards the cyclic chain and restarts from a fresh commitment at position j.

Verification algorithm. Ring-ML-DSA inherits pkDecode, ExpandA, SampleInBall, NTT, HighBits, and w1Encode. It adds CanonRing, ringEncode, ringSigDecode, and the indexed challenge hash.

The verifier invokes CanonRing before decoding the signature. This check is necessary for the present interface, in which Verify receives a raw, potentially adversarial ring: omitting it would extend the acceptance domain to keys outside the admissible-key domain used by the universal min-entropy statement and hence by the security theorem, unless the input type itself certifies prior canonicalization.

At position i, the augmented key gives the full public value $\boldsymbol{t}_i = 2^d\boldsymbol{t}_{1,i} + \boldsymbol{t}_{0,i}$. After checking the response norm, the verifier reconstructs the commitment directly as $\boldsymbol{w}'_{1,i} = \mathsf{HighBits}(\boldsymbol{A}_i\boldsymbol{z}_i - c_i\boldsymbol{t}_i)$. The verifier uses the hash of this reconstructed commitment as the next challenge seed. The ring verifier starts from $\tilde{c}_0$ and applies this verification step at each ring position. It accepts if and only if every local step accepts and the chain returns to the initial seed.

Correctness follows from the full public-key equation. At the signer position, $\boldsymbol{z}_j = \boldsymbol{y}_j + c_j\boldsymbol{s}_{1,j}$ and $\boldsymbol{t}_j = \boldsymbol{A}_j\boldsymbol{s}_{1,j} + \boldsymbol{s}_{2,j}$ imply

$$\boldsymbol{A}_j\boldsymbol{z}_j - c_j\boldsymbol{t}_j = \boldsymbol{A}_j\boldsymbol{y}_j - c_j\boldsymbol{s}_{2,j}.$$

The LowBits rejection condition and $\|c_j\boldsymbol{s}_{2,j}\|_\infty \le \beta$ ensure that

$$\mathsf{HighBits}(\boldsymbol{A}_j\boldsymbol{z}_j - c_j\boldsymbol{t}_j) = \mathsf{HighBits}(\boldsymbol{A}_j\boldsymbol{y}_j).$$

Thus, the verifier reconstructs the commitment used by the signer, while at each simulated position the signer and verifier compute the same commitment directly. Hence every accepted signing attempt closes the cyclic challenge chain.

Size overhead. The size calculation uses the FIPS 204 challenge-seed and response encodings. ringSigEncode stores one initial seed and one response vector per ring member. The verification input R may be transmitted, context-fixed, or canonically derived, and is excluded from the payload. The generic FSwA payload has size $|\tilde{c}| + \sum_{i=0}^{n-1}|\boldsymbol{z}_i|$: one initial challenge seed is stored, and each ring position contributes an ML-DSA response vector. The table retains the standardized ML-DSA signature size only as a comparison column. Removing the hint saves $\omega + k$ bytes per ring member: 84, 61, and 83 bytes for parameter sets 44, 65, and 87, respectively.

Table 4. Estimated ML-DSA-based ring-signature payload sizes. Here n is the ring size.

Parameter set	Per-member payload	Total payload
ML-DSA-44, $(k,\ell,\tau,\lambda)=(4,4,39,128)$	2304 bytes	$32+2304n$ bytes
ML-DSA-65, $(k,\ell,\tau,\lambda)=(6,5,49,192)$	3200 bytes	$48+3200n$ bytes
ML-DSA-87, $(k,\ell,\tau,\lambda)=(8,7,60,256)$	4480 bytes	$64+4480n$ bytes

C Simulation and Entropy Properties of Ring-ML-DSA

This section establishes the three simulator properties used by the generic FSwA security theorem: universal simulation correctness, honest and simulated commitment min-entropy, and perfect ke-acHVZK.

Universal simulation correctness. The generic proof uses simulator $\mathsf{Sim}_{\mathrm{ML}}$, which is not part of the public Ring-ML-DSA interface. For a parsed augmented key $pk'=(\rho,\boldsymbol{t}_1,\boldsymbol{t}_0)$, put $\boldsymbol{A}\coloneqq\mathsf{NTT}^{-1}(\mathsf{ExpandA}(\rho))$ and $\boldsymbol{t}\coloneqq 2^d\boldsymbol{t}_1+\boldsymbol{t}_0$, and define

$$\mathcal{Z}\coloneqq[-\gamma_1+\beta+1,\gamma_1-\beta-1]^{256\ell},$$
$$p_{\mathrm{low}}(pk',c)\coloneqq\Pr_{\boldsymbol{z}\leftarrow\mathcal{Z}}\left[\left\|\mathsf{LowBits}(\boldsymbol{A}\boldsymbol{z}-c\boldsymbol{t})\right\|_\infty<\gamma_2-\beta\right].$$

On input (pk',c), $\mathsf{Sim}_{\mathrm{ML}}$ parses the key, samples $\boldsymbol{z}\leftarrow\mathcal{Z}$ until $\|\mathsf{LowBits}(\boldsymbol{A}\boldsymbol{z}-c\boldsymbol{t})\|_\infty<\gamma_2-\beta$, and returns $(\mathsf{HighBits}(\boldsymbol{A}\boldsymbol{z}-c\boldsymbol{t}),\boldsymbol{z})$.

Following the heuristic used in Dilithium analysis of [KLS18, BBD+23], we model the coefficients of $\mathsf{LowBits}(\boldsymbol{A}\boldsymbol{z}-c\boldsymbol{t})$ as independent uniform values in their coefficient range. We use this heuristic uniformly over every admitted augmented key and challenge, which is stronger than its usual honest-key application, and formalize it by the bound

$$p_{\mathrm{low}}(pk',c)\geq p_{\mathrm{unif}}\coloneqq\left(\frac{2(\gamma_2-\beta)-1}{2\gamma_2}\right)^{256k},\qquad p_{\mathrm{unif}}^{-1}=\mathsf{poly}(\lambda),\tag{7}$$

for every well-formed augmented key admitted by $\mathsf{CanonRing}$ and every $c\in\mathcal{B}_\tau$.

Lemma C.1 (Universal simulation correctness under the uniform-$\mathsf{LowBits}$ heuristic). *Under* (7), $\mathsf{Sim}_{\mathrm{ML}}$ *satisfies universal simulation correctness in the sense of Definition 2.6. In particular, for every admitted augmented key* pk' *and every* $c\in\mathcal{B}_\tau$, *it terminates after at most* $1/p_{\mathrm{unif}}$ *trials in expectation, and every output* $(\boldsymbol{w}_1,\boldsymbol{z})$ *satisfies* $\mathsf{V}(pk',\boldsymbol{w}_1,c,\boldsymbol{z})=1$ *and* $\mathsf{Rec}(pk',c,\boldsymbol{z})=\boldsymbol{w}_1$.

Proof. Every simulator output is accepted and reconstructed correctly. Sampling $\boldsymbol{z}$ from $\mathcal{Z}$ guarantees the verifier's response-range condition $\|\boldsymbol{z}\|_\infty<\gamma_1-\beta$. For each accepted trial, the simulator computes $\boldsymbol{r}=\boldsymbol{A}\boldsymbol{z}-c\boldsymbol{t}$ and sets $\boldsymbol{w}_1=\mathsf{HighBits}(\boldsymbol{r})$. Because the verifier parses the same $\boldsymbol{A}$ and $\boldsymbol{t}=2^d\boldsymbol{t}_1+\boldsymbol{t}_0$, it recomputes exactly the same $\boldsymbol{r}$ and therefore recovers exactly the simulator's output $\boldsymbol{w}_1=\mathsf{HighBits}(\boldsymbol{r})$. Thus, local verification and commitment recovery hold deterministically for every admitted key, independently of how that key was generated. The simulator's additional $\mathsf{LowBits}$ test is not needed for this algebraic reconstruction; it ensures that the simulated response follows the same acceptance condition used in the signing distribution and is needed for the subsequent termination and entropy analyses.

The uniform-$\mathsf{LowBits}$ heuristic gives universal expected-polynomial termination. For fixed (pk',c), the number of trials is geometric with success probability $p_{\mathrm{low}}(pk',c)$. Equation (7) therefore bounds its expectation by $1/p_{\mathrm{unif}}=\mathsf{poly}(\lambda)$, uniformly over all admitted keys and challenges. □

Signer-response rejection bound. Theorem 5.1 requires a uniform upper bound on the rejection probability of the response at the signer position, conditioned on the preceding execution history. For a fixed challenge c,

the signer computes $\boldsymbol{z} = \boldsymbol{y} + c\boldsymbol{s}_1$. Since $\|c\boldsymbol{s}_1\|_\infty \le \beta$, the standard ML-DSA rejection-sampling argument gives

$$\Pr[\|\boldsymbol{z}\|_\infty < \gamma_1 - \beta] = \left(\frac{2(\gamma_1 - \beta) - 1}{2\gamma_1 - 1}\right)^{256\ell}.$$

Conditioned on this event, $\boldsymbol{z}$ is uniform over $\mathcal{Z}$. Under the uniform-$\mathsf{LowBits}$ heuristic, applied conditionally to each fresh signing attempt after the challenge chain closes, we therefore have

$$1 - \varepsilon_{\mathrm{ML}} \ge \left(\frac{2(\gamma_1 - \beta) - 1}{2\gamma_1 - 1}\right)^{256\ell} \left(\frac{2(\gamma_2 - \beta) - 1}{2\gamma_2}\right)^{256k}.$$

Consequently, the expected number of outer signing attempts is at most

$$(1 - \varepsilon_{\mathrm{ML}})^{-1} \le \left(\frac{2\gamma_1 - 1}{2(\gamma_1 - \beta) - 1}\right)^{256\ell} \left(\frac{2\gamma_2}{2(\gamma_2 - \beta) - 1}\right)^{256k} = \mathsf{poly}(\lambda).$$

For ML-DSA-44, ML-DSA-65, and ML-DSA-87, respectively, this gives expected attempt bounds of approximately 4.28, 5.11, and 3.87.

