Signatures with Memory-Tight Security in the Quantum Random Oracle Model [*]

草川 恵太 (Keita Xagawa)

Technology Innovation Institute, UAE
keita.xagawa@tii.ae

June 7, 2024

Abstract. Memory tightness of reductions in cryptography, in addition to the standard tightness related to advantage and running time, is important when the underlying problem can be solved efficiently with large memory, as discussed in Auerbach, Cash, Fersch, and Kiltz (CRYPTO 2017). Diemert, Gellert, Jager, and Lyu (ASIACRYPT 2021) and Ghoshal, Ghosal, Jaeger, and Tessaro (EUROCRYPT 2022) gave memory-tight proofs for the multi-challenge security of digital signatures in the random oracle model.

This paper studies the memory-tight reductions for *post-quantum* signature schemes in the *quantum* random oracle model. Concretely, we show that signature schemes from lossy identification are multi-challenge secure in the quantum random oracle model via memory-tight reductions. Moreover, we show that the signature schemes from lossy identification achieve more enhanced securities considering *quantum* signing oracles proposed by Boneh and Zhandry (CRYPTO 2013) and Alagic, Majenz, Russel, and Song (EUROCRYPT 2020). We additionally show that signature schemes from preimage-sampleable functions achieve those securities via memory-tight reductions.

Keywords: memory-tight reductions · signature · provable security · post-quantum cryptography · quantum random oracle model (QROM) · plus-one unforgeability · blind unforgeability

Table of Contents

[*] This article is based on an earlier article: Keita Xagawa: Signatures with Memory-Tight Security in the Quantum Random Oracle Model, EUROCRYPT2024, ©IACR 2024, https://doi.org/10.1007/978-3-031-58754-2_2

1 Introduction

Memory-tight reductions: Provable security in cryptography consists of reductions and assumptions; we assume the hardness of a computational problem, design a cryptographic scheme, and then make a reduction algorithm $\mathcal{R}$ to solve the underlying problem by using an adversary $\mathcal{A}$ breaking the security of the scheme. The tightness of the reduction is measured by how close the resources of $\mathcal{R}$ and $\mathcal{A}$ are, where resources are success probability, running time, the number of queries, etc. The tightness of the security reduction is essential because it impacts the parameters of the cryptographic schemes and, thus, the scheme's efficiency. (See e.g., [BR96, Cor00, KM07, CMS12, CKMS16].)

Auerbach, Cash, Fersch, and Kiltz [ACFK17] explicitly put forth *memory tightness* of the reduction from the view of *memory usage*.[1] This concept is important when the underlying computational problems are memory-sensitive; the problem can be solved efficiently with large memory. Examples of such problems are factoring, lattice problems, the lerning-parity-with-noise (LPN) problem, and the multi-collision problem of the hash function.

After that, memory-tight reductions have gathered much attention and become an active area of cryptography: Auerbach et al. [ACFK17] gave several techniques to make security proofs memory-tight.[2] Using those techniques, they gave a memory-tight reduction for the standard security, the existential unforgeability under chosen-message attacks (EUF-CMA security), of RSA-FDH [RSA78, BR96] in the random oracle model (ROM) [BR93]. Diemert, Gellert, Jager, and Lyu [DGJL21] studied memory-tight proofs of the *strong* existential unforgeability under chosen-message attacks in the *multi-challenge* setting (msEUF-CMA security), where an adversary can submit multiple attempts of forgery and it wins if one of them is a 'new' forgery. They gave memory-tight reductions in the ROM for msEUF-CMA security of RSA-PFDH [BR96], the BLS signature [BLS01], and RFS-LID [AFLT12, FS87], where RFS denotes the Fiat-Shamir transform with random nonces and LID denotes a lossy identification. Ghoshal, Ghosal, Jaeger, and Tessaro [GGJT22] also gave a memory-tight proof of the msEUF-CMA security of RSA-PFDH in the multi-challenge setting in the ROM. Bhattacharyya [Bha20] and Jaeger and Kumar [JK22] gave memory-tight proofs for the security of key encapsulation mechanisms based on the variants of the Diffie-Hellman problem in the ROM. There are studies for symmetric-key cryptography, e.g., [Din20, GJT20, GGJT22], and the lower bound of memory usage of black-box reductions [ACFK17, WMHT18, GT20, GJT20].

Post-quantum signatures and quantum random oracle model: Post-quantum signatures are an emerging area of cryptography as NIST had run the standardization of PQC and selected three post-quantum signatures (Falcon, Dilithium, and SPHINCS+) [AAC$^+$22] and they started the standardization of additional signature schemes. The security of those post-quantum signatures is proven in the *quantum random oracle model (QROM)* [BDF$^+$11], in which an adversary can make *quantum* queries to a random oracle. As far as we surveyed, the sEUF-CMA security proof for PSF-DFDH in Boneh et al. [BDF$^+$11] is only one memory-tight proof for signature in the QROM, where PSF is preimage-sampleable functions [GPV08] and DFDH is FDH derandomized by a pseudo-random function (PRF). The following natural question arises:

> Can we construct memory-tight reductions for the msEUF-CMA security of post-quantum PSF-based signatures in the QROM?

In addition, the memory-tight security proof of the msEUF-CMA security of the signature scheme from LID in Diemert et al. [DGJL21] assumes that the underlying LID is perfectly correct and commitment-recoverable[3] and has statistical honest-verifier zero-knowledge (HVZK) with a special simulator[4] and perfect unique response property[5]. Moreover, their proof is considered in the ROM. Thus, it is natural to ask the following question:

> Can we construct memory-tight reductions for the msEUF-CMA security of post-quantum LID-based signatures in the QROM and eliminate the conditions on the underlying LID?

Quantum signing oracles: Furthermore, there are extended security models for signature schemes in the quantum setting by giving *quantum access* to the signing oracle. The first one is proposed by Boneh and Zhandry and dubbed EUF-qCMA security [BZ13b]. But, we call it plus-one unforgeability (PO security in

[1] 2023-11-20: Bernstein [Ber11] considered a memory-bounded adversary/reduction. There might be other studies considering memory-bounded adversary/reduction.

[2] 2023-11-20: Bernstein [Ber11] proposed techniques (e.g., use PRF instead of lazy sampling) to reduce the memory of the adversary simulating the random oracles.

[3] The verification algorithm taking a transcript (w, c, z) computes commitment w' from challenge c and response z and accepts if and only if $w = w'$.

[4] Their deterministic simulator takes a challenge and a response chosen uniformly at random and outputs a commitment.

[5] For any (vk, w, c), where vk is honestly generated public key, there is at most one response z that makes the verifier accepting. See their proof of the third claim in [DGJL21, Appendix D, ePrint]

short) following [AMRS20], because, in the security game, an adversary can access the signing oracle with q quantum queries and is required to output $q + 1$ distinct valid message/signature pairs. The other is proposed by Alagic, Majenz, Russell, and Song [AMRS20] and dubbed (strong) blind unforgeability (BU/sBU security in short). In the security game, an adversary can access the signing oracle with quantum queries, while some signatures are blinded if the corresponding messages are in a filter. The adversary is required to output a valid signature on a filtered message. Doosti, Delavar, Kashefi, and Arapinis [DDKA21] also gave parametrized security definitions using quantum signing oracles and showed that some of their definition are equivalent to the blind unforgeability.

Boneh and Zhandry [BZ13b] showed that the Lamport one-time signature and the Merkle signature are one-time PO-secure and PO-secure in the standard model, respectively. They further showed that some weakly-secure signature schemes under classical chosen message attacks can be converted into PO-secure signature schemes. They also directly showed that PSF-DFDH is PO-secure in the QROM. Chatterjee, Garg, Hajiabadi, Khurana, Liang, Malavolta, Pandey, and Shiehian [CGH$^+$21] defined a PO-like security of ring signature and proposed a ring signature scheme satisfying their security notion. Chatterjee, Chung, Liang, and Malavolta [CCLM22] showed that PSF-DFDH is BU-secure in the QROM and their lattice-based signature is BU-secure in the standard model. They also extended a BU-like security of ring signature and proposed a ring signature satisfying their BU-like security. Majenz, Manfouo, and Ozols [MMO21] showed that the Lamport one-time signature and the Winternitz one-time signature are BU-secure in the QROM by extending an argument in Alagic et al. [AMRS20]. Yuan, Tibouchi, and Abe [YTA23] showed that a variant of SPHINCS+ is PO-secure in the QROM.

To the best of the authors' knowledge, there is no memory-tight proof for such enhanced securities for post-quantum signatures based on PSF and LID. Our third question is:

> Can we construct memory-tight reductions for those advanced securities (PO, BU, and sBU) of post-quantum signatures based on PSF and LID in the QROM?

1.1 Contributions

We affirmatively answer those three questions: The main contributions of this paper are four-fold. First, we give a *memory-tight* MSEUF-CMA security proof for LID-based signature schemes. We remove the constraints on the underlying LID scheme as much as possible, and we can employ lattice-based LID schemes with imperfect correctness, say, Dilitihium-QROM [KLS18, DFPS23][6] and G+G [DPS23]. Second, we extend the MSEUF-CMA security proof into (memory-tight) PO and sBU security proofs for LID-based signature schemes. Those are the first PO and sBU security proof of LID-based signature schemes. Third, we modify the existing sEUF-CMA, PO, and BU security proofs for PSF-based signature schemes into memory-tight MSEUF-CMA, PO, and sBU security proofs. Fourth, we pointed out a gap between BU security and PO security.

New memory-tight MSEUF-CMA security proofs for LID-based signatures: We will give *memory-tight* MSEUF-CMA, PO, and sBU security proofs for LID-based signature schemes. Our starting point is the MSEUF-CMA security proof, and we extend it into PO and sBU security proofs.

The MSEUF-CMA *security proof for LID-based signatures:* We give a memory-tight MSEUF-CMA security proof for RFS-LID in the QROM, where LID can be *imperfectly* correct and not commitment-recoverable and can have *ordinal* statistical HVZK and *computational unique response* (CUR) property. As Diemert et al. [DGJL21], we first show the MSEUF-CMA1 security of FS-LID with memory-tight reduction, where CMA1 denotes chosen-message attacks in the *one-siganture-per-message* setting [KLS18] and FS denotes the Fiat-Shamir transform *with bounded aborts* [FS87, Lyu09]; we then obtain a memory-tight MSEUF-CMA security proof for RFS-LID by using a lemma in [DGJL21].

Our core contribution is a new memory-tight security proof of the MSEUF-CMA1 security of FS-LID. We carefully merge (and correct) the memory-tight MSEUF-CMA1 security proof in the ROM by Diemert et al. [DGJL21] and the memory-loose sEUF-CMA1 security proof in the QROM by Devevey, Fallahpour, Passelègue, and Stehlé [DFPS23], where the latter is a correction of the history-free programming proof in Abdalla, Fouque, Lyubashevsky, and Tibouchi [AFLT12] and Kiltz, Lyubashevsky, and Schaffner [KLS18].

Let us briefly remind the Fiat-Shamir with aborts applied to the LID scheme. Let w, c, and z denote a commitment, a challenge, and a response of the underlying LID scheme, respectively. On a message m, the signer generates a commitment w and computes challenge $c = \mathsf{H}(m, w)$, where H is a random oracle, and response z until $z \neq \bot$, and outputs a signature (w, z). The verifier verifies the transcript (w, c, z) via the LID's verification algorithm by computing $c = \mathsf{H}(m, w)$.

Our proof is summarized as follows: We first derandomize the signing oracle by using the random function. We next exclude the event that the signing oracle fails to produce a valid signature on the submitted message.

[6] We need a statistical HVZK simulator in [DFPS23, Section 4] instead of a non-aborting HVZK simulator in [KLS18].

This exclusion is required to invoke the CUR property of LID correctly. We then replace the winning condition that the adversary outputs a *new* pair of a message m^* and a valid signature (w^*, z^*) with the condition that the adversary's signature (w^*, z^*) on m^* is different from the signature $(\tilde{w}, \tilde{z})$ the signing oracle produces on m^* as Diemert et al. [DGJL21]. This modification allows us to remove the list containing a pair of messages queried by the adversary and signatures produced by the signing oracle and makes reductions memory-tight. In order to analyze the effect of this replacement, we will need to analyze the min-entropy of $\tilde{w}$ produced by the signing oracle carefully for imperfect-correct LID because the random oracle *leaks* the information of $\tilde{w}$. After those modifications, we follow the proof of Devevey et al. [DFPS23] to eliminate some events and simulate the signing oracle without the signing key. Before replacing the real verification key with the lossy one, we exclude the event that w^* in the adversary's signature is equivalent to $\tilde{w}$ in the signature produced by the signing oracle. This event cannot happen if LID has the CUR property. For the details, see Section 4. We then replace the verification key with a lossy one. In the final game, the adversary cannot win [AFLT12, KLS18]. While merging the proofs carefully and removing the constraints are technical contributions, we have two additional technical byproducts discussed below.

1: Flaw in the previous sEUF-CMA1 *proofs in the QROM:* We found a flaw related to the CUR property in the previous sEUF-CMA1 proof in the QROM [DFPS23]. Roughly speaking, to reduce the sEUF-CMA1 security to the EUF-NMA security of the signature, we want the adversary to output a pair m^* and (w^*, z^*) such that *we do not program the random oracle at* (m^*, w^*). In order to do so, we exclude the event that $w^* = \tilde{w}$ where $\tilde{w}$ is a part of the signature $(\tilde{w}, \tilde{z})$ produced by the signing oracle on m^* and $(m^*, \tilde{w})$ is programmed.
Devevey et al. [DFPS23] and Barbosa et al. [BBD$^+$23] pointed out that programming the random oracle only on succeeding signatures introduces a bias on the distribution of the signing oracle and the random oracle and the existing proofs do not care about the bias. To fix this error in the history-free programming approach [AFLT12, KLS18], Devevey et al. [DFPS23] programmed the random oracle at $(m, \tilde{w}_i)$, where $\tilde{w}_i$ is the commitment for the i-th signer's attempt on the message m. Unfortunately, they ignored the fact that the adversary would output a message m^* and a signature (w^*, z^*) with $w^* = \tilde{w}$. To fix this problem, we exclude the event that $w^* = \tilde{w}_i$ for some i. We are interested especially in the case of $w^* = \tilde{w}$. To make a reduction to the CUR property, both the transcript $(\tilde{w}, \tilde{c}, \tilde{z})$ generated by the signing oracle and $(\tilde{w}, \tilde{c}, z^*)$ generated by the adversary's signature are valid. To ensure the validity of $(\tilde{w}, \tilde{c}, \tilde{z})$ generated by the signing oracle, we exclude the event that the signing oracle fails to output a valid signature. Thus, our security bound involves the term related to correctness.

2: The divergence HVZK case: Some LID-based signatures based on lattices employ *divergence* HVZK instead of statistical HZVK to achieve a smaller signature size. See, e.g., [dPRS23, DPS23]. Roughly speaking, statistical HVZK requires that the real transcript and the simulated one are statistically close, while divergence HVZK requires that the Rényi divergence of the distribution of the real transcript from that of the simulated transcript is sufficiently small, say, $(1+r) \in [1, \infty]$. This paper only considers the Rényi divergence of infinity order.
The adaptive reprogramming approach for (s)EUF-CMA1 proof reprograms the random oracle on $O(q_S)$ points with the simulated transcript, where q_S is the number of the signing queries, and the bound with random oracle reprogrammed with real transcripts will be $(1+r)^{O(q_S)} \cdot \epsilon$ due to the property of the Rényi divergence, where ϵ is the bound with the random oracle reprogrammed with simulated transcripts. In contrast, the history-free approach reprograms the random oracle on $O(1)$ points per message with the simulated transcripts, and the multiplicative security loss can be $(1+r)^{O(M)}$, where M is the size of the message space.
In this paper, we show other bound $(1+r)^{O(1) \cdot \ell}(\epsilon + O(q^3)/\ell) + \mathsf{negl}(\kappa)$ for any positive ℓ. This allows us to set $\ell = q^3/\mathsf{negl}(\kappa)$, which can be smaller than M. Moreover, if we $r = 1/\ell$, then the multiplicative security loss can be constant. This result is obtained by extending [BZ13b, Lemma 2.5] for the statistical case into the Rényi divergence case, which may be of independent interest.

First (memory-tight) PO **and** sBU **security proofs for LID-based signatures:** We further extend the security proof into the PO and sBU securities. We give a memory-loose PO security proof for DFS-LID and give a memory-tight PO security proof for DFS$^+$-LID, where DFS and DFS$^+$ denote the Fiat-Shamir transform derandomized by PRF and random oracle, respectively. We also give a memory-tight sBU security proof for DFS-LID. As far as we know, those are the first direct PO and sBU security proofs for the LID-based signature schemes in the QROM.
In order to consider memory tightness, we modify the PO security model with the forgery-checking oracle; the forgery-checking oracle maintains the list $\mathcal{Q}$ of the pairs of messages and valid signatures the adversary submits, and the adversary wins if the size of the list is larger than the number of quantum singing queries q_S, that is, $\#\mathcal{Q} > q_S$. In the PO security proof, we need to replace this winning condition A ($\#\mathcal{Q} > q_S$), which requires memory of $O(q_S)$ size, with the condition B that the adversary's signature (w^*, z^*) on m^* is different from $(\tilde{w}, \tilde{z})$ the signing oracle produced on the m^*. To treat this change, we first consider the difference

Table 1. Summary of security proofs for LID-based signatures in the QROM. IND, CUR, and PRF in the column "Assumptions" denote the key indistinguishability of LID, the computational unique response of LID, and the pseudorandomness of PRF, respectively. The mark ✓ in the column "Adv." and "Time" indicates the multiplicative loss of advantage and time is $O(1)$, respectively. The marks ✓ and x in the column "Mem." indicate the additive loss of memory usage is $O(1)\cdot\mathsf{poly}$ and $O(q)\cdot\mathsf{poly}$, respectively.