Commitment min-entropy. We next derive the honest and simulated commitment min-entropy parameters required by Theorem 5.1.

Lemma C.2 (Honest and simulated commitment min-entropy). *Let δ satisfy*

$$\Pr_{pk' \leftarrow \mathsf{Gen}(\lambda)}\left[\mathrm{corank}_q(pk') > \chi\right] \le \delta,$$

where pk' is the augmented public key produced by honest key generation. Suppose further that there exists $p_\star > 0$ such that $p_{\mathrm{low}}(pk', c) \ge p_\star$ for every augmented key pk' satisfying $\mathsf{RankCheck}(pk') = \top$ and every challenge $c \in \mathcal{B}_\tau$. Define

$$\alpha := \max\left\{0,\ 256\ell \log_2\left(\frac{2\gamma_1 - 1}{2\gamma_2 + 1}\right) - \chi \log_2 q\right\},$$
$$\alpha' := \max\left\{0,\ 256\ell \log_2\left(\frac{2(\gamma_1 - \beta) - 1}{2\gamma_2 + 1}\right) - \chi \log_2 q - \log_2(1/p_\star)\right\}.$$

Then, the honest augmented ML-DSA protocol has (α, δ)-min-entropy commitments. Moreover, for every augmented key pk' satisfying $\mathsf{RankCheck}(pk') = \top$ and every $c \in \mathcal{B}_\tau$, the commitment output by $\mathsf{Sim}_{\mathrm{ML}}(pk', c)$ has min-entropy at least α'.

Proof. The honest-commitment claim is exactly the ML-DSA min-entropy bound of Barbosa et al. [BBD+23, Appendix A, Eq. (21)]. Neither augmenting the public key with $\boldsymbol{t}_0$ nor omitting the hint changes the honest mask $\boldsymbol{y}$ or the commitment $\mathsf{HighBits}(\boldsymbol{A}\boldsymbol{y})$, so their analysis applies without modification. In particular, their rank-dependent bound and rank-tail estimate, with $256\ell - r \le \chi$ except with probability δ, give the stated (α, δ)-min-entropy property.

The simulated-commitment claim extends the same counting argument to the simulator's response distribution. Fix an admitted augmented key pk' and a challenge $c \in \mathcal{B}_\tau$, and let r be the rank over $\mathbb{F}_q$ of the map induced by the top-most square submatrix $\boldsymbol{A}^\circ$. Before the $\mathsf{LowBits}$ test, the simulator samples $\boldsymbol{z}$ uniformly from $\mathcal{Z}$, which has cardinality $(2(\gamma_1 - \beta) - 1)^{256\ell}$, and forms $\mathsf{HighBits}(\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t})$. For each fixed commitment, the argument of Barbosa et al. bounds the number of corresponding responses by $(2\gamma_2 + 1)^{256\ell} q^{256\ell - r}$; the fixed term $-c\boldsymbol{t}$ does not change this number. Consequently,

$$H_\infty(\mathsf{HighBits}(\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t})) \ge \max\left\{0,\ 256\ell \log_2\left(\frac{2(\gamma_1 - \beta) - 1}{2\gamma_2 + 1}\right) - (256\ell - r)\log_2 q\right\}.$$

Table 5. Rank-tail and commitment min-entropy bounds for ML-DSA.

Parameter set	χ	$-\log_2 \delta$	α	α'
ML-DSA-44	3	≥ 64	402.89	400.79
ML-DSA-65	19	≥ 362	844.83	842.48
ML-DSA-87	28	≥ 540	1150.56	1148.61

Since $\mathsf{RankCheck}(pk') = \top$, we may replace $256\ell - r$ by χ in this lower bound.

The simulator's rejection step accounts for the remaining entropy loss. Conditioning on $\mathsf{LowBits}$ acceptance can increase any point probability by at most $1/p_{\mathrm{low}}(pk', c)$ and hence reduces min-entropy by at most $\log_2(1/p_{\mathrm{low}}(pk', c))$. The uniform bound $p_{\mathrm{low}}(pk', c) \geq p_\star$ therefore reduces the preceding rank-uniform bound by at most $\log_2(1/p_\star)$, which gives α'. □

The bounds on α and α' depend on the chosen corank threshold χ: increasing χ decreases the rank-tail probability δ, but also decreases α and α'. Using the rank-tail estimates of Barbosa et al. [BBD+23, Appendix A, Fig. 12], we obtain the values of Table 5. The values of α' use $p_\star = p_{\mathrm{unif}}$ from Equation (7). Here, δ is the probability that an honestly generated key exceeds the chosen corank threshold; it is unrelated to the simulator's rejection probability.

Perfect ke-acHVZK. We finally show that the accepted real and simulated transcript distributions remain identical even when the key-generation coins are exposed.

Lemma C.3 (Perfect ke-acHVZK). *For every QPT distinguisher $\mathcal{D}$,*

$$\mathsf{Adv}^{\mathsf{ke\text{-}acHVZK}}_{\mathsf{ID}_{\mathrm{ML}},\mathsf{Sim}_{\mathrm{ML}},\mathcal{D}}(\lambda) = 0.$$

Proof. Fix the key-generation tape and a challenge c, so that $\boldsymbol{A}, \boldsymbol{s}_1, \boldsymbol{s}_2$, and $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2$ are fixed. Define

$$\mathcal{Z}_{pk',c} \coloneqq \{\boldsymbol{z} \in \mathcal{Z} : \|\mathsf{LowBits}(\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t})\|_\infty < \gamma_2 - \beta\}.$$

The standard ML-DSA rejection-sampling argument [KLS18, Lemma 4.3] [KX25, Lemma 6] applies pointwise to the fixed key and challenge. Indeed, under the bijection $\boldsymbol{z} = \boldsymbol{y} + c\boldsymbol{s}_1$, we have

$$\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} = \boldsymbol{A}\boldsymbol{y} - c\boldsymbol{s}_2.$$

Hence the real response-range and $\mathsf{LowBits}$ tests map exactly to the conditions defining $\mathcal{Z}_{pk',c}$. The accepted real response and the simulator output are therefore both uniform over $\mathcal{Z}_{pk',c}$.

It remains to compare the commitments associated with each matched pair $(\boldsymbol{y}, \boldsymbol{z})$. The $\mathsf{LowBits}$ condition and $\|c\boldsymbol{s}_2\|_\infty \leq \beta$ imply, by decomposition stability, that

$$\mathsf{HighBits}(\boldsymbol{A}\boldsymbol{y}) = \mathsf{HighBits}(\boldsymbol{A}\boldsymbol{y} - c\boldsymbol{s}_2) = \mathsf{HighBits}(\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}).$$

Thus, the real and simulated transcripts $(\boldsymbol{w}_1, c, \boldsymbol{z})$ are identically distributed for every fixed key-generation tape and challenge. This pointwise equality remains valid when the tape is revealed, and hence the ke-acHVZK advantage is zero. □

D Concrete Bounds for Hardness Assumptions for Ring-ML-DSA

This section reduces the three residual security properties of the augmented-key ML-DSA to concrete module-lattice problems. The organization is as follows. Appendix D.1 fixes the norm convention, formally defines the MLWE, MSIS, and CCB advantages, and derives one verification equation shared by all three reductions. Appendix D.2 treats CUR, following the notion and the Dilithium analysis of Kiltz, Lyubashevsky, and Schaffner [KLS18]. Appendix D.3 treats CSS; this is an additional property needed by our ring-signature proof, not a condition used in the original Dilithium security bound. Appendix D.4 reduces IMP directly to chosen-coordinate binding (CCB), and Appendix D.5 instantiates the resulting bounds.

D.1 Common Setup

This subsection develops the common setup for the concrete analyses of CUR, CSS, and IMP. We first fix the algebraic notation, norm convention, and challenge space, and formally define the MLWE, MSIS, and CCB problems used in the reductions. We then state the idealized sampling assumptions and revisit the ML-DSA verification procedure in sufficient detail to derive the accepted-response relations shared by the three reductions. Finally, we clarify that the analysis concerns the augmented-key identification relation from the case study, rather than the relation induced by an unmodified FIPS 204 public key.

Algebraic model and norm convention. Let

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

For $f = \sum_{j=0}^{255} f_j X^j \in R_q$, choose each coefficient f_j in its centered representative interval and define

$$\|f\|_\infty \coloneqq \max_{0\le j<256} |f_j|.$$

For a module vector $\boldsymbol{x} = (x_1, \ldots, x_r) \in R_q^r$, set

$$\|\boldsymbol{x}\|_\infty \coloneqq \max_{1\le i\le r} \|x_i\|_\infty.$$

Thus, a concatenated vector satisfies

$$\|(\boldsymbol{x},\boldsymbol{y})\|_\infty = \max\{\|\boldsymbol{x}\|_\infty, \|\boldsymbol{y}\|_\infty\}.$$

Let

$$S_\eta \coloneqq \{f \in R_q : \|f\|_\infty \le \eta\}$$

with the coefficient distribution used by ML-DSA, and let $\mathcal{B}_\tau \subset R_q$ contain the polynomials with exactly τ nonzero coefficients, each in $\{-1,1\}$. For $c \in \mathcal{B}_\tau$, we have $\|c\|_\infty = 1$ and $\|cf\|_\infty \le \tau\|f\|_\infty$.

Module-lattice hard problems. We use the conventions of Kiltz, Lyubashevsky, and Schaffner [KLS18, Section 4.5.1] and of Jackson, Miller, and Wang [JMW24, Definitions 1, 2, and 8].

Definition D.1 (Module Learning with Errors (MLWE)). *For a QPT distinguisher $\mathcal{D}$, define*

$$\mathsf{Adv}_{\mathcal{D}}^{\mathsf{MLWE}_{k,\ell,\eta}} \coloneqq \left| \Pr\left[b = 1 : \begin{array}{c} \boldsymbol{A} \leftarrow R_q^{k\times\ell}, \\ \boldsymbol{t} \leftarrow R_q^k, \\ b \leftarrow \mathcal{D}(\boldsymbol{A},\boldsymbol{t}) \end{array} \right] - \Pr\left[b = 1 : \begin{array}{c} \boldsymbol{A} \leftarrow R_q^{k\times\ell}, \\ \boldsymbol{s}_1 \leftarrow S_\eta^\ell,\ \boldsymbol{s}_2 \leftarrow S_\eta^k, \\ b \leftarrow \mathcal{D}(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s}_1+\boldsymbol{s}_2) \end{array} \right] \right|$$

The $\mathsf{MLWE}_{k,\ell,\eta}$ assumption states that this advantage is negligible for every QPT $\mathcal{D}$.

Definition D.2 (Module Short Integer Solution (MSIS)). *For a QPT adversary $\mathcal{A}$, define*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{MSIS}_{m,r,B}} \coloneqq \Pr\left[\begin{array}{c} [\,I_m \mid \boldsymbol{G}\,]\boldsymbol{x} = 0 \\ \wedge\ 0 < \|\boldsymbol{x}\|_\infty \le B \end{array} : \begin{array}{c} \boldsymbol{G} \leftarrow R_q^{m\times r}, \\ \boldsymbol{x} \leftarrow \mathcal{A}(\boldsymbol{G}) \end{array} \right],$$

where $\boldsymbol{x} \in R_q^{m+r}$. The $\mathsf{MSIS}_{m,r,B}$ assumption states that this advantage is negligible for every QPT $\mathcal{A}$.

Definition D.3 (Chosen-coordinate binding (CCB)). *For a two-stage QPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, define*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{CCB}_{\tau,m,L,B}} \coloneqq \Pr\left[\begin{array}{c} \boldsymbol{H}\boldsymbol{y} = \boldsymbol{v} \\ \wedge\ \|\boldsymbol{y}\|_\infty \le B \\ \wedge\ y_L = c \end{array} : \begin{array}{c} \boldsymbol{H} \leftarrow R_q^{m\times L}, \\ (\boldsymbol{v}, st) \leftarrow \mathcal{A}_1(\boldsymbol{H}), \\ c \leftarrow \mathcal{B}_\tau, \\ \boldsymbol{y} \leftarrow \mathcal{A}_2(st, c) \end{array} \right],$$

where $\boldsymbol{v} \in R_q^m$ and $\boldsymbol{y} = (y_1, \ldots, y_L) \in R_q^L$. $\mathsf{CCB}_{\tau,m,L,B}$ assumption states that $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{CCB}_{\tau,m,L,B}} \le |\mathcal{B}_\tau|^{-1} + \mathsf{negl}(\lambda)$ holds for every QPT $\mathcal{A}$. The term $|\mathcal{B}_\tau|^{-1}$ accounts for guessing the challenge before the first stage fixes $\boldsymbol{v}$.

For clarity, our reductions analyze an idealized setting in which the public matrix $\boldsymbol{A}$ is sampled uniformly and the challenge $c \in \mathcal{B}_\tau$ is sampled uniformly. Ring-ML-DSA generates these values differently: $\boldsymbol{A}$ is expanded from the public seed ρ, while c is derived from a hash output using $\mathsf{SampleInBall}$. Consequently, the bounds proved below apply directly to the idealized setting. Applying them to the algorithms specified by FIPS 204 additionally requires an argument that these generation procedures can replace the uniform sampling assumed in the reductions.

From verification to short lattice relations. An accepted response determines two bounded vectors in a module relation. For every accepted $\boldsymbol{z}$, define

$$\boldsymbol{r} \coloneqq \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t}, \qquad \boldsymbol{e} \coloneqq \mathsf{LowBits}(\boldsymbol{r}).$$

The verification equation gives $\boldsymbol{w}_1 = \mathsf{HighBits}(\boldsymbol{r})$ and hence

$$\boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} = 2\gamma_2\boldsymbol{w}_1 + \boldsymbol{e}, \qquad \|\boldsymbol{e}\|_\infty \le \gamma_2, \tag{8}$$

using the FIPS 204 decomposition convention. Since the centered coefficients of $\boldsymbol{z}$ are integers, the verification condition $\|\boldsymbol{z}\|_\infty < \gamma_1 - \beta$ equivalently gives $\|\boldsymbol{z}\|_\infty \le \gamma_1 - \beta - 1$; moreover, $\|c\|_\infty = 1$.

D.2 Computational Unique Response (CUR)

Recalled notion. CUR was introduced in this form by Kiltz, Lyubashevsky, and Schaffner [KLS18, Definition 2.7]; recall the generic definition from Definition 2.12. For the augmented-key ML-DSA protocol, the specialized winning probability is

$$\mathsf{Adv}^{\mathsf{CUR}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{cur}}}(\lambda) = \Pr\left[\begin{array}{c|c} c \in \mathcal{B}_\tau \ \wedge\ \boldsymbol{z} \neq \boldsymbol{z}' & (pk', sk) \leftarrow \mathsf{ID}_{\mathrm{ML}}.\mathsf{Gen}(1^\lambda), \\ \wedge\ \mathsf{V}(pk', \boldsymbol{w}_1, c, \boldsymbol{z}) = 1 \ :\ & (\boldsymbol{w}_1, c, \boldsymbol{z}, \boldsymbol{z}') \leftarrow \mathcal{A}_{\mathrm{cur}}(pk') \\ \wedge\ \mathsf{V}(pk', \boldsymbol{w}_1, c, \boldsymbol{z}') = 1 & \end{array}\right].$$

Thus, the commitment and challenge are identical, while the complete response vectors must differ.

Reduction. We adapt the response-collision argument of Kiltz–Lyubashevsky–Schaffner [KLS18, Lemmas 4.8–4.9] to the augmented-key ML-DSA notation. Two accepted responses to the same commitment and challenge yield the homogeneous relation needed for MSIS. Suppose that the CUR adversary wins the recalled experiment. Applying (8) to both responses and subtracting gives $\boldsymbol{A}(\boldsymbol{z} - \boldsymbol{z}') + (\boldsymbol{e}' - \boldsymbol{e}) = 0$.

The differences form an MSIS solution in the sense of Definition D.2. Set $\boldsymbol{v} \coloneqq \boldsymbol{z} - \boldsymbol{z}'$, $\boldsymbol{u} \coloneqq \boldsymbol{e}' - \boldsymbol{e}$, and $\boldsymbol{x} \coloneqq (\boldsymbol{u}, \boldsymbol{v}) \in R_q^{k+\ell}$. In the MSIS definition, take $m = k$, $r = \ell$, and $\boldsymbol{G} = \boldsymbol{A}$. The preceding argument gives $\boldsymbol{v} \neq 0$ and hence $\boldsymbol{x} \neq 0$, while

$$[\, I_k \mid \boldsymbol{A} \,]\boldsymbol{x} = \boldsymbol{u} + \boldsymbol{A}\boldsymbol{v} = 0,$$
$$0 < \|\boldsymbol{x}\|_\infty \le \max\{2(\gamma_1 - \beta - 1), 2\gamma_2\} = 2(\gamma_1 - \beta - 1) =: B_{\mathrm{cur}}.$$

Thus, $\boldsymbol{x}$ is a valid $\mathsf{MSIS}_{k,\ell,B_{\mathrm{cur}}}$ solution. For the corresponding MSIS adversary $\mathcal{A}_{\mathrm{msis}}$,

$$\mathsf{Adv}^{\mathsf{CUR}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{cur}}}(\lambda) \le \mathsf{Adv}^{\mathsf{MSIS}_{k,\ell,B_{\mathrm{cur}}}}_{\mathcal{A}_{\mathrm{msis}}}. \tag{9}$$

The reduction has coefficient one; it samples the ML-DSA secrets itself and uses the MSIS matrix as $\boldsymbol{A}$.