Proof	Scheme	Security	Assumptions	Adv.	Time	Mem.
KLS18+DFPS23 [KLS18, DFPS23]	FS-LID	sEUF-CMA1	IND, CUR	✓	✓	x
KLS18+DFPS23 [KLS18, DFPS23]	DFS-LID	sEUF-CMA	IND, CUR, PRF	✓	✓	x
Section 4	FS-LID	msEUF-CMA1	IND, CUR	✓	✓	✓
Section 4	RFS-LID	msEUF-CMA	IND, CUR	✓	✓	✓
Section 4	DFS-LID	msEUF-CMA	IND, CUR, PRF	✓	✓	x
Section 4	DFS$^+$-LID	msEUF-CMA	IND, CUR	✓	✓	✓
Section 6	DFS-LID	PO	IND, CUR, PRF	✓	✓	x
Section 6	DFS$^+$-LID	PO	IND, CUR	✓	✓	✓
Section 5	DFS-LID	sBU	IND, CUR, PRF	✓	✓	✓

Table 2. Summary of security proofs for PSF-based signatures in the QROM. CR, INV, OW, and PRF in the column "Assumptions" denote the collision resistance, non-invertibility, and one-wayness of PSF and the pseudorandomness of PRF, respectively. The marks ✓ and x in the columns "Adv." and "Time" indicate whether the multiplicative loss of advantage and time is $O(1)$ or not. The marks ✓ and x in the column "Mem." indicate whether the additive loss of memory usage is $O(1)\cdot\mathsf{poly}$ and $O(q)\cdot\mathsf{poly}$, respectively. In the signing algorithm of PSF-PFDH*, the randomness for PSF is chosen as 1) choose a random pairwise hash function Q and 2) compute a randomness by $Q(m)$.

Proof	Scheme	Security	Assumptions	Adv.	Time	Mem.
BDFLSZ11 [BDF$^+$11]	PSF-DFDH$^+$	sEUF-CMA	CR	✓	✓	✓
KX24 [KX24]	PSF-PFDH	EUF-CMA	INV	x	✓	x
CCLM22 [CCLM22]	PSF-DFDH	BU	CR, PRF	✓	✓	x
BZ13 [BZ13b]	PSF-PFDH*	PO	OW, CR	x	x	x
BZ13 [BZ13b]	PSF-DFDH	PO	CR, PRF	✓	✓	x
subsection D.3	PSF-FDH	msEUF-CMA1	CR	✓	✓	✓
subsection D.3	PSF-PFDH	msEUF-CMA	CR	✓	✓	✓
subsection D.3	PSF-DFDH	msEUF-CMA	CR, PRF	✓	✓	x
subsection D.3	PSF-DFDH$^+$	msEUF-CMA	CR	✓	✓	✓
subsection D.4	PSF-DFDH	PO	CR, PRF	✓	✓	x
subsection D.4	PSF-DFDH$^+$	PO	CR	✓	✓	✓
subsection D.5	PSF-DFDH	sBU	CR, PRF	✓	✓	✓

between the winning condition A and the winning condition $A \wedge B$. Very roughly speaking, if the adversary makes the difference, it should guess $(q_S + 1)$ signatures on *distinct* messages. The min-entropy of $\tilde{w}$ allows us to upper-bound this guessing probability. After that, we relax the winning condition as B. This modification is the core of our PO security proof. For the details, see Section 6.
The sBU security proof is obtained by following the sEUF-CMA1 security proof, and we omit the details. For the details, see Section 5.

New memory-tight security proofs for PSF-based signatures: We extend the memory-tight sEUF-CMA security proof for PSF-DFDH in the QROM in Boneh et al. [BDF$^+$11] into a memory-tight msEUF-CMA1 security proof for PSF-FDH in the QROM. We then obtain a memory-tight msEUF-CMA security proof for PSF-PFDH in the QROM by using the lemma in [DGJL21] as in the case of RFS-LID. Furthermore, we show a memory-loose PO security proof of PSF-DFDH and a memory-tight PO security proof of PSF-DFDH$^+$ in the QROM, where DFDH$^+$ denotes FDH derandomized by a random function. We also give a memory-tight sBU security proof of PSF-DFDH by modifying a memory-loose BU security proof of PSF-DFDH in the QROM by Chatterjee et al. [CCLM22]. See Section D for the details.
See the summary and comparison in Table 2.

A gap between PO and BU security: As a byproduct, we found that BU security does not imply PO security, which refutes the conjecture that BU security implies PO security of message authentication code (MAC) by Alagic et al. [AMRS20].[7] We exemplify this by constructing a BU-secure but PO-insecure MAC and signature scheme from a BU-secure MAC and signature scheme, respectively. We observe that PO-secure scheme should be sEUF-CMA-secure, but BU-secure scheme can be sEUF-CMA-insecure. Thus, making BU-secure MAC and signature scheme sEUF-CMA-insecure, the new scheme is PO-insecure. We think the conjecture should be that sBU security implies PO security. See Section C for the details.

Open problems: We have managed the imperfect correctness of LID in the memory-tight security proofs of LID-based signature schemes by following the history-free approach [KLS18, DFPS23] instead of the adaptive reprogramming appraoch [GHHM21, DFPS23, BBD$^+$23]. The history-free approach requires LID to have statistical HVZK. While we extend the above approach into the case of *divergence* HVZK [dPRS23, DPS23] for small divergence $(1 + \mathsf{negl}(\kappa))$, we fail to treat large divergence, say, $(1 + 1/\mathsf{poly}(\kappa))$, which will yield shorter signature sizes. We leave it as an open problem to construct memory-tight security proofs treating *large* divergence HVZK and *computational* HVZK, which would require the adaptive reprogramming approach.
We currently can not give the memory-tight security proofs of the PSF-based signature scheme with *imperfectly correct* PSFs. Kosuge and Xagawa [KX24] gave memory-loose security proofs using the adaptive reprogramming technique [GHHM21]. We leave it as an open problem to give memory-tight security proof of signature schemes based on imperfectly-correct PSFs.
Jaeger and Kumar [JK22] gave memory-tight reductions for multi-challenge, multi-user CCA security of PKE/KEM. It is interesting to consider the multi-challenge, multi-user security of signature schemes with memory-tight reductions in the ROM and QROM.

1.2 Organization

Section 2 reviews basic notions and notations, quantum computations, and lossy identification. Section 3 reviews digital signatures, their security notions, and LID-based signature schemes. Section 4, Section 5, and Section 6 show MSEUF-CMA, sBU, and PO security of the LID-based signature schemes. Section A proves the lemma on the Rényi divergence. Section B reviews the instantiations of lossy identifications. Section C discusses the relationship between PO security and BU security of MAC and signatures. Section D includes the review of PSF-based signature schemes and shows their MSEUF-CMA, PO, and sBU securities.

1.3 Version Notes

- 2023-11-09: This is the original version.
- 2023-11-20: We update the references on the reductions related to memory-bounded adversaries.
- 2023-12-13: We correct errors related to the min-entropy of a signature generated by the singing oracle and introduce the special correctness of LID. We also correct errors related to the correctness of the signature and the CUR property of LID.
- 2024-01-09: We modify the proofs to simplify the bounds and remove the restriction on the special correctness of LID. We also add the lemmas related to divergence HVZK.
- 2024-02-28: We move some sections in the appendix to the main body and modify the bounds slightly.
- 2024-06-07: We correct typos.

2 Preliminaries

The security parameter is denoted by $\kappa \in \mathbb{Z}^+$. We use the standard O-notations. For $n \in \mathbb{Z}^+$, we let $[n] := \{1, \dots, n\}$. For a statement P, $[\![P]\!]$ denotes the truth value of P.
Let $\mathcal{X}$ and $\mathcal{Y}$ be two finite sets. $\mathsf{Func}(\mathcal{X}, \mathcal{Y})$ denotes a set of all functions whose domain is $\mathcal{X}$ and codomain is $\mathcal{Y}$. For a set of disributions over $\mathcal{Y}$ indexed by $x \in \mathcal{X}$, $D = \{D_x : x \in \mathcal{X}\}$, we define $\mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D)$ as a distribution of f in $\mathsf{Func}(\mathcal{X}, \mathcal{Y})$ such that, for each $x \in \mathcal{X}$, $f(x)$ is independently drawn from a ditribution D_x. When every D_x is the same as D' on every x, we simply write $\mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D')$.
For a distribution D, we often write "$x \leftarrow D$," which indicates that we take a sample x according to D. For a finite set $\mathcal{S}$, $U(\mathcal{S})$ denotes the uniform distribution over $\mathcal{S}$. We often write "$x \leftarrow \mathcal{S}$" instead of "$x \leftarrow U(\mathcal{S})$." If inp is a string, then "$\mathsf{out} \leftarrow \mathsf{A}^{O}(\mathsf{inp})$" denotes the output of algorithm A running on input inp with an access to a set of oracles O. If A and oracles are deterministic, then out is a fixed value and we write "$\mathsf{out} := \mathsf{A}^{O}(\mathsf{inp})$." We also use the notation "$\mathsf{out} := \mathsf{A}(\mathsf{inp}; r)$" to make the randomness r of A explicit. For a probabilistic algorithm A, $\mathcal{R}_{\mathsf{A}}$ denotes the radnomness space of A.

[7] The conference version [AMRS20] claims that BU-secure MAC is also PO-secure, but, the newest version, '20230420:091107' [AMRS18] reported an error in their proof and removed the claim.

For an algorithm or adversary A, $\mathsf{Time}(\mathsf{A})$ and $\mathsf{Mem}(\mathsf{A})$ denotes the time and memory complexity of the algorithm A, respectively. For a scheme S, $\mathsf{Time}(\mathsf{S})$ and $\mathsf{Mem}(\mathsf{S})$ denotes the maximum time and memory complexity of the algorithms in the scheme S, respectively.
For any function $f : \{0,1\}^n \to \{0,1\}^m$, a *quantum access* to f is modeled as oracle access to unitary $O_f : |x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$. By convention, we will use the notation $\mathsf{A}^{|f\rangle,g}$ to stress A's *quantum* and classical access to f and g.

Distributions: For two distributions D, D' over $\mathcal{Y}$, we say that D is ϵ-close to D' if the distance $|D - D'| := \sum_{y\in\mathcal{Y}}|D(y) - D'(y)|$ is at most ϵ.

Definition 2.1 (Rényi divergence; exponential form). *Let P and Q be two discrete distributions such that* $\mathrm{Supp}(P) \subseteq \mathrm{Supp}(Q)$. *Let $\alpha \in (0,1) \cup (1,\infty)$. The Rényi divergence of order α of P from Q is defined as*

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

For $\alpha = \infty$,

$$R_\infty(P;Q) = \sup_{x\in\mathrm{Supp}(P)} \frac{P(x)}{Q(x)} \in [1, +\infty].$$

Lemma 2.1 ([vEH14]). *Let P and Q be two discrete distributions such that* $\mathrm{Supp}(P) \subseteq \mathrm{Supp}(Q)$. *Let $\alpha \in (0,1) \cup (1,\infty]$.*

- *R1: data processing: For a (randomized) function f,*

$$R_\alpha(f(P); f(Q)) \leq R_\alpha(P;Q).$$

- *R2: probability preservation: For any event $E \subseteq \mathrm{Supp}(Q)$,*

$$P(E) \leq (Q(E) \cdot R_\alpha(P;Q))^{\frac{\alpha-1}{\alpha}} \leq Q(E)^{\frac{\alpha-1}{\alpha}} \cdot R_\alpha(P;Q).$$

  Especially, when $\alpha = \infty$, we have

$$P(E) \leq R_\infty(P;Q) \cdot Q(E).$$

- *R3: Multiplicativity:*

$$R_\alpha\Big(\prod_i P_i; \prod_i Q_i\Big) = \prod_i R_\alpha(P_i;Q_i).$$

We adopt the conventions that $0/0 = 0$ and $x/0 = +\infty$ for $x > 0$.

2.1 Lemmas on Quantum Computations

Lemma 2.2 ([BZ13b, Lemma 2.5, ePrint]). *Let $\mathcal{X}$ and $\mathcal{Y}$ be two finite sets. Let $D = \{D_x\}$ and $D' = \{D'_x\}$ be two sets of efficiently sampleable distributions over $\mathcal{Y}$ indexed by $x \in \mathcal{X}$. let $\mathcal{A}$ be a quantum adversary making q (quantum) queries to an oracle $f : \mathcal{X} \to \mathcal{Y}$. If for each $x \in X$, $|D_x - D'_x| \leq \epsilon$ holds, then* $|\Pr[f \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D) : \mathcal{A}^{|f\rangle} = 1] - \Pr[f \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D') : \mathcal{A}^{|f\rangle} = 1]| \leq \sqrt{(6q)^3\epsilon}$. [8]

Lemma 2.3 ([BZ13b, Lemma 2.6, ePrint]). *Fix two finite sets $\mathcal{X}$ and $\mathcal{Y}$. Fix a set D of distributions D_x over $\mathcal{Y}$ indexed by $x \in \mathcal{X}$. Let α be the minimum over all $x \in \mathcal{X}$ of the min-entropy of the distribution D_x. Now, let $f : \mathcal{X} \to \mathcal{Y}$ be a function chosen according to* $\mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D)$. *Then, any q-query quantum algorithm can only produce $(q+1)$ input/output pairs of f with probability at most $(q+1)/\lfloor 2^\alpha \rfloor$.*

Generic quantum search [Zha12a, HRS16, KLS18]: Let $\mathcal{X}$ be a finite set. The generic search problem (GSP, in short) is finding $x \in \mathcal{X}$ satisfying $g(x) = 1$ given access to an oracle $g : \mathcal{X} \to \{0,1\}$, where for each $x \in \mathcal{X}$, $g(x)$ is drawn independently according to Ber_λ, that is, $g \leftarrow \mathsf{Func}_{\mathcal{X},\{0,1\}}(\mathsf{Ber}_\lambda)$. Kiltz et al. [KLS18] generalized this problem by modifying the global distribution Ber_λ into $\mathsf{Ber}_{\lambda(x)}$ on each x, that is, the case that $g \leftarrow \mathsf{Func}_{\mathcal{X},\{0,1\}}(\{\mathsf{Ber}_{\lambda(x)} : x \in \mathcal{X}\})$.

Lemma 2.4 (Generic search problem with bounded probabilities [KLS18]). *Let $\lambda \in [0,1]$. For any quantum algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ making at most q queries to $|g\rangle$, we have*

$$\Pr[\mathsf{GSPB}_{\lambda,\mathcal{A}} = 1] \leq 8(q+1)^2\lambda,$$

where game $\mathsf{GSPB}_{\lambda,\mathcal{A}}$ is defined in Figure 1.

[8] The value $6^3 = 27 \cdot 8$ is composed from 27 in [Zha12a, Corollary 7.5, ePrint] (denoted by C_0 in [BZ13b]) and 8 in [BZ13b, Lemma 2.5].

```
$\mathsf{GSPB}_{\lambda,\mathcal{A}}$
$\{\lambda(x)\}_{x\in\mathcal{X}} \leftarrow \mathcal{A}_1$
if $\exists x \in \mathcal{X}$ s.t. $\lambda(x) > \lambda$ then return $\perp$
foreach $x \in \mathcal{X}$ : $g(x) \leftarrow \mathsf{Ber}_{\lambda(x)}$
$x \leftarrow \mathcal{A}_2^{|g\rangle}$
return $g(x)$
```

Fig. 1. The generic search game $\mathsf{GSPB}_{\lambda,\mathcal{A}}$.

2.2 Adversaries with Access to Random Functions

We adopt an adversary with access to random functions by following [GGJT22, Section 3]. The reductions in this paper are adversary $\mathcal{A}$ on the left side, consisting of a set of functions $\mathcal{F}$ and algorithm $\mathcal{A}_2$. We call such an adversary *an $\mathcal{F}$-oracle adversary.*

```
Adversary $\mathcal{A}^{O}(\mathsf{in})$
$f \leftarrow \mathcal{F}$
$\mathsf{out} \leftarrow \mathcal{A}_2^{O,|f\rangle}(\mathsf{in})$
return out
```

```
Adversary $\mathcal{A}_F^{O}(\mathsf{in})$
$K \leftarrow \mathcal{K}$
$\mathsf{out} \leftarrow \mathcal{A}_2^{O,|F_K\rangle}(\mathsf{in})$
return out
```

Ghoshal et al. [GGJT22] defined the *reduced complexity* of $\mathcal{A}$ by $\mathsf{Time}^*(\mathcal{A}) := \mathsf{Time}(\mathcal{A}_2)$ and $\mathsf{Mem}^*(\mathcal{A}) := \mathsf{Mem}(\mathcal{A}_2)$. We employ $\mathcal{F}$-oracle adversaries as [GGJT22] for simplicity and clean notation. This approach is justified by pseudorandom adversary $\mathcal{A}_F$ on the right-hand side as long as the game $\mathcal{A}$ plays is efficient.

Lemma 2.5 ([GGJT22, Lemma 2], quantum version). *Let $\mathcal{A}$ be an $\mathcal{F}$-oracle adversary for a game* G*. Then, for any function family* F *with $F = \mathcal{F}$, there exists an adversary $\mathcal{A}_F$ such that*

$$\begin{aligned} |\Pr[\mathsf{G}(\mathcal{A})] - \Pr[\mathsf{G}(\mathcal{A}_F)]| &\le \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{F}}(\mathcal{A}_F), \\ \mathsf{Time}(\mathcal{A}_F) &= \mathsf{Time}^*(\mathcal{A}) + \mathsf{Time}(\mathsf{G}(\mathcal{A})), \\ \mathsf{Mem}(\mathcal{A}_F) &= \mathsf{Mem}^*(\mathcal{A}) + \mathsf{Mem}(\mathsf{G}(\mathcal{A})), \\ \mathsf{Query}(\mathcal{A}_F) &= q, \end{aligned}$$

where q is an upper bound on the number of queries $\mathcal{A}_2$ makes to the oracle $|f\rangle$ or $|F_K\rangle$.

See [ACFK17] and [GGJT22, Lemma 2] for further discussions.

2.3 Pseudorandom Functions

Definition 2.2 (Pseudorandom functions). *Let κ be the security parameter. Let $\delta = \delta(\kappa)$ and $\rho = \rho(\kappa)$ be two polynomial functions. Let* PRF : $\{0,1\}^\kappa \times \{0,1\}^\delta \to \{0,1\}^\rho$ *be a deterministic polynomial-time algorithm. We say that* PRF *is* pseudorandom *if for any QPT adversary $\mathcal{A}$, its advantage*

$$\mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}}(\kappa) := \left| \begin{array}{c} \Pr[K \leftarrow \{0,1\}^\kappa : \mathcal{A}^{|\mathsf{PRF}(K,\cdot)\rangle}(1^\kappa) \to 1] \\ - \Pr[\mathsf{RF} \leftarrow \mathsf{Func}(\{0,1\}^\delta, \{0,1\}^\rho) : \mathcal{A}^{|\mathsf{RF}(\cdot)\rangle}(1^\kappa) \to 1] \end{array} \right|$$

is negligible in κ.

2.4 Lossy Identification

Abdalla et al. [AFLT12, AFLT16] defined lossy identification as a special case of a (cryptographic) identification scheme. A lossy identification scheme involves an additional *lossy* key-generation algorithm. The syntax follows:

Definition 2.3 (Lossy identification). *A lossy identification scheme* LID *consists of the following tuple of PPT algorithms* $(\mathsf{Gen}_{\mathsf{LID}}, \mathsf{LossyGen}_{\mathsf{LID}}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$

- $\mathsf{Gen}_{\mathsf{LID}}(1^\kappa) \to (vk, sk)$*: a normal key-generation algorithm that, on input* 1^κ, *where* κ *is the security parameter, outputs a pair of keys* (vk, sk). vk *and* sk *are public verification and secret keys, respectively.*
- $\mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa) \to vk$*: a lossy key-generation algorithm that on input* 1^κ *outputs a lossy verification key* vk.
- $\mathsf{P}_1(sk) \to (w, s)$*: a first prover algorithm that takes as input signing key* sk *and outputs commitment* w *and state* s.
- $\mathsf{P}_2(sk, w, c, s) \to z$*: a second* deterministic *prover algorithm that takes as input signing key* sk*, commitment* w*, challenge* c*, and state* s*, and outputs response* z.
- $\mathsf{Vrfy}(vk, w, c, z) \to \mathsf{true}/\mathsf{false}$*: a* deterministic *verification algorithm that takes as input verification key* vk*, commitment* w*, challenge* c*, and response* z*, and outputs its decision* true *or* false.

We assume that a verification key vk *defines the challenge space* $\mathcal{C}$ *and the response space* $\mathcal{Z}$.

Definition 2.4 (Completeness). *For non-negligible* $\rho = \rho(\kappa)$*, we call* LID ρ-complete *if*

$$\Pr\left[\begin{matrix}(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa), (w, s) \leftarrow \mathsf{P}_1(sk), \\ c \leftarrow \mathcal{C}, z := \mathsf{P}_2(sk, w, c, s)\end{matrix} : \mathsf{V}(vk, w, c, z) = \mathsf{true}\right] \geq \rho(\kappa).$$

We call LID perfecly complete *if it is* 1*-complete.*

In order to analyze the completeness carefully, Devevey et al. [DFPS23] introduced another definition as follows:

Definition 2.5 (Correctness [DFPS23, Definition 2], adapted). *Let* $\gamma, \beta > 0$*. We call* LID (γ, β)-correct *if for every* (vk, sk) *generated by* $\mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$*, the following holds:*

- *The verifier accepts with probability at least* γ *if the response* z *is not* $\bot$*. That is,*

$$\Pr\left[(w, s) \leftarrow \mathsf{P}_1(sk), c \leftarrow \mathcal{C}, z := \mathsf{P}_2(sk, w, c, s) : \mathsf{V}(vk, w, c, z) = \mathsf{true} \mid z \neq \bot\right] \geq \gamma.$$

- *The prover aborts with probability at most* β*. That is,*

$$\Pr\left[(w, s) \leftarrow \mathsf{P}_1(sk), c \leftarrow \mathcal{C}, z := \mathsf{P}_2(sk, w, c, s) : z = \bot\right] \leq \beta.$$

We note that if LID is $(1, \beta)$-correct, then it is $(1 - \beta)$-complete.
The security properties of a lossy identification scheme are defined as follows:

Definition 2.6 (Key indistinguishability [AFLT16, Definition 16]). *We say that* LID *is* key indistinguishable *if for any QPT adversary* $\mathcal{A}$*, its advantage* $\mathsf{Adv}^{\text{ind-key}}_{\mathsf{LID},\mathcal{A}}(\kappa)$ *is negligible in* κ*, where*

$$\mathsf{Adv}^{\text{ind-key}}_{\mathsf{LID},\mathcal{A}}(\kappa) := \left|\begin{matrix}\Pr[(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa) : \mathcal{A}(vk) = 1] \\ -\Pr[vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa) : \mathcal{A}(vk) = 1]\end{matrix}\right|.$$

Definition 2.7 (Lossiness [AFLT16, Definition 16], adapted). *We say that* LID *is* ϵ_ℓ-lossy *if for any unbounded adversary* $\mathcal{A}$*, its advantage* $\mathsf{Adv}^{\text{imp}}_{\mathsf{LID},\mathcal{A}}(\kappa)$ *is at most* ϵ_ℓ*, where*

$$\mathsf{Adv}^{\text{imp}}_{\mathsf{LID},\mathcal{A}}(\kappa) := \Pr\left[\begin{matrix}vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa), (w, s) \leftarrow \mathcal{A}(vk), \\ c \leftarrow \mathcal{C}, z \leftarrow \mathcal{A}(c, s)\end{matrix} : \mathsf{V}(vk, w, c, z) = \mathsf{true}\right].$$

Remark 2.1 (On Lossiness). In the original definition, adversary $\mathcal{A}$ can access the simulated transcript oracle to produce (w, s). However, since the simulated transcript oracle has no access to sk, we do need to consider this oracle.

Definition 2.8 (Statistical honest-verifier zero knowledge [DFPS23], adapted). *Let* (vk, sk) *be a key pair generated by* $\mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$*. We call* LID ϵ_{zk}-HVZK *if there exists a PPT algorithm* Sim *that takes a public verification key* vk *and* c *as input and outputs* (w, z) *such that the distribution of* (w, c, z) *where* $c \leftarrow \mathcal{C}$ *and* $(w, z) \leftarrow \mathsf{Sim}(vk, c)$ *is* ϵ_{zk}*-close to the distribution of the real transcript between honest prover and verifier.*

Remark 2.2. In [AFLT16, KLS18], "the distribution of the real transcript" is defined as follows: compute (w, c, z) by using the real prover and verifier; if $z = \bot$, then return $(\bot, \bot, \bot)$; otherwise return (w, c, z). Devevey et al. [DFPS23] pointed out that this definition is one of the causes of the error in the simulation. They defined "the distribution of the real transcript" as follows: compute (w, c, z) by using the real prover and verifier and output (w, c, z) *as it is.*

Definition 2.9 (Divergence honest-verifier zero knowledge [DFPS23], simplified). *Let* (vk, sk) *be a key pair generated by* $\mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$*. We call* LID $(1 + \epsilon_{\text{zk}})$-divergence HVZK *if there exists a PPT algorithm* Sim *that takes a public verification key* vk *and* c *as input and outputs* (w, z) *such that* $R_\infty((w, c, z); (\tilde{w}, \tilde{c}, \tilde{z})) \leq 1 + \epsilon_{\text{zk}}$ *holds, where* (w, c, z) *is the real transcript between an honest prover and verifier and* $\tilde{c} \leftarrow \mathcal{C}$ *and* $(\tilde{w}, \tilde{z}) \leftarrow \mathsf{Sim}(vk, \tilde{c})$.

Definition 2.10 (Commitment recoverability [KLS18, Defnition 2.4]). *We say that* LID *is* commitment-recoverable *if for any* (vk, sk) *generated by* $\mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$*,* $c \in \mathcal{C}$*, and* $z \in \mathcal{Z}$*, there exists a* unique w *such that* $\mathsf{V}(vk, w, c, z) = \mathsf{true}$*. In addition, we require that this unique* w *can be computed by a deterministic commitment-recovery algorithm* Rec*, that is,* $w = \mathsf{Rec}(vk, c, z)$*.*

Definition 2.11 (Computational unique response [KLS18, Defnition 2.7]). *We also say that* LID *has the* computational unique response (CUR) *property if for any QPT adversary* $\mathcal{A}$*, its advantage defined below is negligible in* κ*:*

$$\mathsf{Adv}^{\mathrm{cur}}_{\mathsf{LID},\mathcal{A}}(\kappa) := \Pr\left[\begin{array}{c}(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa), (w, c, z, z') \leftarrow \mathcal{A}(vk) : \\ z \neq z' \wedge \mathsf{V}(vk, w, c, z) \wedge \mathsf{V}(vk, w, c, z')\end{array}\right].$$

Definition 2.12 (Min-entropy of commitment [KLS18, Defnition 2.6], modified). *We say that* LID *has* (α, ϵ_m)-min-entropy *if*

$$\Pr[(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa) : H_\infty(w \mid (w, s) \leftarrow \mathsf{P}_1(sk)) \geq \alpha] \geq 1 - \epsilon_m.$$

In the original definition ([KLS18, Defnition 2.6]), ϵ_m is $2^{-\alpha}$. Devevey et al. [DFPS23, Definition 5] defined the min-entropy of commitment as $(\alpha, 0)$-min entropy in the above definition.

3 Digital Signature

The model for digital signature schemes is summarized as follows:

Definition 3.1. *A digital signature scheme* DS *consists of the following triple of PPT algorithms* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$*:*

- $\mathsf{Gen}(1^\kappa) \to (vk, sk)$*: a key-generation algorithm that, on input* 1^κ*, where* κ *is the security parameter, outputs a pair of keys* (vk, sk)*.* vk *and* sk *are called verification and signing keys, respectively.*
- $\mathsf{Sign}(sk, \mu) \to \sigma$*: a signing algorithm that takes as input signing key* sk *and message* $\mu \in \mathcal{M}$ *and outputs signature* $\sigma \in \mathcal{S}$*.*
- $\mathsf{Vrfy}(vk, \mu, \sigma) \to \mathsf{true}/\mathsf{false}$*: a verification algorithm that takes as input verification key* vk*, message* $\mu \in \mathcal{M}$*, and signature* σ *and outputs its decision* true *or* false*.*

Definition 3.2 (Completeness). *We say that* DS *is* ρ-complete *if for any message* $\mu \in \mathcal{M}$*, we have*

$$\Pr[(vk, sk) \leftarrow \mathsf{Gen}(1^\kappa), \sigma \leftarrow \mathsf{Sign}(sk, \mu) : \mathsf{Vrfy}(vk, \mu, \sigma) = \mathsf{true}] \geq \rho.$$

Security notions: We review the standard security notions of digital signature schemes. The standard security notion, existential unforgeability against chosen-message attack (EUF-CMA), is captured by the game $\mathsf{Expt}^{\mathrm{euf\text{-}cma}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$ in Figure 2. The multi-challenge version allows an adversary to call Forge freely [ACFK17]. We also consider a strong version, in which the adversary wins if its forgery (m^*, σ^*) is not equal to the pairs returned by Sign. For signing oracles, we have two variants, one-signature-per-message / many-signature-per-message versions, which are denoted by CMA1 and CMA, respectively. We note that for deterministic signature schemes, CMA1 security implies CMA security. The formal definition follows:

Definition 3.3 (Security notions for digital signature schemes). *Let* $\mathsf{DS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a digital signature scheme. For any* $\mathcal{A}$*,* goal $\in$ {euf, seuf, meuf, mseuf}*, and* atk $\in$ {cma, cma1}*, we define its* goal-atk *advantage against* DS *as follows:*

$$\mathsf{Adv}^{\mathrm{goal\text{-}atk}}_{\mathsf{DS},\mathcal{A}}(\kappa) := \Pr[\mathsf{Expt}^{\mathrm{goal\text{-}atk}}_{\mathsf{DS},\mathcal{A}}(1^\kappa) = 1],$$

where $\mathsf{Expt}^{\mathrm{goal\text{-}atk}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$ *is an experiment described in Figure 2. For* GOAL $\in$ {EUF, sEUF, mEUF, msEUF} *and* ATK $\in$ {CMA, CMA1}*, we say that* DS *is* GOAL-ATK-*secure if* $\mathsf{Adv}^{\mathrm{goal\text{-}atk}}_{\mathsf{DS},\mathcal{A}}(\kappa)$ *is negligible for any QPT adversary* $\mathcal{A}$*.*

Security with respect to quantum signing oracles: Boneh and Zhandry [BZ13b] defined a new security notion of digital signature schemes with respect to a *quantum* signing oracle and dubbed it as EUF-qCMA security. We refer to this security notion as *plus-one security* (PO security) [AMRS20] because an adversary in the security game is asked to output $q + 1$ distinct valid message/signature pairs after making q quantum queries to the signing oracle. They defined it in the same spirit as the *strong* EUF security. In the original definition, the adversary outputs $q + 1$ pairs at once and stops. We introduce the oracle Forge to the security game of the PO security since we want to consider memory tightness. The formal definition follows:

Definition 3.4 (Plus-One Security [BZ13b], adapted). *Let* $\mathsf{DS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a digital signature scheme. For any* $\mathcal{A}$*, we define its* po *advantage against* DS *as follows:*

$$\mathsf{Adv}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(\kappa) := \Pr[\mathsf{Expt}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(1^\kappa) = 1],$$

where $\mathsf{Expt}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$ *is an experiment described in Figure 3. We say that* DS *is* PO-*secure if* $\mathsf{Adv}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(\kappa)$ *is negligible for any QPT adversary* $\mathcal{A}$*.*

```
$\mathsf{Expt}^{\text{goal-atk}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$ for goal $\in \{\text{euf}, \text{seuf}\}$
$(vk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
$\mathcal{Q} := \emptyset$; $\mathsf{win} := \mathsf{false}$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\textsc{Sign}}(vk)$
$\textsc{Forge}(m^*, \sigma^*)$
return $\mathsf{win}$

$\textsc{Sign}(m)$ for atk = cma
$\sigma \leftarrow \mathsf{Sign}(sk, m)$
$\mathcal{Q} := \mathcal{Q} \cup \{(m, \sigma)\}$
return $\sigma$

$\textsc{Forge}(m^*, \sigma^*)$ for goal $\in \{\text{euf}, \text{meuf}\}$
if $\mathsf{Vrfy}(vk, m^*, \sigma^*) = \mathsf{true}$ then
    if $\forall \sigma : (m^*, \sigma) \notin \mathcal{Q}$ then
        $\mathsf{win} := \mathsf{true}$
```

```
$\mathsf{Expt}^{\text{goal-atk}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$ for goal $\in \{\text{meuf}, \text{mseuf}\}$
$(vk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
$\mathcal{Q} := \emptyset$; $\mathsf{win} := \mathsf{false}$
run $\mathcal{A}^{\textsc{Sign},\textsc{Forge}}(vk)$
return $\mathsf{win}$

$\textsc{Sign}(m)$ for atk = cma1
if $\exists (m, \sigma) \in \mathcal{Q}$ then
    return $\sigma$
$\sigma \leftarrow \mathsf{Sign}(sk, m)$
$\mathcal{Q} := \mathcal{Q} \cup \{(m, \sigma)\}$
return $\sigma$

$\textsc{Forge}(m^*, \sigma^*)$ for goal $\in \{\text{seuf}, \text{mseuf}\}$
if $\mathsf{Vrfy}(vk, m^*, \sigma^*) = \mathsf{true}$ then
    if $(m^*, \sigma^*) \notin \mathcal{Q}$ then
        $\mathsf{win} := \mathsf{true}$
```

Fig. 2. $\mathsf{Expt}^{\text{goal-atk}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$ for goal $\in \{\text{euf}, \text{seuf}, \text{meuf}, \text{mseuf}\}$ and atk $\in \{\text{cma}, \text{cma1}\}$.

```
$\mathsf{Expt}^{\text{po}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$
$(vk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
$\mathcal{Q} := \emptyset$
run $\mathcal{A}^{|\textsc{Sign}\rangle,\textsc{Forge}}(vk)$
return $[\![\#\mathcal{Q} > q_S]\!]$

$\textsc{Forge}(m^*, \sigma^*)$
if $\mathsf{Vrfy}(vk, m^*, \sigma^*) = \mathsf{true}$ then
    if $(m^*, \sigma^*) \notin \mathcal{Q}$ then
        $\mathcal{Q} := \mathcal{Q} \cup \{(m^*, \sigma^*)\}$
```

```
$\textsc{Sign} : |m\rangle |y\rangle \mapsto |m\rangle |y \oplus \sigma\rangle$
/generate randomness $r$ on each query
/share $r$ on every message
$\sigma := \mathsf{Sign}(sk, m; r)$
return $\sigma$
```

Fig. 3. $\mathsf{Expt}^{\text{po}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$. q_S denotes the number of the signing queries.

Alagic et al. [AMRS20] introduced another new security notion concerning a quantum signing oracle and called it *blind unforgeability* (BU security). Let $\epsilon \in \{0/2^p, 1/2^p, \dots, (2^p - 1)/2^p\}$ for some $p = p(\kappa)$ be a parameter. Let B_ϵ be a random subset of the message space $\mathcal{M}$ where each $m \in \mathcal{M}$ is independently selected with probability ϵ. Roughly speaking, an adversary is asked to output a valid signature on a message in B_ϵ while it can access a quantum signing oracle that returns a signature on a message *not* in B_ϵ. The strong version is defined by a subset of the product of the message space $\mathcal{M}$ and the signature space $\mathcal{S} \subseteq \{0,1\}^\lambda$ for some $\lambda = \lambda(\kappa)$. For $f : \mathcal{M} \to \mathcal{S}$, $B \subseteq \mathcal{M}$, and $B' \subseteq \mathcal{M} \times \mathcal{S}$, we define

$$Bf(x) := \begin{cases} \bot & x \in B \\ f(x) & \text{otherwise} \end{cases} \text{ and } B'f(x) := \begin{cases} \bot & (x, f(x)) \in B' \\ f(x) & \text{otherwise.} \end{cases}$$

Remark 3.1. We consider the oracle $|Bf\rangle$ as a mapping $|x\rangle |y\rangle \mapsto |x\rangle |y \oplus Bf(x)\rangle$, where $y \in \{0,1\}^{\lambda+1}$, $f(x)$ is considered as $f(x)\|0 \in \{0,1\}^{\lambda+1}$, and $\bot$ is considered as $0^\lambda\|1 \in \{0,1\}^{\lambda+1}$.
In the security proofs, instead of choosing a random subset B_ϵ, we will consider $\mathsf{RF}_B : \mathcal{M} \to \mathcal{P}$, where $\mathcal{P} = \{0, 1, \dots, 2^p - 1\}$, and we will interpret the condition $m \in B_\epsilon$ as $\mathsf{RF}_B(m) < \epsilon 2^p$. The cost of this procedure is denoted by $\mathsf{Time}(B_\epsilon)$ and $\mathsf{Mem}(B_\epsilon)$.