D.3 Computational Special Soundness (CSS)

Recalled notion. Recall the non-extractive definition from Definition 2.13. For the augmented-key ML-DSA protocol, a CSS adversary outputs two response vectors and has advantage

$$\mathsf{Adv}^{\mathsf{CSS}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{css}}}(\lambda) = \Pr\left[\begin{array}{c|c} c, c' \in \mathcal{B}_\tau \ \wedge\ c \neq c' & (pk', sk) \leftarrow \mathsf{ID}_{\mathrm{ML}}.\mathsf{Gen}(1^\lambda), \\ \wedge\ \mathsf{V}(pk', \boldsymbol{w}_1, c, \boldsymbol{z}) = 1 \ :\ & (\boldsymbol{w}_1, c, \boldsymbol{z}, c', \boldsymbol{z}') \leftarrow \mathcal{A}_{\mathrm{css}}(pk') \\ \wedge\ \mathsf{V}(pk', \boldsymbol{w}_1, c', \boldsymbol{z}') = 1 & \end{array}\right].$$

Unlike CUR, the commitment is the same but the challenges are distinct. This computational formulation directly rules out such a pair; it does not assume an extractor that recovers $\boldsymbol{s}_1$ or $\boldsymbol{s}_2$.

Reduction. Condition on a winning pair and write $\Delta\boldsymbol{e} = \boldsymbol{e} - \boldsymbol{e}'$, $\Delta\boldsymbol{z} = \boldsymbol{z} - \boldsymbol{z}'$, and $\Delta c = c - c'$. Subtracting the full-public-value form (8) yields

$$[\, I_k \mid \boldsymbol{A} \mid -\boldsymbol{t} \,] \begin{pmatrix} -\Delta\boldsymbol{e} \\ \Delta\boldsymbol{z} \\ \Delta c \end{pmatrix} = 0.$$

Because $c, c' \in \mathcal{B}_\tau$,

$$\|\Delta\boldsymbol{e}\|_\infty \le 2\gamma_2, \qquad \|\Delta\boldsymbol{z}\|_\infty \le 2(\gamma_1 - \beta - 1), \qquad \|\Delta c\|_\infty \le 2.$$

The vector is nonzero because $\Delta c \neq 0$. Replacing the MLWE sample $(\boldsymbol{A}, \boldsymbol{t})$ by uniform makes $[\, \boldsymbol{A} \mid -\boldsymbol{t} \,] \in R_q^{k\times(\ell+1)}$ uniform. Hence, there are MLWE and MSIS adversaries $\mathcal{A}_{\mathrm{mlwe}}$ and $\mathcal{A}'_{\mathrm{msis}}$ such that

$$\mathsf{Adv}^{\mathsf{CSS}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{css}}}(\lambda) \le \mathsf{Adv}^{\mathsf{MLWE}_{k,\ell,\eta}}_{\mathcal{A}_{\mathrm{mlwe}}} + \mathsf{Adv}^{\mathsf{MSIS}_{k,\ell+1,B_{\mathrm{css}}}}_{\mathcal{A}'_{\mathrm{msis}}},$$

$$B_{\mathrm{css}} \coloneqq \max\{2(\gamma_1 - \beta - 1),\, 2,\, 2\gamma_2\} = 2(\gamma_1 - \beta - 1).$$

The first term is the usual $\mathsf{MLWE}_{k,\ell,\eta}$ public-key hybrid: it replaces $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2$ by an independent uniform vector. This is the same MLWE parameter shape used in the Dilithium security analysis [KLS18, Equations (21) and (24)]. After this replacement, however, CSS uses the enlarged matrix $[\, \boldsymbol{A} \mid -\boldsymbol{t} \,] \in R_q^{k\times(\ell+1)}$ and therefore the new instance $\mathsf{MSIS}_{k,\ell+1,B_{\mathrm{css}}}$. The additional column records the nonzero challenge difference Δc. This is *not* the $\mathsf{MSIS}_{k,\ell,B_{\mathrm{cur}}}$ instance in the existing Dilithium strong-unforgeability bound [KLS18, Equation (21)] [JMW24, Equation (4)]. The original analysis therefore does not supply a hardness estimate for this exact instance. We instead evaluate it separately, under the same BKZ/Core-SVP cost model used in the concrete Dilithium analysis, in Appendix D.5.

D.4 Impersonation (IMP)

Recalled notion. Recall the two-stage impersonation experiment from Definition 2.9, which follows Beckmann and Majenz [BM26, Definition 10]. Specialized to augmented-key ML-DSA, its advantage is

$$\mathsf{Adv}^{\mathsf{IMP}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{imp}}}(\lambda) = \Pr\left[\mathsf{V}(pk', \boldsymbol{w}_1, c, \boldsymbol{z}) = 1 : \begin{array}{c} (pk', sk) \leftarrow \mathsf{ID}_{\mathrm{ML}}.\mathsf{Gen}(1^\lambda), \\ (\boldsymbol{w}_1, st) \leftarrow \mathcal{A}_1(pk'), \\ c \leftarrow \mathcal{B}_\tau, \\ \boldsymbol{z} \leftarrow \mathcal{A}_2(st, c) \end{array}\right].$$

The first stage must commit to $\boldsymbol{w}_1$ before seeing the uniformly sampled challenge c.

Reduction. The reduction first replaces the ML-DSA public value by a uniform one. In the real-key experiment, $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s}_1 + \boldsymbol{s}_2$; replacing $\boldsymbol{t}$ by an independent uniform vector changes the IMP success probability by at most $\mathsf{Adv}^{\mathsf{MLWE}_{k,\ell,\eta}}_{\mathcal{A}_{\mathrm{mlwe}}}$. The remainder of the argument therefore works with uniform and independent $(\boldsymbol{A}, \boldsymbol{t})$.

The two stages of IMP then match the two stages of CCB. Before seeing the challenge, the first CCB stage constructs the augmented public key from $(\boldsymbol{A}, \boldsymbol{t})$, runs the first IMP stage to obtain $\boldsymbol{w}_1$, and fixes the CCB target to be $2\gamma_2\boldsymbol{w}_1$. The CCB experiment next samples $c \leftarrow \mathcal{B}_\tau$. The second CCB stage gives this same c to the second IMP stage and obtains $\boldsymbol{z}$.

Acceptance supplies the required short CCB preimage. Let $\boldsymbol{e}$ be the residual associated with the accepted response by (8). The second CCB stage returns the concrete vector $(-\boldsymbol{e}, \boldsymbol{z}, c)$, for which

$$[\, I_k \mid \boldsymbol{A} \mid -\boldsymbol{t} \,] \begin{pmatrix} -\boldsymbol{e} \\ \boldsymbol{z} \\ c \end{pmatrix} = \boldsymbol{A}\boldsymbol{z} - c\boldsymbol{t} - \boldsymbol{e} = 2\gamma_2\boldsymbol{w}_1,$$

$$\|(-\boldsymbol{e}, \boldsymbol{z}, c)\|_\infty \le \max\{\gamma_1 - \beta - 1, 1, \gamma_2\} =: B_{\mathrm{imp}}.$$

Table 6. Concrete algebraic bounds for the augmented-key ML-DSA relation.

Parameter set	MSIS dim. for CUR	MSIS dim. for CSS	CCB dim. for IMP	B_{cur}	B_{css}	B_{imp}
ML-DSA-44	$(4,4)$	$(4,5)$	$(4,9)$	261 986	261 986	130 993
ML-DSA-65	$(6,5)$	$(6,6)$	$(6,12)$	1 048 182	1 048 182	524 091
ML-DSA-87	$(8,7)$	$(8,8)$	$(8,16)$	1 048 334	1 048 334	524 167

The last coordinate of the returned vector is the sampled challenge c. Hence, all three CCB winning conditions hold: the vector maps to the target fixed by the first stage, it is short, and its designated coordinate equals the challenge.

The only remaining mismatch is the matrix distribution in the CCB definition. The preceding argument uses the Hermite-form matrix $[\, I_k \mid \boldsymbol{A} \mid -\boldsymbol{t}\,]$, whereas Definition D.3 samples a fully uniform matrix $\boldsymbol{H} \leftarrow R_q^{k\times(k+\ell+1)}$. Apply the row-reduction procedure of Jackson, Miller, and Wang while leaving the last, challenge-carrying column fixed [JMW24, Proposition 1]. Row reduction therefore uses the first $k+\ell$ columns, which form a uniform $k \times (k+\ell)$ module matrix. At each of the 256 CRT coordinates, the corresponding field matrix must have full row rank. Proposition 1 of the cited work, followed by a union bound over the CRT coordinates, gives the row-reduction failure bound $256q^{-\ell}$. Except with probability at most $\epsilon_{\mathrm{row}}(k,\ell,q)$, row reduction gives invertible row operations U and a permutation P fixing the last coordinate such that $U\boldsymbol{H}P = [\, I_k \mid \boldsymbol{A} \mid -\boldsymbol{t}\,]$, with $(\boldsymbol{A},\boldsymbol{t})$ distributed as in the uniform-key hybrid. Accordingly, the first CCB stage outputs $U^{-1}(2\gamma_2\boldsymbol{w}_1)$ as its target, and the second stage outputs $P(-\boldsymbol{e},\boldsymbol{z},c)^T$. The permutation preserves the norm and the last coordinate, while the row operation converts the displayed Hermite-form equation into the original equation for $\boldsymbol{H}$.