We again introduce the oracle $\textsc{Forge}$ to the security game and consider the multi-challenge situation. The formal definition follows:

Definition 3.5 (Blind Unforgeability [AMRS20], adapted). *Let* $\mathsf{DS} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a digital signature scheme. For any* $\mathcal{A}$*, any efficiently computable function* $\epsilon : \mathbb{Z}^+ \to [0,1)$*, and* sec $\in \{\text{bu}, \text{sbu}\}$*, we define its* goal-atk *advantage against* DS *as follows:*

$$\mathsf{Adv}^{\text{sec}}_{\mathsf{DS},\mathcal{A}}(\kappa) := \Pr[\mathsf{Expt}^{\text{sec}}_{\mathsf{DS},\mathcal{A},\epsilon}(1^\kappa) = 1],$$

where $\mathsf{Expt}^{\text{sec}}_{\mathsf{DS},\mathcal{A}}(1^\kappa)$ *is an experiment described in Figure 4. We say that* DS *is* BU-secure *(*sBU-secure*, resp.) if* $\mathsf{Adv}^{\text{bu}}_{\mathsf{DS},\mathcal{A},\epsilon}(\kappa)$ *(*$\mathsf{Adv}^{\text{sbu}}_{\mathsf{DS},\mathcal{A},\epsilon}(\kappa)$*, resp.) is negligible for any QPT adversary* $\mathcal{A}$ *and any efficiently computable function* $\epsilon : \mathbb{Z}^+ \to [0,1)$.

```
$\underline{\mathsf{Expt}^{\mathrm{sec}}_{\mathsf{DS},\mathcal{A}}(1^\kappa) \text{ for sec} \in \{\mathrm{bu}, \mathrm{sbu}\}}$
$(vk, sk) \leftarrow \mathsf{Gen}(1^\kappa)$
$B_\epsilon \leftarrow \mathsf{Func}_{\mathcal{M},\{0,1\}}(\mathsf{Ber}_\epsilon)$                    /bu
$B_\epsilon \leftarrow \mathsf{Func}_{\mathcal{M}\times\mathcal{S},\{0,1\}}(\mathsf{Ber}_\epsilon)$                   /sbu
$\mathsf{win} := \mathsf{false}$
run $\mathcal{A}^{|B_\epsilon \textsc{Sign}\rangle,\textsc{Forge}}(vk)$
return $\mathsf{win}$
```

```
$\underline{\textsc{Sign} : |m\rangle\,|y\rangle \mapsto |m\rangle\,|y \oplus \sigma\rangle}$
/generate randomness $r$ on each query
/share $r$ on every message
$\sigma := \mathsf{Sign}(sk, m; r)$
return $\sigma$
```

```
$\underline{\textsc{Forge}(m^*, \sigma^*)}$
if $\mathsf{Vrfy}(vk, m^*, \sigma^*) = \mathsf{true}$ then
    if $m^* \in B_\epsilon$ then $\mathsf{win} := \mathsf{true}$                    /bu
    if $(m^*, \sigma^*) \in B_\epsilon$ then $\mathsf{win} := \mathsf{true}$              /sbu
```

Fig. 4. $\mathsf{Expt}^{\mathrm{bu}}_{\mathsf{DS},\mathcal{A},\epsilon}(1^\kappa)$ and $\mathsf{Expt}^{\mathrm{sbu}}_{\mathsf{DS},\mathcal{A},\epsilon}(1^\kappa)$.

```
$\underline{\mathsf{Gen}(1^\kappa)}$
$(vk, sk) \leftarrow \mathsf{Gen}'(1^\kappa)$
return $(vk, sk)$
```

```
$\underline{\mathsf{Sign}(sk, m)}$
$n \leftarrow \{0,1\}^\lambda$
$\sigma' \leftarrow \mathsf{Sign}'(sk, (m, n))$
return $\sigma := (\sigma', n)$
```

```
$\underline{\mathsf{Vrfy}(vk, m, \sigma)}$
parse $\sigma = (\sigma', n)$
return $\mathsf{Vrfy}'(vk, (m, n), \sigma')$
```

Fig. 5. Scheme $\mathsf{DS} := \mathsf{RDS}[\mathsf{DS}', \lambda]$. We require $\mathcal{M}' = \mathcal{M} \times \{0,1\}^\lambda$.

3.1 From CMA1 Security to CMA Security

Diemert et al. [DGJL21] shows the following lemma, which is the multi-challenge version of [BPS16, Theorem 5].

Lemma 3.1 ([DGJL21, Theorem 14]). *Let DS' be a signature scheme whose message space is $\mathcal{M}' = \mathcal{M} \times \{0,1\}^\lambda$ and let DS be $\mathsf{RDS}[\mathsf{DS}', \lambda]$ in Figure 5. Let $\mathcal{A}$ be an adversary against the* MSEUF-CMA *security of DS which queries to* Sign *q_S times. Then, there exists an adversary $\mathcal{B}$ against the* MSEUF-CMA1 *security of DS' such that*

$$\mathsf{Adv}^{\text{mseuf-cma}}_{\mathsf{DS},\mathcal{A}}(\kappa) \leq \mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS}',\mathcal{B}}(\kappa) + q_S^2 \cdot 2^{-\lambda},$$
$$\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}), \text{ and } \mathsf{Mem}(\mathcal{B}) = \mathsf{Mem}(\mathcal{A}).$$

If the signature scheme is *deterministic*, then the CMA1 security implies the CMA security [KLS18]. Thus, if we derandomize a CMA1-secure signature scheme, the obtained scheme archives CMA security. See, e.g., [MNPV99, KM15] for the derandomization by PRF.[9] Unfortunately, the derandomization by PRF is sometimes annoying when we consider memory-tight reductions.

3.2 Signature from Lossy Identification

We review a signature scheme constructed from a lossy identification scheme with abort [AFLT16, KLS18]. Let $\mathsf{LID} = (\mathsf{Gen}_{\mathsf{LID}}, \mathsf{LossyGen}_{\mathsf{LID}}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ be a lossy idenfitication scheme. The signature scheme obtained by applying a variant of the Fiat-Shamir transform $\mathsf{FS}_{B,wz}$ is depicted in Figure 6.
One might think if the underlying LID scheme is ρ-complete, then the obtained scheme is $(1-(1-\rho)^B)$-complete. For the correctness of the general case, Devevey et al. [DFPS23] gave a careful analysis of the correctness of the obtained signature scheme:

Lemma 3.2 ([DFPS23, Theorem 8], adapted). *Let $\gamma > 0$ and $\beta \in (0,1)$. Let $B > 0$. Let $\mathsf{H} : \mathcal{M} \times \mathcal{W} \to \mathcal{C}$ be a hash function modeled as a random oracle. Let LID be a LID scheme that is (γ, β)-correct and has (α, ϵ_m)-commitment min-entropy. Let $\mathsf{DS} = \mathsf{FS}_{B,wz}[\mathsf{LID}, \mathsf{H}]$. Then, for any message $\mu \in \mathcal{M}$, we have*

$$\Pr_{(vk,sk)\leftarrow\mathsf{Gen}(1^\kappa)}\left[\Pr[\mathsf{V}(vk, m, \mathsf{Sign}(sk, m)) = \mathsf{true}] \geq \rho'(\alpha, \beta, \gamma, B)\right] \geq 1 - \epsilon_m,$$

where the inner probability is taken over the choice of H and the coins of Sign and

$$\rho'(\alpha, \beta, \gamma, B) := \gamma \cdot \left(1 - \beta^B - \frac{2^{-\alpha}}{(1-\beta)^3}\right).$$

[9] Let K be a secret key of PRF independent of the signing key sk. Kiltz et al. [KLS18] credited the security proof of a signature scheme derandomized by $\mathsf{PRF}(K, m)$ to Bellare et al. [BPS16]. Unfortunately, the derandomization proposed by Bellare et al. [BPS16] computes randomness as $\mathsf{RF}(sk \| m)$ instead of $\mathsf{PRF}(K, m)$ and their proof does not work for the case of $\mathsf{H}(K, m)$.

```
$\mathsf{Gen}(1^\kappa)$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$
return $(vk, sk)$
```

```
$\mathsf{Sign}_{wz}(sk, m)/\mathsf{Sign}_{cz}(sk, m)$
$k := 1; z := \bot$
while $z = \bot \wedge k \leq B$ do
    $(w, s) \leftarrow \mathsf{P}_1(sk)$
    $c := \mathsf{H}(m, w)$
    $z := \mathsf{P}_2(sk, w, c, s)$
    $k := k + 1$
if $z = \bot$ then return $\bot$
return $\sigma := (w, z)$          $/\mathsf{Sign}_{wz}$
return $\sigma := (c, z)$          $/\mathsf{Sign}_{cz}$
```

```
$\mathsf{Vrfy}_{wz}(vk, m, \sigma)$
parse $\sigma = (w, z)$
$c := \mathsf{H}(m, w)$
return $\mathsf{V}(vk, w, c, z)$

$\mathsf{Vrfy}_{cz}(vk, m, \sigma)$
parse $\sigma = (c, z)$
$w' := \mathsf{Rec}(vk, c, z)$
$c' := \mathsf{H}(m, w')$
return $[\![c = c']\!]$
```

Fig. 6. Scheme $\mathsf{FS}_{B,wz}[\mathsf{LID}, \mathsf{H}] = (\mathsf{Gen}, \mathsf{Sign}_{wz}, \mathsf{Vrfy}_{wz})$ and $\mathsf{FS}_{B,cz}[\mathsf{LID}, \mathsf{H}] = (\mathsf{Gen}, \mathsf{Sign}_{cz}, \mathsf{Vrfy}_{cz})$. $\mathsf{H} \colon \mathcal{M} \times \mathcal{W} \to \mathcal{C}$ is a random oracle.

For simplicity, in what follows, we just say that the signature scheme is $\rho'(\alpha, \beta, \gamma, B)$-correct with probability at least $1 - \epsilon_m$ over the choice of key.

When the underlying LID is commitment-recoverable, we can apply another variant $\mathsf{FS}_{B,cz}$ depicted in Figure 6 whose signature is of the form (c, z), which is often shorter than (w, z). If P_1 is derandomized by PRF, say, $\mathsf{P}_1(sk; \mathsf{PRF}(K, (m, k)))$, then we call this conversion as DFS instead of FS and denote $\mathsf{DFS}[\mathsf{LID}, \mathsf{H}, \mathsf{PRF}]$. If we use RF instead of PRF, then we denote it as $\mathsf{DFS}^+[\mathsf{LID}, \mathsf{H}, \mathsf{RF}]$. If we apply RDS in subsection 3.1 to the obtained scheme, then we call the conversion as RFS and denote $\mathsf{RFS}[\mathsf{LID}, \mathsf{H}, \lambda]$.

In this paper, we employ $\mathsf{FS}_{B,wz}$ to capture generic case, while [DGJL21] only consider $\mathsf{FS}_{B,cz}$. We can show the security of $\mathsf{FS}_{B,cz}$ by modifying our proofs for MSEUF-CMA, sBU, and PO securities.

4 Multi-Challenge Security of Signature from Lossy Identification

Theorem 4.1 **(MSEUF-CMA1 security of $\mathsf{FS}_{B,wz}[\mathsf{LID}, \mathsf{H}]$).** *Let $B \geq 1$. Let $\mathsf{H} \colon \mathcal{M} \times \mathcal{W} \to \mathcal{C}$ be a hash function modeled as a random oracle. Let LID be a lossy identification scheme that is (γ, β)-correct, ϵ_{zk}-HVZK, and ϵ_ℓ-lossy, and has (α, ϵ_m)-commitment min-entropy. Let $\mathsf{DS} := \mathsf{FS}_{B,wz}[\mathsf{LID}, \mathsf{H}]$ and let ρ' be the completeness of DS.*
Then, for a quantum adversary $\mathcal{A}$ breaking the MSEUF-CMA1 security of DS that issues at most q_H quantum queries to the random oracle H, q_S classical queries to the signing oracle, and q_F classical queries to the forgery oracle, there exist quantum $\mathcal{F}$-oracle adversaries $\mathcal{A}_{\mathrm{cur}}$ against computationally unique response of LID and $\mathcal{A}_{\mathrm{ind}}$ against key indistinguishability of LID such that

$$\begin{aligned}
\mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS},\mathcal{A}}(\kappa) &\leq \mathsf{Adv}^{\text{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa) + \mathsf{Adv}^{\text{ind-key}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa) + 8(q+1)^2\epsilon_\ell \\
&\qquad + 8(q+1)^2(1-\rho') + q_F B 2^{-\alpha} + 2qB2^{-\frac{-\alpha-1}{2}} + 3\epsilon_m + \sqrt{(6q)^3 B\epsilon_{\mathrm{zk}}}, \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{cur}}) &= \mathsf{Time}(\mathcal{A}) + q \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cur}}) &= \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})), \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{ind}}) &= \mathsf{Time}(\mathcal{A}) + q \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{ind}}) &= \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})),
\end{aligned}$$

where $q = q_S + q_H + q_F$ and $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R})$.

Applying Lemma 3.1, we obtain the following corollary.

Corollary 4.1 **(MSEUF-CMA security of $\mathsf{RFS}_{B,wz}[\mathsf{LID}, \mathsf{H}, \lambda]$).** *For sufficiently large $\lambda = \omega(\kappa)$, $\mathsf{RFS}_{B,wz}[\mathsf{LID}, \mathsf{H}, \lambda]$ has a memory-tight MSEUF-CMA security proof.*

We also have the following corollary for divergence HZVK.

Corollary 4.2. *Let ℓ be an arbitrary positive integer. If LID is $(1 + \epsilon_{\mathrm{zk}})$-divergence HVZK, then the bound is*

$$\begin{aligned}
\mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS},\mathcal{A}}(\kappa) &\leq (1+\epsilon_{\mathrm{zk}})^{B\ell}\left(\mathsf{Adv}^{\text{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa) + \mathsf{Adv}^{\text{ind-key}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa) + 8(q+1)^2\epsilon_\ell + 27q^3/\ell\right) \\
&\qquad + 27q^3/\ell + 8(q+1)^2(1-\rho') + q_F B2^{-\alpha} + 2qB2^{-\frac{-\alpha-1}{2}} + 3\epsilon_m.
\end{aligned}$$

Especially, if we can set $\ell = q^3/\delta$ for some negligible function δ and $\epsilon_{\mathrm{zk}} = \delta/q^3$, then we have