It follows that there are MLWE and CCB adversaries $\mathcal{A}_{\mathrm{mlwe}}$ and $\mathcal{A}_{\mathrm{ccb}}$ satisfying

$$\mathsf{Adv}^{\mathsf{IMP}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{imp}}}(\lambda) \le \mathsf{Adv}^{\mathsf{MLWE}_{k,\ell,\eta}}_{\mathcal{A}_{\mathrm{mlwe}}} + \mathsf{Adv}^{\mathsf{CCB}_{\tau,k,k+\ell+1,B_{\mathrm{imp}}}}_{\mathcal{A}_{\mathrm{ccb}}} + 256q^{-\ell}. \tag{10}$$

This reduction uses the challenge sampled by the IMP and CCB experiments and does not introduce a random oracle.

D.5 Concrete Parameters and Consequences

Numerical bounds. Substitution of the FIPS 204 parameters gives Table 6. CUR uses $\mathsf{MSIS}_{k,\ell,B_{\mathrm{cur}}}$, CSS uses $\mathsf{MSIS}_{k,\ell+1,B_{\mathrm{css}}}$, and IMP uses $\mathsf{CCB}_{\tau,k,k+\ell+1,B_{\mathrm{imp}}}$. The challenge-space size is $|\mathcal{B}_\tau| = 2^\tau\binom{256}{\tau}$.

For ML-DSA-44, ML-DSA-65, and ML-DSA-87, $\log_2|\mathcal{B}_\tau|$ is approximately 192.76, 225.35, and 257.01, respectively. The upper bound of $256q^{-\ell}$ is approximately $2^{-83.99}$, $2^{-106.99}$, and $2^{-152.99}$, respectively. This is a proof-loss term rather than a known attack, but it means that the displayed IMP reduction alone does not establish the nominal security category of any standard parameter set.

BKZ estimates for the CUR and CSS instances. The CUR and CSS instances can be evaluated by the same calculation that Jackson, Miller, and Wang use for Dilithium MSIS [JMW24, Section 4.2]. Following their method, we view $\mathsf{MSIS}_{a,b,\xi}$ as an SIS instance with $256\cdot a$ constraints and $256\cdot(a+b)$ variables, and choose the smallest BKZ block size satisfying their root-Hermite-factor estimate [JMW24, Equations (44)–(47)]. Applying this calculation to $\mathsf{MSIS}_{k,\ell,B_{\mathrm{cur}}}$ and $\mathsf{MSIS}_{k,\ell+1,B_{\mathrm{css}}}$ gives the CUR and CSS estimates in Table 7, respectively. Concretely, with base-2 logarithms and

$$\delta_0(b) = \left(\frac{(b\pi)^{1/b}\,b}{2\pi e}\right)^{1/(2(b-1))},$$

we select the least integer $b \ge 50$ satisfying

$$\frac{2^{2\sqrt{256a\log_2(q)\log_2(\delta_0(b))}}}{\sqrt{256(a+b')}} \le \xi$$

Table 7. BKZ/Core-SVP estimates for the CUR and CSS MSIS instances. For BKZ block size b, the classical and quantum attack costs are modeled as $2^{0.292b}$ and $2^{0.265b}$, respectively; their base-2 logarithms are reported in bits and rounded to the nearest integer.

	MSIS for CUR			MSIS for CSS		
Parameter set	b	Classical	Quantum	b	Classical	Quantum
ML-DSA-44	388	113	103	384	112	102
ML-DSA-65	533	156	141	530	155	140
ML-DSA-87	773	226	205	769	225	204

for $\mathsf{MSIS}_{a,b',\xi}$. Re-evaluating this inequality with the exact integer norm bounds in Table 6 gives the block sizes reported in Table 7; the one- or two-unit tightening of ξ does not change any selected block size.

The Core-SVP exponents are the conventional costs used in the Dilithium parameter analysis [LDK+20, Appendix C.3]. Under this model, the new CSS instance is weaker than the corresponding CUR instance, but still has the nontrivial attack costs shown in the table.

Consequences and limitations. Combining (10) with Lemma 5.1 conditionally discharges the residual EUF-NRA term under MLWE and CCB, but retains the generic exponential factor $\frac{1}{2}(2(Q_{\mathsf{H}}+N)+1)^{2N}$. Jackson, Miller, and Wang further show that CCB can be reduced to collapsing and, under additional conditions, to MLWE [JMW24, Propositions 3 and 4]. Those conditions include constraints on the modulus and on the norm of the CCB preimage. Since the required parameter inequalities are not verified here for the standard ML-DSA parameter sets, we do not invoke this further reduction. Instead, we treat CCB as an independent random-oracle-free computational assumption.

E Detailed HSwR Instantiation Guidance

This section provides concrete guidance for instantiating HSwR.

For Falcon, replacing the signing Gaussian with public Gaussian sampling cannot be justified by adding a negligible advantage term. Simulation replaces the secret-basis Gaussian sampling with sampling a short pair (s_1, s_2) from the corresponding public Gaussian distribution without the secret basis. The Falcon specification [PFH+22] instead multiplies the resulting advantage by a Rényi-divergence factor [PFH+22, Sec. 2.6]. Strong unforgeability additionally requires SPR for the NTRU map. Recent work characterizes this requirement through a second-preimage variant of inhomogeneous SIS, whose hardness relative to standard SIS remains an additional assumption [GJK24a]. We therefore do not claim SPR for Falcon.

For salt-UOV and QR-UOV, the public sampler draws uniformly from the domain. The modified-UOV analysis proves that retrying the public salt gives the same preimage distribution and applies the result to QR-UOV [KX24, Prop. 5.3]; salt-UOV has this retry structure by definition [BCD+24, App. A]. We use salt-UOV because the recommended Round 2 procedure instead retries vinegar variables. The cited argument is statistical: for salt-UOV and QR-UOV, the public uniform-domain sampler has the same preimage distribution as the relevant signing experiment. Since this sampler directly outputs a domain element and does not use a trapdoor, it satisfies universal sampling correctness for every well-formed public key. Moreover, to the best of our knowledge, SPR for MQ-based constructions such as salt-UOV, QR-UOV, and MAYO has not been studied in sufficient depth. Accordingly, the corresponding rows of Table 2 support only the ordinary-unforgeability and anonymity interfaces, and we do not claim complete strong-unforgeability instantiations for these schemes.

MAYO follows the same idea, but full-rank failures limit the signing samples covered by the statistical argument. Its sampled vinegar vectors determine the matrix passed to $\mathsf{SampleSolution}$. If the matrix has full rank, the output is uniform in the corresponding affine solution space; if rank deficient, signing resamples them and uniform-domain simulation no longer describes the output. If τ bounds the failure probability of the first sample in each signing query, then a union bound over at most Q_{Sign} ring-signing queries gives a total

failure probability of at most $Q_{\mathsf{Sign}}\tau$. The cited reduction therefore requires $Q_{\mathsf{Sign}}\tau < 1$ [KX24, Prop. 5.5]. For the Round 2 parameters, the specification gives a restart bound of approximately 2^{-12} for MAYO1, MAYO3, and MAYO5 and 2^{-20} for MAYO2 [BCC+24, Sec. 2.1.2, Sec. 2.1.5, Sec. 5.2]. We therefore state MAYO only as a conditional instantiation.

F Missing Proofs

F.1 Canceling Reprogramming in Theorem 5.1

Lemma F.1 (Canceling rejected reprogramming in Theorem 5.1). *In the setting of Theorem 5.1, we have*

$$\begin{aligned}|\Pr[\mathsf{G}_1 : W_1] - \Pr[\mathsf{G}_2 : W_2]| \le{}& 5n(Q'_{\mathsf{Sign}})^2 2^{-\alpha} + Q'_{\mathsf{Sign}} Q_{\mathsf{H}} 2^{-\alpha} + 2Q'_{\mathsf{Sign}}\sqrt{Q_{\mathsf{H}} 2^{-\alpha}} \\ &+ (n-1)\left(5n(Q'_{\mathsf{Sign}})^2 2^{-\alpha'} + Q'_{\mathsf{Sign}} Q_{\mathsf{H}} 2^{-\alpha'} + 2Q'_{\mathsf{Sign}}\sqrt{Q_{\mathsf{H}} 2^{-\alpha'}}\right) + N\delta.\end{aligned}$$

Proof. Following the proof strategy of Fallahpour, Fehr, and Huang [FFH25, Prop. 1], we proceed by a sequence of abort-cancellation hybrids. We first exclude honest keys that do not satisfy the required real-commitment min-entropy, incurring a failure probability of at most $N\delta$. We then change the signer-position reprogramming rule one signing attempt at a time through Q'_{Sign} hybrids and bound each transition using their abort-cancellation argument. Finally, we repeat the same argument at each of the at most $n-1$ simulated positions, using simulated-commitment min-entropy in place of real-commitment min-entropy. Summing these hybrid losses and the initial bad-key probability gives the claimed bound.

Estimate for one hybrid hop. At each ring position, change the reprogramming rule one signing attempt at a time. We first bound one hop differing at one position and attempt.

For the signer position, let $\mathsf{Adv}(Q_r, Q_w)$ be the maximal distinguishing advantage between the following two games. The first game reprograms H at $(j+1, R, m, a_j)$ before checking whether $z_j = \perp$ and leaves this reprogramming in place even if $z_j = \perp$. The second game reprograms H at this point only if $z_j \neq \perp$. The distinguisher makes at most Q_r quantum queries to H and an expected Q_w classical reprogramming queries to H. We show that

$$\mathsf{Adv}(Q_r, Q_w) \le (5Q_w + Q_r)2^{-\alpha} + 2\sqrt{Q_r 2^{-\alpha}}. \tag{11}$$

The abort-cancellation argument of [FFH25, Prop. 1] bounds the difference between the two games by decomposing it into two adaptive-reprogramming hops and one classical collision event. The games are identical when $z_j \neq \perp$, so only an aborting attempt with $z_j = \perp$ can distinguish them. In that case, the commitment at which the oracle was reprogrammed is not revealed. If the probability of guessing this commitment is at most ϵ, then each of the two adaptive-reprogramming hops contributes at most

$$\left(2Q_w + \frac{Q_r}{2}\right)\epsilon + \sqrt{Q_r \epsilon}$$

by Lemma 3.1. The probability that one of the at most Q_w classical write queries hits the same hidden commitment is at most $Q_w\epsilon$. Summing these three contributions gives

$$2\left(\left(2Q_w + \frac{Q_r}{2}\right)\epsilon + \sqrt{Q_r\epsilon}\right) + Q_w\epsilon = (5Q_w + Q_r)\epsilon + 2\sqrt{Q_r\epsilon}.$$

In our signer-position hybrid, the reprogrammed oracle input is $(j+1, R, m, a_j)$. Once $(j+1, R, m)$ is fixed, guessing this input amounts to guessing the commitment a_j, whose probability is at most $2^{-\alpha}$ by the assumed commitment min-entropy. Moreover, if $z_j = \perp$, the signing attempt is discarded without revealing a_j, whereas if $z_j \neq \perp$, the two games behave identically. Proposition 1 therefore applies with $\epsilon = 2^{-\alpha}$ and yields (11).

Summing the signer-position hybrids. Apply (11) one signing attempt at a time, keeping simulated-position rules fixed. For $h \in \{0, \ldots, Q'_{\mathsf{Sign}}\}$, let Hyb_h reprogram at the signer position only when $z_j \neq \perp$ in the first h attempted signing iterations, and reprogram there regardless of z_j in the remaining iterations. Thus, Hyb_0 uses the reprogramming rule of Game 1 at the signer position in every iteration, whereas $\mathsf{Hyb}_{Q'_{\mathsf{Sign}}}$ uses the rule of Game 2 there in every iteration. We write W_h for the event that Hyb_h outputs 1.

For $h \in \{1, \ldots, Q'_{\mathsf{Sign}}\}$, Hyb_{h-1} and Hyb_h differ only at the signer position in attempt h. Hence, (11) applies to this pair of hybrids. The adversary makes at most Q_{H} quantum read queries, and the signing simulation performs at most nQ'_{Sign} classical reprogramming operations over the entire execution. We may therefore set $(Q_r, Q_w) = (Q_{\mathsf{H}}, nQ'_{\mathsf{Sign}})$. It follows that

$$\begin{aligned}\left|\Pr[\mathsf{Hyb}_{h-1} : W_{h-1}] - \Pr[\mathsf{Hyb}_h : W_h]\right| &\leq \mathsf{Adv}(Q_{\mathsf{H}}, nQ'_{\mathsf{Sign}}) \\ &\leq (5nQ'_{\mathsf{Sign}} + Q_{\mathsf{H}})2^{-\alpha} + 2\sqrt{Q_{\mathsf{H}}2^{-\alpha}}.\end{aligned}$$

Summing over $h = 1, \ldots, Q'_{\mathsf{Sign}}$ gives the total loss from changing the reprogramming rule at the signer position:

$$\left|\Pr[\mathsf{Hyb}_0 : W_0] - \Pr[\mathsf{Hyb}_{Q'_{\mathsf{Sign}}} : W_{Q'_{\mathsf{Sign}}}]\right| \leq Q'_{\mathsf{Sign}}\left((5nQ'_{\mathsf{Sign}} + Q_{\mathsf{H}})2^{-\alpha} + 2\sqrt{Q_{\mathsf{H}}2^{-\alpha}}\right). \tag{12}$$

Summing the simulated-position hybrids. Starting from $\mathsf{Hyb}_{Q'_{\mathsf{Sign}}}$, change each simulated position in turn. For fixed $i \neq j$, use Q'_{Sign} hybrids while holding the others fixed. Fallahpour, Fehr, and Huang's argument [FFH25, Prop. 1] applies to each adjacent pair. The point is now $(i+1, R, m, a_i)$, whose guessing probability is at most $2^{-\alpha'}$ by the simulated-commitment assumption. If $z_j = \perp$, the attempt is discarded and a_i remains hidden; otherwise the hybrids agree. Other positions are identical.

The query bounds are again $(Q_r, Q_w) = (Q_{\mathsf{H}}, nQ'_{\mathsf{Sign}})$. Therefore, the loss for one hybrid hop at position i is at most

$$(5nQ'_{\mathsf{Sign}} + Q_{\mathsf{H}})2^{-\alpha'} + 2\sqrt{Q_{\mathsf{H}}2^{-\alpha'}}.$$

Summing over the Q'_{Sign} attempted signing iterations gives

$$Q'_{\mathsf{Sign}}\left((5nQ'_{\mathsf{Sign}} + Q_{\mathsf{H}})2^{-\alpha'} + 2\sqrt{Q_{\mathsf{H}}2^{-\alpha'}}\right)$$

for each simulated position. Since there are at most $n-1$ simulated positions, their total contribution is at most

$$(n-1)Q'_{\mathsf{Sign}}\left((5nQ'_{\mathsf{Sign}} + Q_{\mathsf{H}})2^{-\alpha'} + 2\sqrt{Q_{\mathsf{H}}2^{-\alpha'}}\right). \tag{13}$$

After all simulated positions have been changed, the resulting game is G_2.

By the triangle inequality, add the signer-position bound (12), the simulated-position bound (13), and the min-entropy bad-event probability $N\delta$. This proves the lemma. □

F.2 Bounding Challenge Collision with Same Commitments in Theorem 6.1

Lemma F.2 (Bounding challenge collision with the same commitments in Theorem 6.1). *In the setting and notation of the proof of Theorem 6.1, there exists a QPT adversary $\mathcal{A}_{\mathrm{inv}}$ such that*

$$|\Pr[\mathsf{G}_4 : W_4] - \Pr[\mathsf{G}_5 : W_5]| \leq NQ_{\mathsf{Sign}}\left(\left(2(Q_{\mathsf{H}} + n) + 1\right)^2 + Q_{\mathsf{Sign}} - 1\right)\mathsf{Adv}^{\mathsf{INV}}_{\mathsf{TDF},\mathcal{A}_{\mathrm{inv}}}(\lambda).$$

The running time of $\mathcal{A}_{\mathrm{inv}}$ is polynomial in the running time of $\mathcal{A}$.

Proof. Whenever the Game 5 rejection event occurs, select the first tuple $(R^*, m^*, i, pk_i^*, \bar{c}_i, \bar{z}_i, \bar{a}_i)$ according to a fixed ordering such that $a_i^* = \bar{a}_i$ and $c_i^* \neq \bar{c}_i$, and consider the input (i, R^*, m^*, a_{i-1}^*) used to compute c_i^*. If this input is not contained in the signing-programming table, then $c_i^* = \mathsf{H}(i, R^*, m^*, a_{i-1}^*)$; we handle this case using Lemma 3.2. If the input is contained in the signing-programming table, the corresponding entry cannot come from the same recorded signing chain as the selected tuple, since this would imply $c_i^* = \bar{c}_i$. It must therefore have been supplied by another signing answer.

Predecessor input outside the table. Define a wrapper $\mathcal{A}'$ that runs $\mathcal{A}$ and simulates both RingSign and the random oracle seen by $\mathcal{A}$. When a RingSign query produces a simulated signing answer as in Game 4, $\mathcal{A}'$ does not modify H and instead records in the signing-programming table every input-output pair required for that answer. For each quantum random-oracle query of $\mathcal{A}$, the wrapper returns the recorded value on an input in the table and the value of H on every other input. Since the table has polynomial size, the wrapper implements this rule by reversibly checking the query input against the recorded inputs, without measuring it.

After $\mathcal{A}$ outputs a candidate forgery (R^*, m^*, σ^*), the wrapper computes its verification chain $(a_i^*, c_i^*)_i$ using the same rule. It selects the first recorded tuple satisfying the Game 5 condition. If no such tuple exists, or if its predecessor input is in the signing-programming table, the wrapper outputs $\perp$; otherwise, it outputs the predecessor input together with the forgery and the selected tuple. Define V to accept iff verification succeeds, its supplied oracle value is c_i^*, and the selected tuple satisfies $a_i^* = \bar{a}_i$ and $c_i^* \neq \bar{c}_i$. In every retained execution, $c_i^* = \mathsf{H}(i, R^*, m^*, a_{i-1}^*)$, so Lemma 3.2 is applied to a value supplied by H, rather than by a programming entry. The wrapper makes at most $Q_{\mathsf{H}} + n$ quantum queries to H: at most Q_{H} while answering $\mathcal{A}$'s queries and at most n while computing the verification chain.