$$\begin{aligned}
\mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS},\mathcal{A}}(\kappa) &\leq e^B \cdot \left(\mathsf{Adv}^{\text{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa) + \mathsf{Adv}^{\text{ind-key}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa) + 8(q+1)^2\epsilon_\ell + 27\delta\right) \\
&\qquad + 27\delta + 8(q+1)^2(1-\rho') + q_F B2^{-\alpha} + 2qB2^{-\frac{-\alpha-1}{2}} + 3\epsilon_m.
\end{aligned}$$

```
$\underline{\mathsf{G}_0, \dots, \mathsf{G}_{12}}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$                         $/\mathsf{G}_0\text{-}\mathsf{G}_{11}$
$vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)$                          $/\mathsf{G}_{12}$
$\mathsf{RF}_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C})$
$\mathsf{RF}'_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C})$                    $/\mathsf{G}_6\text{-}$
$\mathsf{RF}_\mathsf{P} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{P}_1})$                    $/\mathsf{G}_2\text{-}\mathsf{G}_6$
$\mathsf{RF}_\mathsf{Sim} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_\mathsf{Sim})$                  $/\mathsf{G}_7\text{-}$
$\mathcal{Q} := \emptyset$                                       $/\mathsf{G}_0\text{-}\mathsf{G}_3$
$\mathsf{win} := \mathsf{false}$
$\textbf{run } \mathcal{A}^{\textsc{Sign},\textsc{Forge},|\mathsf{H}\rangle}(vk)$
$\textbf{return } \mathsf{win}$
```

```
$\underline{\textsc{Sign}(m)}$
$\textbf{if } \exists(m, \sigma) \in \mathcal{Q} \textbf{ then return } \sigma$              $/\mathsf{G}_0\text{-}\mathsf{G}_1$
$\textbf{if } \textsc{GetTrans}(m) = \Finv \textbf{ then return } \Finv$              $/\mathsf{G}_6\text{-}$
$\left(w^{(k)}, c^{(k)}, z^{(k)}\right) \leftarrow \textsc{GetTrans}(m)$                 $/\mathsf{G}_0$
$\left\{\left(w^{(i)}, c^{(i)}, z^{(i)}\right)\right\}_{i \in [k]} \leftarrow \textsc{GetTrans}(m)$           $/\mathsf{G}_1\text{-}$
$\textbf{if } z^{(k)} = \perp \textbf{ then}$
    $\sigma := \perp$
$\textbf{else}$
    $\sigma := \left(w^{(k)}, z^{(k)}\right)$
$\mathcal{Q} := \mathcal{Q} \cup \{(m, \sigma)\}$                            $/\mathsf{G}_0\text{-}\mathsf{G}_3$
$\textbf{return } \sigma$
```

```
$\underline{\mathsf{H} :\ |m, w\rangle\,|y\rangle \mapsto |m, w\rangle\,|y \oplus c'\rangle}$
$\textbf{return } c' := \mathsf{RF}_\mathsf{H}(m, w)$                                      $/\mathsf{G}_0\text{-}\mathsf{G}_4$
$\textbf{if } \textsc{GetTrans}(m) = \Finv \textbf{ then return } \Finv$                         $/\mathsf{G}_6\text{-}$
$\left\{\left(w^{(i)}, c^{(i)}, z^{(i)}\right)\right\}_{i \in [k]} \leftarrow \textsc{GetTrans}(m)$                      $/\mathsf{G}_5\text{-}$
$\textbf{if } \exists i : w = w^{(i)} \textbf{ then } c' := c^{(i)} \textbf{ else } c' := \mathsf{RF}_\mathsf{H}(m, w)$     $/\mathsf{G}_5\text{-}$
```

```
$\underline{\textsc{GetTrans}(m)}$
$k := 1; z^{(0)} := \perp$
$\textbf{while } z^{(k-1)} = \perp \wedge k \leq B \textbf{ do}$
    $(w^{(k)}, s) \leftarrow \mathsf{P}_1(sk)$                                    $/\mathsf{G}_0\text{-}\mathsf{G}_1$
    $(w^{(k)}, s) := \mathsf{P}_1(sk; \mathsf{RF}_\mathsf{P}(m, k))$                         $/\mathsf{G}_2\text{-}\mathsf{G}_8$
    $c^{(k)} := \mathsf{RF}_\mathsf{H}(m, w^{(k)})$                                  $/\mathsf{G}_0\text{-}\mathsf{G}_6$
    $c^{(k)} := \mathsf{RF}'_\mathsf{H}(m, k)$                                      $/\mathsf{G}_7\text{-}$
    $z^{(k)} := \mathsf{P}_2(sk, w^{(k)}, c^{(k)}, s)$                              $/\mathsf{G}_0\text{-}\mathsf{G}_8$
    $(w^{(k)}, z^{(k)}) := \mathsf{Sim}(vk, c^{(k)}; \mathsf{RF}_\mathsf{Sim}(m, k))$                  $/\mathsf{G}_9$
    $k := k + 1$
$k := k - 1$                                /cancel the last increment
$\textbf{if } \mathsf{Coll}(\{w^{(i)}\}_{i \in [k]}) = \mathsf{true} \textbf{ then return } \Finv$                 $/\mathsf{G}_6\text{-}$
$\textbf{return } (w^{(k)}, c^{(k)}, z^{(k)})$                                  $/\mathsf{G}_0$
$\textbf{return } \left\{\left(w^{(i)}, c^{(i)}, z^{(i)}\right)\right\}_{i \in [k]}$                        $/\mathsf{G}_1\text{-}$
```

```
$\underline{\textsc{Forge}(m^*, \sigma^*) \text{ where } \sigma^* = (w^*, z^*)}$
$\textbf{if } \textsc{GetTrans}(m) = \Finv \textbf{ then return } \Finv$                                      $/\mathsf{G}_6\text{-}$
$\left\{\left(\tilde{w}^{(i)}, \tilde{c}^{(i)}, \tilde{z}^{(i)}\right)\right\}_{i \in [k]} \leftarrow \textsc{GetTrans}(m^*)$                                   $/\mathsf{G}_3\text{-}$
$\textbf{if } \mathsf{V}(vk, \tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{false} \textbf{ then return } \Finv$                        $/\mathsf{G}_3\text{-}$
$c^* := \mathsf{H}(m^*, w^*)$
$\textbf{if } \mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true} \wedge (m^*, \sigma^*) \notin \mathcal{Q} \textbf{ then}$                         $/\mathsf{G}_0\text{-}\mathsf{G}_3$
    $\mathsf{win} := \mathsf{true}$                                                    $/\mathsf{G}_0\text{-}\mathsf{G}_3$
$\textbf{if } \mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true} \wedge (w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)}) \textbf{ then}$                   $/\mathsf{G}_4\text{-}$
    $\mathsf{win} := \mathsf{true}$                                                    $/\mathsf{G}_4\text{-}\mathsf{G}_7$
    $\mathcal{L}_{m^*} := \{\tilde{w}^{(i)}\}_{i \in [k]}; \mathcal{L}'_{m^*} := \{\tilde{w}^{(i)}\}_{i \in [k-1]}$                        $/\mathsf{G}_8\text{-}\mathsf{G}_{10}$
    $\textbf{if } (w^* \notin \mathcal{L}'_{m^*}) \vee (w^* \in \mathcal{L}'_{m^*} \wedge c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*)) \textbf{ then } \mathsf{win} := \mathsf{true}$   $/\mathsf{G}_8\text{-}\mathsf{G}_9$
    $\textbf{if } (w^* \notin \mathcal{L}_{m^*}) \vee (w^* \in \mathcal{L}'_{m^*} \wedge c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*)) \textbf{ then } \mathsf{win} := \mathsf{true}$    $/\mathsf{G}_{10}$
    $\textbf{if } w^* \neq \tilde{w}^{(k)} \wedge c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*) \textbf{ then } \mathsf{win} := \mathsf{true}$                   $/\mathsf{G}_{11}\text{-}$
```

Fig. 7. G_i for $i \in \{0, 1, \dots, 12\}$ for MSEUF-CMA1 security.

4.1 Proof of Theorem

Roadmap: We define thirteen games G_i for $i \in \{0, 1, \dots, 12\}$ to show our theorem. Let W_i denote the event that G_i outputs true. Before describing games, we briefly give intuitions for games. In what follows, $\textsc{GetTrans}(m)$ denotes the oracle generating at most B transcripts invoked from the signing oracle. Let $(w^{(i)}, c^{(i)}, z^{(i)})$ be the i-th transcript of $\textsc{GetTrans}(m)$.

While we mainly follow the proof by Devevey et al. [DFPS23], the details are different. We consider the original game (G_0), in which the signing oracle queried on m calls $\textsc{GetTrans}(m)$ internally and uses this real transcript as a signature. After derandomizing in G_2, we modify the forgery-checking oracle to output a special symbol if the signature $(w^{(k)}, z^{(k)})$ generated from $\{(w^{(i)}, c^{(i)}, z^{(i)})\}_{i \in [k]}$ output by $\textsc{GetTrans}(m^*)$ yields an *invalid* signature (G_3). We then remove the list $\mathcal{Q}$ and replace the condition $(m^*, (w^*, z^*)) \notin \mathcal{Q}$ with $(w^*, z^*) \neq (w^{(k)}, z^{(k)})$ in G_4 as Diemert et al. [DGJL21]. While the random oracle leaks the information of $w^{(k)}$, we can show that the min-entropy of $w^{(k)}$ is high unless m^* is queried to the signing oracle and the adversary's guessing probability is at most $B2^{-\alpha}$. We then modify the random oracle to patch the hash value on $\mathsf{H}(m, w^{(i)})$ by $c^{(i)}$ instead of $\mathsf{RF}_\mathsf{H}(m, w^{(i)})$, where $(w^{(i)}, c^{(i)}, z^{(i)})$ is the i-th transcript of $\textsc{GetTrans}(m)$ (G_7). We further implement $\textsc{GetTrans}(m)$ by the simulator (G_9), which removes the use of sk in the following games. We then consider the case that $w^* = \tilde{w}^{(k)}$ in G_{10}, which violates the CUR property. After additional small modifications, we arrive at G_{12} in which we replace a normal verification key with a lossy verification key (G_{12}) as in [AFLT12, KLS18], and this replacement is justified key indistinguishability of LID. Finally, in G_{12}, the adversary wins with negligible probability as in Kiltz et al. [KLS18] due to ϵ_ℓ-lossiness.

The formal definitions of games follow.

Game G_0*:* This is the original game. See Figure 7 for a concrete definition of G_0, where we expand the Sign algorithm and H is defined as RF_H. By the definition, we have

$$\Pr[W_0] = \mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS},\mathcal{A}}(\kappa).$$

Game G_1*:* In this game, GetTrans outputs *all* transcripts instead of the last one. The signing oracle also takes the last one as a candidate for a signature. See G_1 in Figure 7 for the detail. Since this is a conceptual change, we have

$$\mathsf{G}_0 = \mathsf{G}_1.$$

Game G_2*:* We next derandomize the prover in GetTrans by RF_P as in Figure 7. Since we consider *one-signature-per-one-message* situation, this derandomization by the random function RF_P changes nothing. We have

$$\mathsf{G}_1 = \mathsf{G}_2.$$

Game G_3*:* We next modify the forgery-checking oracle as follows: On a query (m^*, σ^*), the oracle Forge first comptues the transcripts $\left\{ \left(\tilde{w}^{(i)}, \tilde{c}^{(i)}, \tilde{z}^{(i)}\right) \right\}_{i \in [k]}$. If $(\tilde{w}^{(k)}, \tilde{z}^{(k)})$ is an *invalid* signature, then the oracle returns a special symbol Ⅎ. See G_3 in Figure 7 for the details.
We note that GetTrans's output yields an *invalid* signature with probability at most $1 - \rho'$ as discussed in subsection 3.2. Notice that the adversary can obtain this information via the oracle Sign *and the random oracle* H.

Lemma 4.1. *Let* Bad *be the event that the oracle* Forge *returns the symbol* Ⅎ*. Suppose that* LID *is* (γ, β)*-correct and has* (α, ϵ_m)*-commitment min-entropy. We have*

$$|\Pr[W_2] - \Pr[W_3]| \le \Pr[\mathsf{Bad}] \le 8(q_S + q_F + q_H + 1)^2(1 - \rho') + \epsilon_m.$$

For the proof, see Lemma 5.2 for the sBU security.

Game G_4*:* In this game, we replace the condition $(m^*, \sigma^*) \notin \mathcal{Q}$ with the condition $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$. See G_4 in Figure 7.

Lemma 4.2. *Suppose that that* LID *has* (α, ϵ_m)*-commit min-entropy. Then, we have*

$$|\Pr[W_3] - \Pr[W_4]| \le q_F \cdot B2^{-\alpha} + \epsilon_m.$$

Proof. Suppose that the adversary queries m^* and $\sigma^* = (w^*, z^*)$ to the oracle Forge.
If m^* is queried to the signing oracle before, then there is no difference between the two games: The signing oracle returns $(\tilde{w}^{(k)}, \tilde{z}^{(k)})$ as a signature on m^* and, thus, the condition $(m^*, \sigma^*) \notin \mathcal{Q}$ and the condition $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ are equivalent.
If m^* is not queried to the signing oracle, then the two games might differ if the adversary queries m^* and $(w^*, z^*) = (\tilde{w}^{(k)}, \tilde{z}^{(k)})$, which implies $w^* = \tilde{w}^{(k)}$. This means that the adversary succeeds to guess $\tilde{w}^{(k)}$ on m^* without knowing it. Let Bad_i denote the event that m^* is not queried before but w^* equals to $\tilde{w}^{(k)}$ happens in G_i. We have

$$|\Pr[W_3] - \Pr[W_4]| \le \Pr[\mathsf{Bad}_3].$$

According to Proposition 4.1, even if we know the *whole* table of RF_H, and $\tilde{w}^{(k)}$ has min-entropy $\alpha - \lg(B)$ with probability $1 - \epsilon_m$ over choice of (vk, sk). Therefore, we have $\Pr[\mathsf{Bad}_3] \le q_F \cdot B2^{-\alpha} + \epsilon_m$. □

Proposition 4.1. *Fix* (vk, sk) *and suppose that the min-entropy of* w *is at least* α*. In* G_3*, we have for any* m^**,*

$$\max_{w \in \mathcal{W}} \Pr[\tilde{w}^{(k)} = w \mid \mathsf{H}] \le B \cdot 2^{-\alpha}.$$

Proof. To simplify the argument, let $\mathcal{W} = [W]$. We consider the distribution of the table on $\mathsf{H}(m^*, \cdot)$ and let $C_1, \dots, C_W$ be random variables representing values of $\mathsf{H}(m^*, 1), \dots, \mathsf{H}(m^*, W)$.
Let p_i denote the probability that $i \in [W]$ is chosen by the prover. We have $\sum_{i \in [W]} p_i = 1$. By the definition of the commitment min-entropy, we have $\max_{i \in [W]} p_i \le 2^{-\alpha}$. Let $q_{i,c}$ denote the probability that the prover outputs $z = \bot$ when it chooses i as the commitment and receives c as the challenge.
Let us fix the values of the table $\mathsf{H}(m^*, \cdot)$ as $\boldsymbol{c} = (c_1, \dots, c_W) \in C^W$. This fix allows us to compute the probability that the prover outputs $z = \bot$, which is $\beta_{\boldsymbol{c}} := \sum_{i \in [W]} p_i \cdot q_{i,c_i} \le 1$. On each try except the last one, $\tilde{w}^{(k)} = i$ is chosen with probability $p_i(1 - q_{i,c_i})$. On the last try, $\tilde{w}^{(k)} = i$ is chosen with probability p_i. Thus, for any $B \ge 1$, we have

$$\begin{aligned}
\Pr[\tilde{w}^{(k)} = i \mid C = \boldsymbol{c}] &= p_i(1 - q_{i,c_i}) + \beta_{\boldsymbol{c}} p_i(1 - q_{i,c_i}) + \cdots + \beta_{\boldsymbol{c}}^{B-2} p_i(1 - q_{i,c_i}) + \beta_{\boldsymbol{c}}^{B-1} p_i \\
&= p_i(1 - q_{i,c_i})(1 + \beta_{\boldsymbol{c}} + \cdots + \beta_{\boldsymbol{c}}^{B-2}) + p_i \beta_{\boldsymbol{c}}^{B-1} \\
&\le p_i(1 + \beta_{\boldsymbol{c}} + \cdots + \beta_{\boldsymbol{c}}^{B-1}) \\
&\le B \cdot p_i,
\end{aligned}$$

where we use the inequatlity $(1 + x + x^2 + \cdots + x^{B-1}) \le B$ for $x \in [0, 1]$. This yields that

$$\max_{i \in [W]} \Pr[\tilde{w}^{(k)} = i \mid C = \boldsymbol{c}] \le \max_{i \in [W]} B \cdot p_i \le B \cdot 2^{-\alpha}$$

as we wanted. □

Game G_5*:* We next modify the random oracle as follows: On a query (m, w), the oracle first comptues the transcripts $\left\{ \left(w^{(i)}, c^{(i)}, z^{(i)}\right) \right\}_{i \in [k]}$. If the input w is equivalent to one of $w^{(i)}$, then it returns $c' := c^{(i)}$; otherwise, it returns $c' := \mathsf{RF}_{\mathsf{H}}(m, w)$. See G_5 in Figure 7 for the details.
Since $c^{(i)}$ is defined as $\mathsf{RF}_{\mathsf{H}}(m, w^{(i)})$ in GetTrans, this change nothing and we have

$$\mathsf{G}_4 = \mathsf{G}_5.$$

Game G_6*:* We next introduce a collision check for $w^{(i)}$'s in GetTrans. If GetTrans finds a collision among $w^{(1)}, \dots, w^{(k)}$, it outputs a special synbol $\Finv$. See G_6 in Figure 7.
For each message, the collision probability is at most $B^2 \cdot 2^{-\alpha-1}$ if $H_\infty(w)$ is α [DFPS23, Lemma 11]. Using the one-sided O2H lemma [AHU19], Devevey et al. showed the following lemma, where we additionally introduce ϵ_m.

Lemma 4.3. *Suppose that* LID *has* (α, ϵ_m)*-commitment min-entropy. Then, we have*

$$|\Pr[W_5] - \Pr[W_6]| \leq 2(q_S + q_H + q_F) \cdot B \cdot 2^{\frac{-\alpha-1}{2}} + \epsilon_m.$$

Game G_7*:* We next modify how to compute $c^{(k)}$ in GetTrans, in which it is computed as $\mathsf{RF}'_{\mathsf{H}}(m, k)$ instead of $\mathsf{RF}_{\mathsf{H}}(m, w^{(k)})$. We note that this does not change the adversary's view because $\mathsf{RF}'_{\mathsf{H}}$ is a random function, and if $w = w^{(i)}$ for the query (m, w), then consistent $c' = c^{(i)} = \mathsf{RF}'_{\mathsf{H}}(m, i)$ is output by H since we already exclude the collision. Thus, we have

$$\mathsf{G}_6 = \mathsf{G}_7.$$

Game G_8*:* To ease the notation, let $\mathcal{L}_{m^*} := \{w^{(i)}\}_{i \in [k]}$ which is the w parts of the transcripts generated by GetTrans(m^*). We additionally define $\mathcal{L}'_{m^*} := \{w^{(i)}\}_{i \in [k-1]}$.
In G_8, Forge additionally checks if $w^* \in \mathcal{L}'_{m^*}$ or not; if so, we additionally checks whether $c^* = \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ or not. See G_8 in Figure 7 for the details.
We have the following lemma.

Lemma 4.4. *We have*

$$\Pr[W_7] = \Pr[W_8].$$

Proof. The two games differ if the adversary queries $w^* = w^{(i)}$ for some $i < k$ but $c^* = c^{(i)} \neq \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$. We call this event in G_i as Bad_i. We have

$$|\Pr[W_7] - \Pr[W_8]| \leq \Pr[\mathsf{Bad}_7] \leq |\Pr[\mathsf{Bad}_7] - \Pr[\mathsf{Bad}_6]| + \Pr[\mathsf{Bad}_6].$$

We have $\Pr[\mathsf{Bad}_6] = 0$ because $c^* = \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ always holds in G_6. Since $\mathsf{G}_6 = \mathsf{G}_7$, we have $\Pr[\mathsf{Bad}_7] = \Pr[\mathsf{Bad}_6]$. Hence, we have $\Pr[W_7] = \Pr[W_8]$. □

Game G_9*:* We next modify GetTrans to use the simulation algorithm. On a query m, the oracle comptues $c^{(i)} := \mathsf{RF}'_{\mathsf{H}}(m, i)$ and $(w^{(i)}, z^{(i)}) := \mathsf{Sim}(vk, c^{(i)}; \mathsf{RF}_{\mathsf{Sim}}(m, i))$. See G_9 in Figure 7 for the details. Since the real transcript and the simulated one is ϵ_{zk}-close, each invocation of GetTrans is $B\epsilon_{\mathrm{zk}}$-close. As Devevey et al. [DFPS23], we have the following lemma by using Lemma 2.2.

Lemma 4.5. *Suppose that* LID *is* ϵ_{zk}*-HVZK. Then, we have*

$$|\Pr[W_8] - \Pr[W_9]| \leq \sqrt{(6(q_S + q_H + q_F))^3 B \epsilon_{\mathrm{zk}}}.$$

Lemma 4.6. *Suppose that* LID *is* $(1 + \epsilon_{\mathrm{zk}})$*-divergence HVZK. Then, for any positive integer* ℓ*, we have*

$$\Pr[W_8] \leq (1 + \epsilon_{\mathrm{zk}})^{B\ell}(\Pr[W_9] + 27q^3/\ell) + 27q^3/\ell.$$

We note that the Rényi divergence of the distribution of GetTrans(m) using the real transcripts from that using the simulator is at most $(1 + \epsilon_{\mathrm{zk}})^B$, since each divergence of the real transcript from the simulated one is at most $1 + \epsilon$ and the total number of transcripts is at most B on each invocation of GetTrans. Applying Lemma A.1 instead of Lemma 2.2, we obtain the lemma.

Game G_{10}*:* We next treat the case $w^* = \tilde{w}^{(k)}$ as a special case. To do so, we replace the condition $w^* \notin \mathcal{L}'_{m^*}$ with $w^* \notin \mathcal{L}_{m^*}$. See G_{10} in Figure 7 for the details.
Because of this modification, if the adversary queries $(m^*, (w^*, z^*))$ satisfying $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$, $w^* = \tilde{w}^{(k)}$, and $\mathsf{Vrfy}(vk, w^*, c^*, z^*) = \mathsf{true}$, then two games may differ. Fortunately, this event is easily treated by the CUR property.

Lemma 4.7. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{cur}}$ such that*

$$\begin{aligned}
&|\Pr[W_9] - \Pr[W_{10}]| \leq \mathsf{Adv}^{\mathrm{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa),\\
&\mathsf{Time}^*(\mathcal{A}_{\mathrm{cur}}) = \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2),\\
&\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cur}}) = \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{Sim}})$.

Proof. Suppose that the queried forgery is (m^*, σ^*) with $\sigma^* = (w^*, z^*)$. Consider the computation in Forge and assume that $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ and $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true}$ for $c^* = \mathsf{H}(m^*, w^*)$ and the flag win is set true in G_9.
Notice that there are no collisions among $\tilde{w}^{(1)}, \dots, \tilde{w}^{(k)}$ and, thus, $\mathcal{L}_{m^*} = \mathcal{L}'_{m^*} \cup \{\tilde{w}^{(k)}\}$. We have the following two cases to analyze G_{10}:

- If $w^* \neq \tilde{w}^{(k)}$, then the flag win is set true in G_{10} also.
- If $w^* = \tilde{w}^{(k)}$, then $c^* := \tilde{c}^{(k)}$. Since $w^* = \tilde{w}^{(k)}$, the flag win is unchanged in G_{10}. However, the check $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ forces $z^* \neq \tilde{z}^{(k)}$, and this z^* leads to break the CUR property by outputting $(w^*, c^*, z^*, \tilde{z}^{(k)})$.

Upon the above observation, we can construct a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{cur}}$ straightforwardly. The analysis of advantage, running time, and memory usage in the lemma are straightforwardly obtained. □

Remark 4.1. We note that $\mathsf{V}(\tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{true}$ holds by the check we introduced in G_3 into Forge. This check is fatal for the above proof because, if $\tilde{z}^{(k)} = \perp$ or $\mathsf{V}(\tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{false}$, then the reduction algorithm fails to output the collision $(w^*, c^*, z^*, \tilde{z}^{(k)})$ breaking the CUR property.

Game G_{11}*:* We again modify the conditions in Forge in G_{10}: Forge checks if $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true}$ for $c^* = \mathsf{H}(m^*, w^*)$, $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$, $w^* \neq \tilde{w}^{(k)}$, and $c^* = \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ or not. If so, the flag is set as true. See G_{11} in Figure 7 for the details.

Lemma 4.8. *We have* $\mathsf{G}_{10} = \mathsf{G}_{11}$.

Proof. Let us consider a forgery $(m^*, (w^*, z^*))$ satisfying $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true}$ for $c^* = \mathsf{H}(m^*, w^*)$ and $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$. Let us consider three cases:

- If $w^* \in \mathcal{L}'_{m^*}$, then there is no diference on the condition $c^* = \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ in both games.
- If $w^* = \tilde{w}^{(k)}$, then win is kept in both games.
- If $w^* \notin \mathcal{L}_{m^*}$, then Forge sets win as true immediately in G_{10} but sets win as true if $c^* = \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ in G_{11}. We note that $c^* := \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ in Forge since $w^* \notin \mathcal{L}_{m^*}$. Thus, Forge in G_{11} also sets $\mathsf{win} := \mathsf{true}$ and there are no differences.

Thus, both games are the same. □

Game G_{12}*:* We finally replace a normal verification key with a lossy verification key. See G_{12} in Figure 7 for the details.

Lemma 4.9. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{ind}}$ such that*

$$\begin{aligned}
&|\Pr[W_{11}] - \Pr[W_{12}]| \leq \mathsf{Adv}^{\mathrm{indkey}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa),\\
&\mathsf{Time}^*(\mathcal{A}_{\mathrm{ind}}) = \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2),\\
&\mathsf{Mem}^*(\mathcal{A}_{\mathrm{ind}}) = \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{Sim}})$.

Proof. We construct $\mathcal{A}_{\mathrm{ind}}$ straightforwardly. The analysis of advantage, running time, and memory usage in the lemma are straightforwardly obtained. □

Lemma 4.10. *Suppose that* LID *is* ϵ_ℓ*-lossy. Then, we have* $\Pr[W_{12}] \leq 8(q_H + q_S + q_F + 1)^2 \epsilon_\ell$.

Since the proof is the same as that in Kiltz et al. [KLS18], we omit it. See the proof of the case for sBU security (Lemma 5.11).

5 Strong Blind Unforgeability of Signature from Lossy Identification

The security proof for MsEUF-CMA1 security can be used to show sBU security of $\mathsf{DFS}_{B,\mathrm{wz}}[\mathsf{LID},\mathsf{H},\mathsf{PRF}]$ as follows:

Theorem 5.1 **(sBU security of $\mathsf{DFS}_{B,\mathrm{wz}}[\mathsf{LID},\mathsf{H},\mathsf{PRF}]$).** *Let $B \geq 1$. Let $\mathsf{H} : \mathcal{M} \times \mathcal{W} \to \mathcal{C}$ be a hash function modeled as a random oracle. Let* LID *be a lossy identification scheme that is (γ, β)-correct, ϵ_{zk}-HVZK, and ϵ_ℓ-lossy, and has (α, ϵ_m)-commitment min-entropy. Let $\mathsf{DS} := \mathsf{DFS}_{B,\mathrm{wz}}[\mathsf{LID},\mathsf{H},\mathsf{PRF}]$ and let ρ' be the completeness of* DS.
Then, for a quantum adversary $\mathcal{A}$ breaking the sBU *security of* DS *that issues at most q_H quantum queries to the random oracle* H, *q_S classical queries to the signing oracle, and q_F classical queries to the forgery-checking oracle, there exist a quantum $\mathcal{F}_{\mathrm{prf}}$-oracle adversary $\mathcal{A}_{\mathrm{prf}}$ against pseudorandomness of* PRF *and quantum $\mathcal{F}$-oracle adversaries $\mathcal{A}_{\mathrm{ind}}$ against key indistinguishability of* LID *and $\mathcal{A}_{\mathrm{cur}}$ against computationally unique response of* LID *such that*

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{sbu}}_{\mathsf{DS},\mathcal{A}}(\kappa) \leq{}& \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}_{\mathrm{prf}}}(\kappa) + \mathsf{Adv}^{\mathrm{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa) + \mathsf{Adv}^{\text{ind-key}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa) + 8(q+1)^2\epsilon_\ell \\
& + 8(q+1)^2(1-\rho') + q_F B 2^{-\alpha} + 2qB2^{-\frac{-\alpha-1}{2}} + 3\epsilon_m + \sqrt{(6q)^3 B \epsilon_{\mathrm{zk}}}, \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{prf}}) ={}& \mathsf{Time}(\mathcal{A}) + (q_S + q_F) \cdot O(B\mathsf{Time}(\mathsf{LID}) + \mathsf{Time}(B_\epsilon)) \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{prf}}) ={}& \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{LID})) + O(\mathsf{Mem}(B_\epsilon)) \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{ind}}) ={}& \mathsf{Time}(\mathcal{A}) + q \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2 + \mathsf{Time}(B_\epsilon)) \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{ind}}) ={}& \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})) + O(\mathsf{Mem}(B_\epsilon)) \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{cur}}) ={}& \mathsf{Time}(\mathcal{A}) + q \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2 + \mathsf{Time}(B_\epsilon)) \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cur}}) ={}& \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})) + O(\mathsf{Mem}(B_\epsilon)),
\end{aligned}$$

where $q = q_H + q_S + q_F$, $\mathcal{F}_{\mathrm{prf}} = \mathsf{Func}(\mathcal{M} \times \mathcal{W} \times \mathcal{Z}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C})$, and $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W} \times \mathcal{Z}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{Sim}})$.

5.1 Proof of Theorem

We define thirteen games G_i for $i \in \{0, 1, \ldots, 12\}$ to show our theorem. Let W_i denote the event that the experiment outputs true in G_i.

Roadmap: The proof of sBU security involves *quantum* singing oracle and the filter B_ϵ. Fortunately, we can take the same approach as the proof of MsEUF-CMA1 security (Theorem 4.1).
The original security game is denoted by G_0, in which the prover in GetTrans is derandomized by PRF. Hence, we replace this PRF with RF in G_1. We modify the games as in the previous proof. In G_4, we modify the winning condition as whether $\mathsf{V}(vk, w^*, c^*, z^*)$, where $c^* = \mathsf{H}(m^*, w^*)$, and $(m^*, (w^*, z^*)) \in B_\epsilon$, *and* $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ or not. We can argue that this modification introduces only a negligible change, as in the previous proof. After that, we continue to modify the games as in the proof of the MsEUF-CMA1 security.

Game G_0*:* This is the original game. See Figure 8 for a concrete definition of G_0, where we expand the Sign algorithm and H is implemented by a random function RF_H. We have

$$\Pr[W_0] = \mathsf{Adv}^{\mathrm{sbu}}_{\mathsf{DS},\mathcal{A}}(\kappa).$$

Game G_1*:* We replace PRF with RF_P in the prover in GetTrans.
By straightforward argument, we have the following lemma:

Lemma 5.1. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{prf}}$ such that*

$$\begin{aligned}
|\Pr[W_0] - \Pr[W_1]| &\leq \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}_{\mathrm{prf}}}(\kappa), \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{cur}}) &= \mathsf{Time}(\mathcal{A}) + (q_S + q_F) \cdot O(B\mathsf{Time}(\mathsf{LID}) + \mathsf{Time}(B_\epsilon)) \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cur}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{LID})) + O(\mathsf{Mem}(B_\epsilon)),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{C} \times \mathcal{Z}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C})$.

Game G_2*:* We next let GetTrans output *all* transcripts instead of the last one. The signing oracle also takes the last one as a candidate for a signature. We have

$$\mathsf{G}_1 = \mathsf{G}_2.$$

```
$\underline{\mathsf{G}_0, \dots, \mathsf{G}_{12}}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$                         $/\mathsf{G}_0\text{-}\mathsf{G}_{11}$
$vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)$                           $/\mathsf{G}_{12}$
$K \leftarrow \{0,1\}^\kappa$                                         $/\mathsf{G}_0$
$\mathsf{RF}_B \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{W} \times \mathcal{Z}, \mathcal{P})$
$\mathsf{RF}_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C})$
$\mathsf{RF}'_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C})$                       $/\mathsf{G}_6\text{-}$
$\mathsf{RF}_\mathsf{P} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{P}_1})$                      $/\mathsf{G}_2\text{-}\mathsf{G}_6$
$\mathsf{RF}_\mathsf{Sim} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_\mathsf{Sim})$                   $/\mathsf{G}_7\text{-}$
$\mathsf{win} := \mathsf{false}$
**run** $\mathcal{A}^{|B_\epsilon\textsc{Sign}\rangle, \textsc{Forge}, |\mathsf{H}\rangle}(vk)$
**return** $\mathsf{win}$
```

```
$\underline{B_\epsilon\textsc{Sign}:\ |m\rangle|y\rangle \mapsto |m\rangle|y \oplus \sigma\rangle}$
**if** $\textsc{GetTrans}(m) = \Finv$ **then return** $\Finv$                      $/\mathsf{G}_6\text{-}$
$(w^{(k)}, c^{(k)}, z^{(k)}) \leftarrow \textsc{GetTrans}(m)$                      $/\mathsf{G}_0\text{-}\mathsf{G}_1$
$\{(w^{(i)}, c^{(i)}, z^{(i)})\}_{i\in[k]} \leftarrow \textsc{GetTrans}(m)$               $/\mathsf{G}_2\text{-}$
**if** $z^{(k)} = \bot \vee (m, (w^{(k)}, z^{(k)})) \in B_\epsilon$ **then**
    $\sigma := \bot$
**else**
    $\sigma := (w^{(k)}, z^{(k)})$
**return** $\sigma$
```

```
$\underline{\mathsf{H}:\ |m,w\rangle|y\rangle \mapsto |m,w\rangle|y \oplus c'\rangle}$
**return** $c' := \mathsf{RF}_\mathsf{H}(m, w)$                                              $/\mathsf{G}_0\text{-}\mathsf{G}_4$
**if** $\textsc{GetTrans}(m) = \Finv$ **then return** $\Finv$                               $/\mathsf{G}_6\text{-}$
$\{(w^{(i)}, c^{(i)}, z^{(i)})\}_{i\in[k]} \leftarrow \textsc{GetTrans}(m)$                        $/\mathsf{G}_5\text{-}$
**if** $\exists i : w = w^{(i)}$ **then** $c' := c^{(i)}$ **else** $c' := \mathsf{RF}_\mathsf{H}(m, w)$      $/\mathsf{G}_5\text{-}$
```

```
$\underline{\textsc{GetTrans}(m)}$
$k := 1;\ z^{(0)} := \bot$
**while** $z^{(k-1)} = \bot \wedge k \leq B$ **do**
    $(w^{(k)}, s) \leftarrow \mathsf{P}_1(sk; \mathsf{PRF}(K, (m, k)))$                          $/\mathsf{G}_0$
    $(w^{(k)}, s) := \mathsf{P}_1(sk; \mathsf{RF}_\mathsf{P}(m, k))$                            $/\mathsf{G}_1\text{-}\mathsf{G}_8$
    $c^{(k)} := \mathsf{RF}_\mathsf{H}(m, w^{(k)})$                                     $/\mathsf{G}_0\text{-}\mathsf{G}_6$
    $c^{(k)} := \mathsf{RF}'_\mathsf{H}(m, k)$                                        $/\mathsf{G}_7\text{-}$
    $z^{(k)} := \mathsf{P}_2(sk, w^{(k)}, c^{(k)}, s)$                                $/\mathsf{G}_0\text{-}\mathsf{G}_8$
    $(w^{(k)}, z^{(k)}) := \mathsf{Sim}(vk, c^{(k)}; \mathsf{RF}_\mathsf{Sim}(m, k))$                  $/\mathsf{G}_9$
    $k := k + 1$
$k := k - 1$                                    /cancel the last increment
**if** $\mathsf{Coll}(\{w^{(i)}\}_{i\in[k]}) = \mathsf{true}$ **then return** $\Finv$                    $/\mathsf{G}_6\text{-}$
**return** $(w^{(k)}, c^{(k)}, z^{(k)})$                                       $/\mathsf{G}_0\text{-}\mathsf{G}_1$
**return** $\{(w^{(i)}, c^{(i)}, z^{(i)})\}_{i\in[k]}$                                 $/\mathsf{G}_2\text{-}$
```

```
$\underline{\textsc{Forge}(m^*, \sigma^*) \text{ where } \sigma^* = (w^*, z^*)}$
**if** $\textsc{GetTrans}(m) = \Finv$ **then return** $\Finv$                                      $/\mathsf{G}_6\text{-}$
$\{(\tilde{w}^{(i)}, \tilde{c}^{(i)}, \tilde{z}^{(i)})\}_{i\in[k]} \leftarrow \textsc{GetTrans}(m^*)$                              $/\mathsf{G}_3\text{-}$
**if** $\mathsf{V}(vk, \tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{false}$ **then return** $\Finv$                         $/\mathsf{G}_3\text{-}$
$c^* := \mathsf{H}(m^*, w^*)$
**if** $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true} \wedge (m^*, (w^*, z^*)) \in B_\epsilon$ **then**
    $\mathsf{win} := \mathsf{true}$                                                       $/\mathsf{G}_0\text{-}\mathsf{G}_3$
    **if** $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ **then**                                      $/\mathsf{G}_4\text{-}$
        $\mathsf{win} := \mathsf{true}$                                                   $/\mathsf{G}_4\text{-}\mathsf{G}_7$
        $\mathcal{L}_{m^*} := \{\tilde{w}^{(i)}\}_{i\in[k]};\ \mathcal{L}'_{m^*} := \{\tilde{w}^{(i)}\}_{i\in[k-1]}$                             $/\mathsf{G}_8\text{-}\mathsf{G}_{10}$
        **if** $(w^* \notin \mathcal{L}'_{m^*}) \vee (w^* \in \mathcal{L}'_{m^*} \wedge c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*))$ **then** $\mathsf{win} := \mathsf{true}$   $/\mathsf{G}_8\text{-}\mathsf{G}_9$
        **if** $(w^* \notin \mathcal{L}_{m^*}) \vee (w^* \in \mathcal{L}'_{m^*} \wedge c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*))$ **then** $\mathsf{win} := \mathsf{true}$    $/\mathsf{G}_{10}$
        **if** $w^* \neq \tilde{w}^{(k)} \wedge c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*)$ **then** $\mathsf{win} := \mathsf{true}$                   $/\mathsf{G}_{11}\text{-}$
```