Applying Lemma 3.2 to $\mathcal{A}'$ and V gives a two-stage simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that

$$\begin{aligned}
&\Pr\left[\mathsf{V}\big((i, R^*, m^*, a_{i-1}^*), \mathsf{H}(i, R^*, m^*, a_{i-1}^*), \mathsf{out}^*\big) : ((i, R^*, m^*, a_{i-1}^*), \mathsf{out}^*) \leftarrow \mathcal{A}'^{|\mathsf{H}\rangle}\right] \\
&\quad \leq \big(2(Q_{\mathsf{H}} + n) + 1\big)^2 \Pr\left[\mathsf{V}\big((i, R^*, m^*, a_{i-1}^*), \theta, \mathsf{out}^*\big) : \begin{array}{r} \theta \leftarrow \mathcal{C}, \\ (i, R^*, m^*, a_{i-1}^*) \leftarrow \mathcal{S}_1^{|\mathsf{H}\rangle}(), \\ \mathsf{out}^* \leftarrow \mathcal{S}_2^{|\mathsf{H}'\rangle}(\theta) \end{array}\right],
\end{aligned} \tag{14}$$

where $\mathsf{H}' = \mathsf{H}[(i, R^*, m^*, a_{i-1}^*) \mapsto \theta]$; the notation suppresses the wrapper's signing-oracle simulation and its signing-programming table. The simulator uses the same table in both stages, together with H in the first stage and H' in the second. Because the measured predecessor input is not in the table, the reprogramming changes the value returned there to θ, while the table continues to supply all values required for simulated signing answers.

The INV adversary guesses one of the Q_{Sign} signing answers and one of the N honest keys. At the corresponding position it samples $\bar{z}_i \leftarrow \mathsf{Samp}(pk_i^*)$ and $\bar{c}_i \leftarrow \mathcal{C}$, sets $\bar{a}_i = \mathsf{F}(pk_i^*, \bar{z}_i) \circ \bar{c}_i^{-1}$, and completes the simulated signing record as in Game 3. Given an INV target y, it sets $\theta = \bar{a}_i^{-1} \circ y$, which is uniform in $\mathcal{C}$, and supplies θ to the measure-and-reprogram simulator. If V accepts, then $c_i^* = \theta$ and $a_i^* = \bar{a}_i$, and hence $\mathsf{F}(pk_i^*, z_i^*) = a_i^* \circ c_i^* = \bar{a}_i \circ \theta = y$. This case incurs loss $NQ_{\mathsf{Sign}}(2(Q_{\mathsf{H}} + n) + 1)^2$.

Predecessor input inside the table. This case is handled directly, without measure-and-reprogramming. The INV adversary guesses the signing answer containing the selected tuple, one of the N honest keys, and the distinct signing answer supplying the programming entry, for a total factor $NQ_{\mathsf{Sign}}(Q_{\mathsf{Sign}} - 1)$. At the selected position it samples $\bar{z}_i \leftarrow \mathsf{Samp}(pk_i^*)$ and $\bar{c}_i \leftarrow \mathcal{C}$, sets $\bar{a}_i = \mathsf{F}(pk_i^*, \bar{z}_i) \circ \bar{c}_i^{-1}$, and completes that signing record as in Game 3. Given an INV target y, it sets $\theta = \bar{a}_i^{-1} \circ y$ and uses θ as the incoming challenge at the corresponding position of the distinct signing answer. Since θ is uniform, this preserves the signing distribution. On correct guesses, the programming entry gives $c_i^* = \theta$, while the selected Game 5 record gives $a_i^* = \bar{a}_i$, so $\mathsf{F}(pk_i^*, z_i^*) = a_i^* \circ c_i^* = \bar{a}_i \circ \theta = y$. Thus, this case incurs loss $NQ_{\mathsf{Sign}}(Q_{\mathsf{Sign}} - 1)$.

Adding the losses of the two disjoint cases proves the claimed bound. □

F.3 Proof of Corollary 7.1

Corollary F.1 (Corollary 7.1 - restated). *Let* Ring-ML-DSA *be the original Ring-ML-DSA scheme and let* sRing-ML-DSA *be the simplified scheme described in Appendix B. For $Q \geq 0$, define*

$$\Lambda_{\mathrm{rnd}}(Q) \coloneqq \frac{Q(Q-1)}{2^{512}} + \frac{Q(Q-1)}{2^{256}} + \frac{\sqrt{Q_{\mathsf{H}}(Q_{\mathsf{H}}+1)Q}}{2^{127}} + \frac{\sqrt{Q_{\mathsf{H}}(Q_{\mathsf{H}}+1)Q}}{2^{255}}.$$

For any QPT adversary $\mathcal{A}$ making at most Q_{Sign} ring-signing queries, suppose that every signing execution samples rnd independently and uniformly from $\{0,1\}^{256}$ and that the complete experiment makes at most Q_{H}

reads to the single oracle H *across all call sites. Then, there is a QPT adversary* $\mathcal{A}'$ *for the corresponding experiment such that*

$$\mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{Ring\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}'}(\lambda) + \Lambda_{\mathrm{rnd}}(Q_{\mathsf{Sign}}),$$
$$\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{Ring\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}'}(\lambda) + \Lambda_{\mathrm{rnd}}(Q_{\mathsf{Sign}}+1).$$

The running time of $\mathcal{A}'$ *is polynomial in the running time of* $\mathcal{A}$ *and in the cost of simulating at most* $Q_{\mathsf{Sign}}+1$ *signatures and answering at most* Q_{H} *random-oracle queries.*

Proof. The proof reaches the simplified signing distribution by randomizing $\rho'' = \mathsf{H}(K\|rnd\|\mu, 64)$ and its $\mathsf{ExpandMask}$ outputs. It leaves the key-generation oracle calls and all values derived from them unchanged throughout. The two experiments use identical hops with different Q. We set $Q = Q_{\mathsf{Sign}}$ for the sEUF-CRA experiment and $Q = Q_{\mathsf{Sign}} + 1$ for the ke-ANON experiment, where the extra signature is the anonymity challenge signature. Let W_i be the event that $\mathcal{A}$ wins the relevant experiment in Game i, denoted by G_i.

Game 0. This is the original experiment for $\mathsf{Ring\text{-}ML\text{-}DSA}$.

Game 1. Game 1 records every internal nonce rnd and aborts if the same nonce is sampled in two different signatures. Since at most Q independent uniform 256-bit nonces are sampled, the birthday bound gives

$$|\Pr[\mathsf{G}_0 : W_0] - \Pr[\mathsf{G}_1 : W_1]| \le \frac{\binom{Q}{2}}{2^{256}} = \frac{Q(Q-1)}{2^{257}}.$$

Outside this event all inputs (K_j, rnd, μ) used for nonce derivation are distinct, even when the same signer signs the same ring and message more than once. This replaces the distinct message/nonce-pair restriction of [KX25, Theorem 2].

Game 2. Game 2 punctures the exact H input used to derive the masking seed in each of the at most Q signatures. For the q-th signature, made by signer j, this point is

$$(K_j, rnd^{(q)}, \mu^{(q)}).$$

Here, puncturing follows the semi-classical O2H lemma [AHU19]. Even if signer j has been corrupted and K_j is known, $rnd^{(q)} \leftarrow \{0,1\}^{256}$ remains hidden and supplies 256 bits of entropy. Outside Game 1's collision event, a measured query hits one of at most Q punctured points with probability at most $Q/2^{256}$. Applying the same semi-classical O2H accounting as in [KX25, Lemma 15] gives

$$|\Pr[\mathsf{G}_1 : W_1] - \Pr[\mathsf{G}_2 : W_2]| \le \sqrt{\frac{4Q_{\mathsf{H}}(Q_{\mathsf{H}}+1)Q}{2^{256}}} = \frac{\sqrt{Q_{\mathsf{H}}(Q_{\mathsf{H}}+1)Q}}{2^{127}}.$$

Game 3. Game 3 samples ρ'' uniformly instead of generating it via H. The corresponding inputs were punctured in G_2, so only the signing oracle's output distribution exposes their values. The inputs are distinct, so the replacements are independent uniform values in both games. Therefore, $\Pr[\mathsf{G}_3 : W_3] = \Pr[\mathsf{G}_2 : W_2]$ holds.