Fig. 8. G_i for $i \in \{0, 1, \dots, 12\}$ for sBU security.

```
$\mathcal{A}_1$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$
compute $\mathcal{B}_{sk,B} \subseteq C_{sk,B} \subseteq (\mathcal{R}_{\mathsf{P}_1} \times C)^B$
$\forall m \in \mathcal{M}, \lambda_{sk}(m) := \lambda_{sk} = \#\mathcal{B}_{sk,B}/\#C_{sk,B}$
return $\{\lambda_{sk}(m)\}_{m\in\mathcal{M}}, vk, sk$
```

```
$\mathsf{Samp}: \ |m\rangle|y\rangle \mapsto |m\rangle|y \oplus \gamma\rangle$
if $g(m) = 1$ then
    $((r_1, c_1), \dots, (r_B, c_B)) := \mathsf{U}(\mathcal{B}_{sk,B}; \mathsf{RF}_{\mathsf{U}}(m))$
else
    $((r_1, c_1), \dots, (r_B, c_B)) := \mathsf{U}(C_{sk,B} \setminus \mathcal{B}_{sk,B}; \mathsf{RF}_{\mathsf{U}}(m))$
return $\gamma := ((r_1, c_1), \dots, (r_B, c_B))$
```

```
$\mathsf{RF}_{\mathsf{P}}: \ |m, k\rangle|y\rangle \mapsto |m, k\rangle|y \oplus r\rangle$
$((r_1, c_1), \dots, (r_B, c_B)) := \mathsf{Samp}(m)$
return $r_k$
```

```
$\mathsf{RF}_{\mathsf{H}}: \ |m, w\rangle|y\rangle \mapsto |m, w\rangle|y \oplus c\rangle$
$((r_1, c_1), \dots, (r_B, c_B)) := \mathsf{Samp}(m)$
comptue $w_i := \mathsf{P}_1(sk, r_i)$ for $i = 1, \dots, B$
if $\exists i : w_i = w$ then
    return $c_i$
else
    return $c := \mathsf{RF}''_{\mathsf{H}}(m, w)$
```

```
$\mathsf{H}: \ |m, w\rangle|y\rangle \mapsto |m, w\rangle|y \oplus c\rangle$
return $c := \mathsf{RF}_{\mathsf{H}}(m, w)$
```

```
$B_\epsilon\textsc{Sign}: \ |m\rangle|y\rangle \mapsto |m\rangle|m \oplus \sigma\rangle$
$\left\{ \left(w^{(i)}, c^{(i)}, z^{(i)}\right) \right\}_{i\in[k]} \leftarrow \textsc{GetTrans}(m)$
if $z^{(k)} = \bot \lor (m, (w^{(k)}, z^{(k)})) \in B_\epsilon$ then
    return $\sigma := \bot$
else
    return $\sigma := (w^{(k)}, z^{(k)})$
```

```
$\mathcal{A}_2^{|g\rangle}(vk, sk)$
$\mathsf{RF}_B \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{W} \times \mathcal{Z}, \mathcal{P})$
$\mathsf{RF}''_{\mathsf{H}} \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{W}, C)$
$\mathsf{RF}_{\mathsf{U}} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{U}})$
$\mathsf{win} := \mathsf{false}; \hat{m} := \bot$
simulate $B_\epsilon\textsc{Sign}$, $\textsc{Forge}$, and $\mathsf{H}$
run $\mathcal{A}^{|B_\epsilon\textsc{Sign}\rangle, \textsc{Forge}, |\mathsf{H}\rangle}(vk)$
return $\hat{m}$
```

```
$\textsc{GetTrans}(m)$
$k := 1; z^{(0)} := \bot$
while $z^{(k-1)} = \bot \land k \leq B$ do
    $(w^{(k)}, s) := \mathsf{P}_1(sk; \mathsf{RF}_{\mathsf{P}}(m, k))$
    $c^{(k)} := \mathsf{RF}_{\mathsf{H}}(m, w^{(k)})$
    $z^{(k)} := \mathsf{P}_2(sk, w^{(k)}, c^{(k)}, s)$
    $k := k + 1$
$k := k - 1$                /cancel the last increment
return $\left\{ \left(w^{(i)}, c^{(i)}, z^{(i)}\right) \right\}_{i\in[k]}$
```

```
$\textsc{Forge}(m^*, \sigma^*)$ where $\sigma^* = (w^*, z^*)$
$\left\{ \left(\tilde{w}^{(i)}, \tilde{c}^{(i)}, \tilde{z}^{(i)}\right) \right\}_{i\in[k]} \leftarrow \textsc{GetTrans}(m^*)$
if $\mathsf{V}(vk, \tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{false}$ then
    $\hat{m} := m^*$                /detect Bad
    return $\Finv$
$c^* := \mathsf{H}(m^*, w^*)$
if $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true}$ then
    if $(m^*, (w^*, z^*)) \in B_\epsilon$ then
        $\mathsf{win} := \mathsf{true}$
```

Fig. 9. Adversary $\mathcal{A}_{\text{gspb}} = (\mathcal{A}_1, \mathcal{A}_2)$ against GSPB for Lemma 5.2. The set of consistent sequences $C_{sk,B}$, the set of bad sequences $\mathcal{B}_{sk,B}$, and an algorithm U are defined in the proof text.

Game G_3*:* We next modify the forgery-checking oracle as follows: Before checking the validity of submitted query (m^*, σ^*), it generates its own signature $(\tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)})$ by using $\textsc{GetTrans}(m^*)$. If $\textsc{GetTrans}(m^*)$ fails to output a valid signature, then the forgery-checking oracle returns the special symbol $\Finv$.

Lemma 5.2. *Let* Bad *be the event that the oracle* $\textsc{Forge}$ *returns the symbol* $\Finv$*. Suppose that* LID *is* (γ, β)*-correct and has* (α, ϵ_m)*-commitment min-entropy. We have*

$$|\Pr[W_2] - \Pr[W_3]| \leq \Pr[\mathsf{Bad}] \leq 8(q_S + q_F + q_H + 1)^2(1 - \rho') + \epsilon_m.$$

We give the concrete proof since we omitted the corresponding proof in msEUF-CMA1 security (Lemma 4.1). We note that the above lemma is for general LID and we do not need special correctness.

Proof. To make a proof simple, we first eliminate the bad event that the key generation algorithm outputs a bad key pair (vk, sk) making the min-entropy of commitment less than α. This elimination introduces an additional difference ϵ_m.

We define some terminology. In order to simulate RF_{P} and RF_{H}, we consider an algorithm Samp that takes B samples of a pair of randomness of P_1 and challenge in $(\mathcal{R}_{\mathsf{P}_1} \times C)^B$. For a signing key sk, we say that a sequence of B-samples $((r_1, c_1), \dots, (r_B, c_B)) \in (\mathcal{R}_{\mathsf{P}_1} \times C)^B$ is *consistent* if

$$\forall i, j \in [B] : w_i = w_j \Longrightarrow c_i = c_j, \text{ where } (w_i, s_i) := \mathsf{P}_1(sk, r_i).$$

Let $C_{sk,B}$ be the set of all consistent sequences. We say that a consistent sequence is *bad* if 1) the signing algorithm using it fails to generate a signature with $z \neq \bot$ or 2) the signing algorithm using it succeeds to output a signature but the signature is invalid. Let $\mathcal{B}_{sk,B}$ be the set of all bad sequences. Formally, it is defined as

$$\mathcal{B}_{sk,B} := \left\{ \begin{array}{c} ((r_1, c_1), \dots, (r_B, c_B)) \in C_{sk,B} \mid \\ (w_i, s_i) := \mathsf{P}_1(sk, r_i), z_i := \mathsf{P}_2(sk, w_i, c_i, s_i) : \\ (\forall i \in [B] : z_i = \bot) \lor (\exists i \in [B] : z_i \neq \bot \land \mathsf{V}(vk, w_i, c_i, z_i) = \mathsf{false}) \end{array} \right\}.$$

By the definition of (γ, β)-correctness of LID and the discussion in subsection 3.2, we have

$$\operatorname*{Exp}_{(vk,sk)}\left[\#\mathcal{B}_{sk,B}/\#C_{sk,B}\right] \leq 1-\rho'.$$

For a finite set $\mathcal{S}$, U is a probabilistic sampling algorithm that returns $s \leftarrow \mathcal{S}$. For convenience, we define the output of $\mathsf{U}(\emptyset)$ as $\perp$.[10]

Let us construct an unbounded adversary $\mathcal{A}_{\text{gspb}} = (\mathcal{A}_1, \mathcal{A}_2)$ against GSPB defined in Figure 9. The first adversary $\mathcal{A}_1$ outputs a set of bounds $\{\lambda_{sk}(m)\}$, vk, sk, where $\lambda_{sk}(m) = \lambda_{sk} = \#\mathcal{B}_{sk,B}/\#C_{sk,B}$. The value of function g on m is selected according to $\mathsf{Ber}_{\lambda_{sk}}$. The second adversary $\mathcal{A}_2$ tries to output m^* on which GetTrans fails to output a valid signature.

We first consider the success probability of $\mathcal{A}_{\text{gspb}}$ by *fixing* (vk, sk) *and* H. Let us verify the distributions of $r_1, \dots, r_B$ in $\mathsf{RF_P}$ and $c_1, \dots, c_B \in \mathsf{RF_H}$. In Samp, if we took a random sample of a sequence from the set $C_{sk,B}$, then the distributions were perfectly simulated. Instead of this, we check the value of $g(m)$, and if it is 1, then we take a bad sequence uniformly at random; otherwise, we take a good sequence uniformly at random. Since the probablity that $g(m)$ takes 1 with probability $\lambda_{sk} = \#\mathcal{B}_{sk,B}/\#C_{sk,B}$, the distribution of Samp is perfect and the distribution of $\mathsf{RF_P}$ and $\mathsf{RF_H}$ are the same as those in G_2 and G_3.

We then check $\mathcal{A}$'s forgery. If $\hat{m}$ is set as m^*, then the adversary submits m^* such that m^* induces a bad sequence. Hence, $g(m^*) = 1$ and $\mathcal{A}_{\text{gspb}}$ wins the game. Thus, we have

$$\Pr[\mathsf{Bad} \mid vk, sk] = \Pr[\mathsf{GSPB}_{\lambda_{sk}, \mathcal{A}_{\text{gspb}}} = 1 \mid vk, sk] \leq 8(q+1)^2 \lambda_{sk},$$

where q is the number of queries to g of $\mathcal{A}_2$, which is $q \leq (q_S + q_F) \cdot (\#$ of queries by GetTrans$) + q_H \leq (q_S + q_F) \cdot 2B + q_H$. We note that we can reduce the number of queries to g by preparing $((r_1, c_1), \dots, (r_B, c_B)) := \mathsf{Samp}(m)$ at the first steps of GetTrans and Forge and the best bound is $q \leq q_S + q_F + q_H$.

Averaging this inequality over keys, we obtain

$$\Pr[\mathsf{Bad}] \leq 8(q+1)^2 \cdot \operatorname*{Exp}_{(vk,sk)}[\lambda_{sk}] \leq 8(q+1)^2 \cdot (1-\rho')$$

Since $|\Pr[W_2] - \Pr[W_3]| \leq \Pr[\mathsf{Bad}]$, we obtain the bound in the lemma as we wanted. □

Game G_4*:* We next modify the winning condition in Forge: After checking $\mathsf{V}(vk, w^*, c^*, z^*)$, where $c^* = \mathsf{H}(m^*, w^*)$, it sets flag win as true if $(m^*, (w^*, z^*)) \notin B_\epsilon$ *and* $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$. See G_4 in Figure 8.

We note that this modified condition equals that introduced in G_4 for the msEUF-CMA1 security (Theorem 4.1). Thus, similarly, we obtain the following lemma.

Lemma 5.3. *Suppose that* LID *has* (α, ϵ_m)*-commitment min-entropy. Then, we have*

$$|\Pr[W_3] - \Pr[W_4]| \leq q_F \cdot B2^{-\alpha} + \epsilon_m.$$

Proof. The difference occurs if the adversary queries a valid pair of message and signature $(m^*, (w^*, z^*)) \in B_\epsilon$ such that $(w^*, z^*) = (\tilde{w}^{(k)}, \tilde{z}^{(k)})$. If $(m^*, (\tilde{w}^{(k)}, \tilde{z}^{(k)}))$ is not in B_ϵ, then this contradicts with the requirement $(m^*, (w^*, z^*)) \in B_\epsilon$. Thus, $(m^*, (\tilde{w}^{(k)}, \tilde{z}^{(k)}))$ should be blinded by B_ϵ. This means that the adversary cannot obtain the signature $(\tilde{w}^{(k)}, \tilde{z}^{(k)})$ on m^* from the blinded signing oracle B_ϵSign. Thus, the adversary succeeds to guess $w^* = \tilde{w}^{(k)}$ *without knowing* $\tilde{w}^{(k)}$.

Let Bad_i be the event that in G_i the adversary submit $(m^*, (w^*, z^*))$ such that $\mathsf{V}(vk, w^*c,^*, z^*) = \mathsf{true}$, $(m^*, (w^*, z^*)) \in B_\epsilon$, and $(w^*, z^*) = (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ which implies $w^* = \tilde{w}^{(k)}$. As in the proof of Lemma 4.4, we have

$$|\Pr[W_3] - \Pr[W_4]| \leq \Pr[\mathsf{Bad}_3].$$

As Lemma 4.2, we have $\Pr[\mathsf{Bad}_3] = q_F \cdot B2^{-\alpha} + \epsilon_m$ because the min-entropy of $\tilde{w}^{(k)}$ is at least $\alpha - \lg(B)$ with probability at least $1 - \epsilon_m$ over the choice of keys. □

Game G_5*:* We next modify the random oracle as follows: On a query (m, w), the oracle first computes the transcripts. If the input w is equivalent to one of $w^{(i)}$, then it returns $c' := c^{(i)}$; otherwise, it returns $c' := \mathsf{RF_H}(m, w)$. See G_4 in Figure 8 for the details. Since $c^{(i)} = \mathsf{RF_H}(m, w^{(i)})$ in GetTrans, this modification changes nothing and we have

$$\mathsf{G}_4 = \mathsf{G}_5.$$

Game G_6*:* The next game introduces a collision check for $w^{(i)}$'s in GetTrans. Since the min-entropy of $w^{(i)}$ is α-bit with probability $1 - \epsilon_m$, on each invocation of GetTrans, the collision occurs with probability at most $B^2 \cdot 2^{-\alpha-1}$. As Lemma 4.3, we have the following lemma using the one-sided O2H lemma.

Lemma 5.4. *Suppose that* LID *has* (α, ϵ_m)*-commitment min-entropy. Then, we have that*

$$|\Pr[W_5] - \Pr[W_6]| \leq 2(q_S + q_H + q_F) \cdot B \cdot 2^{\frac{-\alpha-1}{2}} + \epsilon_m.$$

[10] But, this never occurs.

Game G_7*:* We next modify how to compute $c^{(k)}$ in GetTrans, in which it is computed as $\mathsf{RF}'_{\mathsf{H}}(m,k)$ instead of $\mathsf{RF}_{\mathsf{H}}(w^{(k)},m)$. We note that this does not change the adversary's view because $\mathsf{RF}'_{\mathsf{H}}$ is a random function, and if $w=w^{(i)}$ for the query (m,w), then consistent $c'=c^{(i)}=\mathsf{RF}'_{\mathsf{H}}(m,i)$ is output by H. (Note that excluding the collision is crucial [DFPS23].) We have

$$\mathsf{G}_6 = \mathsf{G}_7.$$

Game G_8*:* To ease the notation, let $\mathcal{L}_{m^*} := \{w^{(i)}\}_{i\in[k]}$ which are the w parts of the transcripts generated by GetTrans(m^*). We additionally define $\mathcal{L}'_{m^*} := \{w^{(i)}\}_{i\in[k-1]}$. We again modify the game as follows: Let $(m^*,(w^*,z^*))$ be a submitted query to Forge. The oracle additionally checks if $\mathcal{L}'_{m^*}$; if so, it requires $c^* = \mathsf{RF}_{\mathsf{H}}(m^*,w^*)$ as defined in ??. As Lemma 4.4, we have the following lemma:

Lemma 5.5. *We have that*

$$\Pr[W_7] = \Pr[W_8].$$

Proof. The two games may differ if the adversary queries $w^* = w^{(i)}$ for $i<k$ but $c^* \neq \mathsf{RF}_{\mathsf{H}}(m^*,w^*)$. We call this event Bad_i in G_i. As in the proof of Lemma 4.4, we have

$$|\Pr[W_7]-\Pr[W_8]| \leq \Pr[\mathsf{Bad}_7] \leq |\Pr[\mathsf{Bad}_7]-\Pr[\mathsf{Bad}_6]| + \Pr[\mathsf{Bad}_6] \leq \Pr[\mathsf{Bad}_6].$$

Notice that, in G_6, $c^* = \mathsf{RF}_{\mathsf{H}}(m^*,w^*)$ always holds and Bad_6 never occurs. Thus, we have $\Pr[W_7]=\Pr[W_8]$ as we wanted. □

Game G_9*:* We next modify GetTrans to use the simulation algorithm. See G_9 in Figure 7 for the details. As Lemma 4.5 and Lemma 4.6, we have the following lemmas:

Lemma 5.6. *Suppose that* LID *is* ϵ_{zk}*-HVZK. Then, we have*

$$|\Pr[W_8]-\Pr[W_9]| \leq \sqrt{(6(q_S+q_H+q_F))^3 B\epsilon_{\mathrm{zk}}}.$$

Lemma 5.7. *Suppose that* LID *is* $(1+\epsilon_{\mathrm{zk}})$*-divergence HVZK. Then, for any positive integer* ℓ*, we have*

$$\Pr[W_8] \leq (1+\epsilon_{\mathrm{zk}})^{B\ell}(\Pr[W_9]+27q^3/\ell)+27q^3/\ell.$$

Game G_{10}*:* We then treat the case $w^* = \tilde{w}^{(k)}$ as a special case to exclude CUR. To do so, we replace the condition $w^* \notin \mathcal{L}'_{m^*}$ with $w^* \notin \mathcal{L}_{m^*}$. See G_{10} in Figure 8 for the details.
Because of this modification, if the adversary queries $(m^*,(w^*,z^*))$ satisfying $(w^*,z^*) \neq (\tilde{w}^{(k)},\tilde{z}^{(k)})$, $w^* = \tilde{w}^{(k)}$, then two games differ. This is easily treated by the CUR property.