Game 4. Game 4 records the mask-expansion inputs (ρ'', κ) and aborts if an input is reused. The counter discipline prevents reuse within one signature, so the bad event requires two of the Q signatures to use the same ρ''. By the seed-collision bound of [KX25, Lemma 16],

$$|\Pr[\mathsf{G}_3 : W_3] - \Pr[\mathsf{G}_4 : W_4]| \le \frac{Q(Q-1)}{2^{513}}.$$

Game 5. Game 5 samples Q independent seeds $\{\rho''^{(q)} : q \in [Q]\}$ in advance, uses $\rho''^{(q)}$ in the q-th signature, and punctures the associated mask-expansion inputs. Since Game 4 excludes repeated seeds, the punctured set is

$$\{\rho''^{(q)} : q \in [Q]\} \times \{0, \ldots, 2^{16} - 1\}.$$

The bad event is that a measured H query hits this set, whose density is at most $Q/2^{512}$. This is the mask-expansion puncturing hop of [KX25, Lemma 17]. Applying the semi-classical O2H lemma gives

$$|\Pr[\mathsf{G}_4 : W_4] - \Pr[\mathsf{G}_5 : W_5]| \leq \frac{\sqrt{Q_{\mathsf{H}}(Q_{\mathsf{H}}+1)Q}}{2^{255}}.$$

Game 6. Game 6 directly samples values from punctured mask-expansion calls. It samples the real masking vector $\boldsymbol{y}_j$ and every simulated candidate response $\boldsymbol{z}_i$ from $\mathcal{Y}$. The local retry loops, the genuine-response restart rule, and the challenge chain are unchanged. After the puncturing step, this output replacement introduces no additional loss: $\Pr[\mathsf{G}_6 : W_6] = \Pr[\mathsf{G}_5 : W_5]$.

Game 7. Game 7 removes the two auxiliary tables containing the internal nonces and the mask-expansion inputs. The bad events are the same collisions controlled in Games 1 and 4. The table-removal hop for the mask-expansion inputs follows [KX25, Lemma 16]; together with the birthday bound of Game 1, it gives

$$|\Pr[\mathsf{G}_6 : W_6] - \Pr[\mathsf{G}_7 : W_7]| \leq \frac{Q(Q-1)}{2^{513}} + \frac{Q(Q-1)}{2^{257}}.$$

The signing oracle of Game 7 is exactly that of sRing-ML-DSA. The winning predicate, including strong freshness for sEUF-CRA, is unchanged. Thus, the final sEUF-CRA and ke-ANON games are those of sRing-ML-DSA. Summing the losses gives the corollary. □

F.4 Proof of Corollary 7.2

Corollary F.2 (Corollary 7.2 - restated). *Let $\mathsf{ID}_{\mathrm{ML}}$ be the ML-DSA identification protocol underlying the above construction, where the ring public key is $pk' = \mathsf{PkEncode}'(pk, \boldsymbol{t}_0)$, and let $\mathsf{Sim}_{\mathrm{ML}}$ be its perfect ke-acHVZK simulator. For any QPT adversary $\mathcal{A}$ against* sRing-ML-DSA *making at most Q_{Sign} ring-signing queries for rings of size at most n, let N be the number of honestly generated keys in the security experiment, and suppose that the complete experiment makes at most Q_{H} reads to the single oracle H across all call sites. There exists a QPT adversary $\mathcal{A}_{\mathrm{nra}}$ such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}_{\mathrm{nra}}}(\lambda) + \Gamma_{\mathrm{ML}}(Q_{\mathsf{Sign}}) + \Delta_{\mathrm{bind}}.$$

Here Γ_{ML} is defined in (4) and Δ_{bind} in (5). If, in addition, $\mathsf{ID}_{\mathrm{ML}}$ satisfies CUR and CSS, then there exist QPT adversaries $\mathcal{A}_{\mathrm{cur}}$ and $\mathcal{A}_{\mathrm{css}}$ such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{sEUF\text{-}CRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \leq{}& \mathsf{Adv}^{\mathsf{EUF\text{-}NRA}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}_{\mathrm{nra}}}(\lambda) + N\,\mathsf{Adv}^{\mathsf{CUR}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{cur}}}(\lambda) + N\,\mathsf{Adv}^{\mathsf{CSS}}_{\mathsf{ID}_{\mathrm{ML}},\mathcal{A}_{\mathrm{css}}}(\lambda)\\ &+ \frac{632}{|\mathcal{C}|}(Q_{\mathsf{H}} + nQ_{\mathsf{Sign}} + n + 1)^3 + \Gamma_{\mathrm{ML}}(Q_{\mathsf{Sign}}) + \Delta_{\mathrm{bind}}.\end{aligned}$$

Moreover,

$$\mathsf{Adv}^{\mathsf{ke\text{-}ANON}}_{\mathsf{sRing\text{-}ML\text{-}DSA},\mathcal{A}}(\lambda) \leq \Gamma_{\mathrm{ML}}(Q_{\mathsf{Sign}} + 1) + \Delta_{\mathrm{bind}}.$$

The running times of $\mathcal{A}_{\mathrm{nra}}$, $\mathcal{A}_{\mathrm{cur}}$, and $\mathcal{A}_{\mathrm{css}}$ are polynomial in the running time of $\mathcal{A}$ and include at most $O(nT(Q_{\mathsf{Sign}} + 1))$ transcript simulations and oracle-programming operations.

Proof. To apply Theorems 5.1 and 5.2, the proof compares the challenge chain of Ring-ML-DSA with the position-indexed chain covered by the generic FSwA theorems. In the generic construction, the challenge at source position i is generated from $(i+1, R, M', a_i)$, whereas Ring-ML-DSA evaluates the shared H at

$$\mu \| \mathsf{IntegerToBytes}(i+1, 2) \| \mathsf{w1Encode}(\boldsymbol{w}_{1,i}), \qquad \mu = \mathsf{H}\big(\mathsf{BytesToBits}(tr_R) \| M', 64\big).$$

Here, μ represents the ring-message context (R, M'), the two-byte encoding $\mathsf{IntegerToBytes}(i+1, 2)$ represents the position index $i+1$, and $\mathsf{w1Encode}(\boldsymbol{w}_{1,i})$ represents the reconstructed commitment a_i. Since H takes μ instead of the message directly, we first show that distinct ring-message contexts yield distinct values of μ except with the stated random-oracle collision probability. We then show that the fixed-length position encoding together with the canonical commitment encoding makes the concrete challenge input faithfully represent the generic tuple $(i+1, R, M', a_i)$. Finally, although all Ring-ML-DSA operations use the same random function H, the generic reprogramming remains valid because a challenge reprogramming fixes only the required prefix of the random-oracle response at the programmed input, while every other use of H, including any query to that same input, is treated as an ordinary quantum random-oracle query and is already covered by the global query bound Q_{H}.

We first establish that μ uniquely identifies the ring-message context. Let Bad_{tr} occur if two distinct ring encodings evaluated in the security experiment produce the same value tr_R, and let Bad_{μ} occur if two distinct inputs $\mathsf{BytesToBits}(tr_R)\|M'$ produce the same value μ. Every occurrence of $\mathsf{Bad}_{tr} \vee \mathsf{Bad}_{\mu}$ yields two distinct inputs with the same output under H. Applying Lemma 3.3 to a finder that records all relevant inputs therefore gives

$$\Pr[\mathsf{Bad}_{tr} \vee \mathsf{Bad}_{\mu}] \leq \Delta_{\mathrm{bind}} := \frac{632(Q_{\mathsf{H}}+1)^3}{2^{512}}.$$

Outside these events, distinct rings have distinct values tr_R; moreover, since tr_R has fixed length, two distinct values of (R, M') give distinct inputs $\mathsf{BytesToBits}(tr_R)\|M'$ and hence distinct values μ. Thus, except with probability Δ_{bind}, the value μ uniquely determines every ring-message context reached in the security experiment.

We next identify the concrete challenge inputs with those of the generic position-indexed chain. For a fixed ring-message context, the challenge update at position i is evaluated at $\mu\|\mathsf{IntegerToBytes}(i+1, 2)\|\mathsf{w1Encode}(\boldsymbol{w}_{1,i})$. The canonical-ring check guarantees $n < 2^{16}$, so $\mathsf{IntegerToBytes}(i+1, 2)$ uniquely identifies the source position. Moreover, outside $\mathsf{Bad}_{tr} \vee \mathsf{Bad}_{\mu}$, the value μ uniquely identifies the ring-message context, while $\mathsf{w1Encode}(\boldsymbol{w}_{1,i})$ is the canonical encoding of the reconstructed commitment at that position. Consequently, the three components of challenge inputs respectively represent the context, position, and commitment components of the generic input $(i+1, R, M', a_i)$. Hence, challenge inputs from distinct contexts or positions cannot coincide, and, within a fixed context and position, equality of concrete challenge inputs is equivalent to equality of the canonically encoded reconstructed commitments. Thus the concrete Ring-ML-DSA challenge updates realize the position-indexed chain required by the generic FSwA theorems.

It remains to justify that the same random function H may be shared by the challenge computation and the other Ring-ML-DSA operations. View the response of H on each input as a sufficiently long uniformly random string, with $\mathsf{H}(x, \ell)$ returning its first ℓ bytes. Programming a challenge input therefore amounts to fixing the required $\lambda/4$-byte prefix of this response and sampling the remaining suffix uniformly and independently, which has exactly the distribution required by the generic reprogramming argument. All calls made by key expansion, ring and message hashing, nonce derivation, $\mathsf{SampleInBall}$, $\mathsf{ExpandA}$, and $\mathsf{ExpandMask}$ remain ordinary reads to the same random function. In particular, no separate domain-separation assumption between these calls and the challenge computation is required: if another operation queries the same input as a programmed challenge point, it simply observes the programmed random-oracle value at that point. Any adversarial query that queries or predicts a programmed challenge input is already included in the global query count Q_{H} used by the adaptive-reprogramming and abort-cancellation bounds. Hence, subject to the entropy and conditional-abort assumptions of Corollary 7.2, the use of one shared H is compatible with the generic FSwA proof. □