Lemma 5.8. *There exists a quantum* $\mathcal{F}$*-oracle adversary* $\mathcal{A}_{\mathrm{cur}}$ *such that*

$$\begin{aligned}
&|\Pr[W_9]-\Pr[W_{10}]| \leq \mathsf{Adv}^{\mathrm{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa),\\
&\mathsf{Time}^*(\mathcal{A}_{\mathrm{cur}}) = \mathsf{Time}(\mathcal{A}) + (q_H+q_S+q_F)\cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2 + \mathsf{Time}(B_\epsilon)),\\
&\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cur}}) = \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})) + O(\mathsf{Mem}(B_\epsilon)),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M}\times\mathcal{C}\times\mathcal{Z},\mathcal{P})\times\mathsf{Func}(\mathcal{M}\times\mathcal{W},\mathcal{C})\times\mathsf{Func}(\mathcal{M}\times[B],\mathcal{C})\times\mathsf{Func}(\mathcal{M}\times[B],\mathcal{R}_{\mathsf{Sim}})$.

Since the proof is straightforwardly obtained, we omit it.

Game G_{11}*:* We again modify the conditions in Forge in G_{10}: Forge checks if $(m^*,(w^*,z^*)) \in B_\epsilon$, $(w^*,z^*) \neq (\tilde{w}^{(k)},\tilde{z}^{(k)})$, $w^* \neq \tilde{w}^{(k)}$, and $c^* = \mathsf{RF}_{\mathsf{H}}(m^*,w^*)$ or not. If so, the flag is set as true. See G_{11} in Figure 8 for the details.

Lemma 5.9. *We have* $\mathsf{G}_{10} = \mathsf{G}_{11}$.

Proof. Let us consider a valid forgery $(m^*,(w^*,z^*)) \in B_\epsilon$ satisfying $(w^*,z^*) \neq (\tilde{w}^{(k)},\tilde{z}^{(k)})$.
If $w^* \in \mathcal{L}'_{m^*}$, then there is no diference on the condition $c^* = \mathsf{RF}_{\mathsf{H}}(m^*,w^*)$ in both games. If $w^* = \tilde{w}^{(k)}$, then win is kept the same in both games. If $w^* \notin \mathcal{L}_{m^*}$, then we have $c^* = \mathsf{RF}_{\mathsf{H}}(m^*,w^*)$; both flags in G_{10} and G_{11} are set true because $c^* = \mathsf{RF}_{\mathsf{H}}(m^*,w^*)$. Summarizing those three cases, both games are the same. □

```
$\underline{\mathcal{A}_1}$
$vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)$
foreach $w \in \mathcal{W}$ do
    compute $\mathcal{G}_{vk}(w) \subseteq \mathcal{C}$
    $\lambda'_{vk}(w) := \#\mathcal{G}_{vk}(w)/\#\mathcal{C}$
    foreach $m \in \mathcal{M}$ do $\lambda_{vk}(m, w) := \lambda'_{vk}(w)$
return $\{\lambda_{vk}(m, w)\}_{m\in\mathcal{M},w\in\mathcal{W}}, vk$

$\underline{\mathsf{RF}_\mathsf{H} : |m, w\rangle |y\rangle \mapsto |m, w\rangle |y \oplus c\rangle}$
if $g(m, w) = 1$ then
    return $c := \mathsf{U}(\mathcal{G}_{vk}(w); \mathsf{RF}_\mathsf{U}(m, w))$
else
    return $c := \mathsf{U}(\mathcal{C} \setminus \mathcal{G}_{vk}(w); \mathsf{RF}_\mathsf{U}(m, w))$

$\underline{\mathsf{H} : |m, w\rangle |y\rangle \mapsto |m, w\rangle |y \oplus c'\rangle}$
if $\textsc{GetTrans}(m) = \Finv$ then return $\Finv$
$\{ (w^{(i)}, c^{(i)}, z^{(i)}) \}_{i\in[k]} \leftarrow \textsc{GetTrans}(m)$
if $\exists i : w = w^{(i)}$ then $c' := c^{(i)}$ else $c' := \mathsf{RF}_\mathsf{H}(m, w)$
return $c'$

$\underline{B_\epsilon\textsc{Sign} : |m\rangle |y\rangle \mapsto |m\rangle |m \oplus \sigma\rangle}$
if $\textsc{GetTrans}(m) = \Finv$ then return $\Finv$
$\{ (w^{(i)}, c^{(i)}, z^{(i)}) \}_{i\in[k]} \leftarrow \textsc{GetTrans}(m)$
if $z^{(k)} = \bot \vee (m, (w^{(k)}, z^{(k)})) \in B_\epsilon$ then
    return $\sigma := \bot$
else
    return $\sigma := (w^{(k)}, z^{(k)})$

$\underline{\textsc{Forge}(m^*, \sigma^*) \text{ where } \sigma^* = (w^*, z^*)}$
if $\textsc{GetTrans}(m) = \Finv$ then return $\Finv$
$\{ (\tilde{w}^{(i)}, \tilde{c}^{(i)}, \tilde{z}^{(i)}) \}_{i\in[k]} \leftarrow \textsc{GetTrans}(m^*)$
if $\mathsf{V}(vk, \tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{false}$ then return $\Finv$
if $\exists i : w^* = \tilde{w}^{(i)}$ then $c^* := \tilde{c}^{(i)}$ else $c^* := \mathsf{RF}_\mathsf{H}(m^*, w^*)$
if $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true} \wedge (m^*, (c^*, z^*)) \in B_\epsilon \wedge (w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ then
    if $w^* \neq \tilde{w}^{(k)} \wedge c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*)$ then                /detect Bad
        $\hat{m} := m^*; \hat{w} := w^*$ $\mathsf{win} := \mathsf{true}$                    /detect Bad
```

```
$\underline{\mathcal{A}_2^{|g\rangle}(vk)}$
$\mathsf{RF}_B \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{W} \times \mathcal{Z}, \mathcal{P})$
$\mathsf{RF}'_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C})$
$\mathsf{RF}_\mathsf{Sim} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_\mathsf{Sim})$
$\mathsf{RF}_\mathsf{U} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_\mathsf{U})$
$\mathsf{win} := \mathsf{false}; \hat{m} := \bot; \hat{w} := \bot$
simulate $B_\epsilon\textsc{Sign}$, $\textsc{Forge}$, and $\mathsf{H}$
run $\mathcal{A}^{|B_\epsilon\textsc{Sign}\rangle,\textsc{Forge},|\mathsf{H}\rangle}(vk)$
if $\mathsf{win} = \mathsf{true}$ then
    return $(\hat{m}, \hat{w})$
else
    return $\bot$

$\underline{\textsc{GetTrans}(m)}$
$k := 1; z^{(0)} := \bot$
while $z^{(k-1)} = \bot \wedge k \leq B$ do
    $c^{(k)} := \mathsf{RF}'_\mathsf{H}(m, k)$
    $(w^{(k)}, z^{(k)}) := \mathsf{Sim}(vk, c^{(k)}; \mathsf{RF}_\mathsf{Sim}(m, k))$
    $k := k + 1$
$k := k - 1$
$\mathcal{L}_m := \{w^{(i)}\}_{i\in[k]}$
if $\mathsf{Coll}(\mathcal{L}_m)$ then return $\Finv$
return $\{ (w^{(i)}, c^{(i)}, z^{(i)}) \}_{i\in[k]}$
```

Fig. 10. Adversary $\mathcal{A}_{\mathrm{gspb}} = (\mathcal{A}_1, \mathcal{A}_2)$ against GSPB for Lemma 5.11. The set of good challenges $\mathcal{G}_{vk}(w)$ and an algorithm U are defined in the proof text.

Game G_{12}*:* We finally replace a normal verification key with a lossy verification key. See G_{12} in Figure 8 for the details.

Lemma 5.10. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{ind}}$ such that*

$$|\Pr[W_{11}] - \Pr[W_{12}]| \leq \mathsf{Adv}^{\mathrm{indkey}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa),$$
$$\mathsf{Time}^*(\mathcal{A}_{\mathrm{ind}}) = \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2 + \mathsf{Time}(B_\epsilon)),$$
$$\mathsf{Mem}^*(\mathcal{A}_{\mathrm{ind}}) = \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})) + O(\mathsf{Mem}(B_\epsilon)),$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{C} \times \mathcal{Z}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_\mathsf{Sim})$.

Since the proof is obtained by a straightforward reduction, we omit it.

Lemma 5.11. *Suppose that* LID *is ϵ_ℓ-lossy. Then, we have*

$$\Pr[W_{12}] \leq 8(q_S + q_H + q_F + 1)^2 \epsilon_\ell.$$

While we omit the proofs for the cases of MsEUF-CMA1 and PO securities because the proofs are the same as in [KLS18], we here include the proof for sBU security for completeness.
Before giving the proof, we review some terminology.
For a verification key vk and commitment $w \in \mathcal{W}$, we define the set of good challenges as

$$\mathcal{G}_{vk}(w) := \{c \in \mathcal{C} \mid \exists z \in \mathcal{Z} : \mathsf{V}(vk, w, c, z) = \mathsf{true}\}. \quad (1)$$

In [KLS18, Section 2.3], Kiltz et al. discussed that

$$\mathsf{Adv}^{\mathrm{imp}}_{\mathsf{LID},\mathcal{A}}(\kappa) \leq \underset{vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)}{\mathrm{Exp}} \left[\max_{w\in\mathcal{W}} \left(\Pr_{c\leftarrow\mathcal{C}}[\exists z \in \mathcal{Z} : \mathsf{V}(vk, w, c, z) = \mathsf{true}] \right) \right]$$
$$= \underset{vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)}{\mathrm{Exp}} \left[\max_{w\in\mathcal{W}} (\#\mathcal{G}_{vk}(w)/\#\mathcal{C}) \right]$$

and the equality holds when the adversary is optimal by choosing the best $w \in \mathcal{W}$. Thus, if LID is ϵ_ℓ-losy, we have

$$\mathop{\mathrm{Exp}}_{vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)} \left[\max_{w \in \mathcal{W}} (\#\mathcal{G}_{vk}(w)/\#C) \right] \leq \epsilon_\ell. \tag{2}$$

Proof. We follow the proof by Kiltz et al. [KLS18, Theorem 3.4]. For a finite set $\mathcal{S}$, U is a probabilistic sampling algorithm that returns $s \leftarrow \mathcal{S}$. For convenience, we define the output of $\mathsf{U}(\emptyset)$ as $\perp$.[11]

Let us construct an unbounded adversary $\mathcal{A}_{\mathrm{gspb}} = (\mathcal{A}_1, \mathcal{A}_2)$ against GSPB. The first adversary $\mathcal{A}_1$ outputs a set of bounds $\{\lambda_{vk}(m, w)\}$ and vk. The value of function g on (m, w) is selected according to $\mathsf{Ber}_{\lambda_{vk}(m,w)}$. The second adversary $\mathcal{A}_2$ tries to output (m^*, w^*) as in Figure 10. We first consider the success probability of $\mathcal{A}_{\mathrm{gspb}}$ by fixing vk. Let us verify the distribution of c in $\mathsf{RF_H}$. We note that $g(m, w) = 1$ with probability $\lambda_{vk}(m, w) = \#\mathcal{G}_{vk}(w)/\#C$. We have

$$\Pr[c = \tilde{c}] = \begin{cases} \lambda_{vk}(m, w) \cdot \frac{1}{\#\mathcal{G}_{vk}(w)} & (c \in \#\mathcal{G}_{vk}(w)) \\ (1 - \lambda_{vk}(m, w)) \cdot \frac{1}{\#C - \#\mathcal{G}_{vk}(w)} & \text{o.w.}, \end{cases}$$

which is $1/\#C$ in both cases. Hence, the distribution of c in $\mathsf{RF_H}$ is uniform over C (as in G_{12}). We then check $\mathcal{A}$'s forgery. Since $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true}$, where $c^* = \mathsf{RF_H}(m^*, w^*)$, c^* should be a good challenge in $\mathcal{G}_{vk}(w^*)$. This means that $g(m^*, w^*) = 1$ and $\mathcal{A}_{\mathrm{gspb}}$ wins the game. Thus, we have

$$\Pr[W_{12} \mid vk] = \Pr[\mathsf{GSPB}_{\lambda_{vk}, \mathcal{A}_{\mathrm{gspb}}} = 1 \mid vk] \leq 8(q+1)^2 \lambda_{vk},$$

where $\lambda_{vk} := \max_{(m,w) \in \mathcal{M} \times \mathcal{W}} \lambda_{vk}(m, w)$ and q is the number of queries to g. We note that g is queried by H and Forge. Thus, we have $q \leq q_H + q_F \leq q_S + q_H + q_F$.

Averaging this inequality over vk generated by $\mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)$, we obtain

$$\Pr[W_{12}] \leq 8(q+1)^2 \cdot \mathop{\mathrm{Exp}}_{vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)} [\lambda_{vk}] \leq 8(q+1)^2 \cdot \epsilon_\ell$$

as we wanted, where we used $\lambda_{vk} = \max_{(m,w)} \lambda_{vk}(m, w) = \max_w (\#\mathcal{G}_{vk}(w)/\#C)$ and Equation 2. □

6 Plus-One Unforgeability of Signature from Lossy Identification

Theorem 6.1 (**PO security of** $\mathsf{DFS}_{B,\mathrm{wz}}[\mathsf{LID}, \mathsf{H}, \mathsf{PRF}]$). *Let $B \geq 1$. Let $\mathsf{H} \colon \mathcal{M} \times \mathcal{W} \to C$ be a hash function modeled as a random oracle. Let LID be a lossy identification scheme that is (γ, β)-correct, ϵ_{zk}-HVZK, and ϵ_ℓ-lossy, and has (α, ϵ_m)-commitment min-entropy. Let $\mathsf{DS} := \mathsf{DFS}_{B,\mathrm{wz}}[\mathsf{LID}, \mathsf{H}, \mathsf{PRF}]$ and let ρ' be the completeness of DS. Then, for a quantum adversary $\mathcal{A}$ breaking the PO security of DS that issues at most q_H quantum queries to the random oracle H, q_S classical queries to the signing oracle, and q_F classical queries to the forgery-checking oracle, there exists a quantum $\mathcal{F}_{\mathrm{prf}}$-oracle adversary $\mathcal{A}_{\mathrm{prf}}$ against pseudorandomness of PRF and quantum $\mathcal{F}$-oracle adversaries $\mathcal{A}_{\mathrm{ind}}$ against key indistinguishability of LID and $\mathcal{A}_{\mathrm{cur}}$ against computationally unique response of LID such that*

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(\kappa) \leq{} & \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}_{\mathrm{prf}}}(\kappa) + \mathsf{Adv}^{\mathrm{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa) + \mathsf{Adv}^{\text{ind-key}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa) + 8(q+1)^2\epsilon_\ell \\
& + 8(q+1)^2(1-\rho') + \frac{(q_S+1)}{\lfloor 2^\alpha/B \rfloor} + 2qB2^{-\frac{-\alpha-1}{2}} + 3\epsilon_m + \sqrt{(6q)^3 B\epsilon_{\mathrm{zk}}}, \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{prf}}) ={} & \mathsf{Time}(\mathcal{A}) + q_S \cdot O(B\mathsf{Time}(\mathsf{LID})) + q_F \cdot O(\mathsf{Time}(\mathsf{LID})), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{prf}}) ={} & \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})) + q_F \cdot O(\mathsf{Mem}(\mathsf{LID})) + O(\lg(q_S)), \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{ind}}) ={} & \mathsf{Time}(\mathcal{A}) + q \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{ind}}) ={} & \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})), \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{cur}}) ={} & \mathsf{Time}(\mathcal{A}) + q \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cur}}) ={} & \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})),
\end{aligned}$$

where $q = q_H + q_S + q_F$, $\mathcal{F}_{\mathrm{prf}} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, C)$, and $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, C) \times \mathsf{Func}(\mathcal{M} \times [B], C) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{Sim}})$.

As a corollary, when we employ a random function $\mathsf{RF_P}$ directly, the above proof can be modified into a memory-tight one.

Corollary 6.1 (**PO security of** $\mathsf{DFS^+}_{B,\mathrm{wz}}[\mathsf{LID}, \mathsf{H}, \mathsf{RF_P}]$). $\mathsf{DFS^+}_{B,\mathrm{wz}}[\mathsf{LID}, \mathsf{H}, \mathsf{RF_P}]$ *has a memory-tight proof for the* PO *security.*

[11] But, this never occurs.

```
$\underline{\mathsf{G}_0, \dots, \mathsf{G}_{12}}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$                                   $/\mathsf{G}_0\text{-}\mathsf{G}_{11}$
$vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)$                                    $/\mathsf{G}_{12}$
$K \leftarrow \{0,1\}^\kappa$                                                  $/\mathsf{G}_0$
$\mathsf{RF}_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C})$
$\mathsf{RF}'_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C})$                                $/\mathsf{G}_6\text{-}$
$\mathsf{RF}_\mathsf{P} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{P}_1})$                               $/\mathsf{G}_2\text{-}\mathsf{G}_6$
$\mathsf{RF}_{\mathsf{Sim}} \leftarrow \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{Sim}})$                             $/\mathsf{G}_7\text{-}$
$\mathcal{Q} := \emptyset$                                                      $/\mathsf{G}_0\text{-}\mathsf{G}_{4.0}$
$\mathsf{win} := \mathsf{false}$                                                 $/\mathsf{G}_{4.0}\text{-}$
run $\mathcal{A}^{|\textsc{Sign}\rangle, \textsc{Forge}, |\mathsf{H}\rangle}(vk)$
return $[\![\#\mathcal{Q} > q_S]\!]$                                         $/\mathsf{G}_0\text{-}\mathsf{G}_3$
return $[\![\#\mathcal{Q} > q_S]\!] \land \mathsf{win}$                                   $/\mathsf{G}_{4.0}$
return $\mathsf{win}$                                                   $/\mathsf{G}_{4.1}-$

$\underline{\textsc{Sign} : |m\rangle|y\rangle \mapsto |m\rangle|y \oplus \sigma\rangle}$
if $\textsc{GetTrans}(m) = Ⅎ$ then return $Ⅎ$                                 $/\mathsf{G}_6\text{-}$
$\left(w^{(k)}, c^{(k)}, z^{(k)}\right) \leftarrow \textsc{GetTrans}(m)$                          $/\mathsf{G}_0\text{-}\mathsf{G}_1$
$\left\{\left(w^{(i)}, c^{(i)}, z^{(i)}\right)\right\}_{i \in [k]} \leftarrow \textsc{GetTrans}(m)$                     $/\mathsf{G}_2\text{-}$
if $z^{(k)} = \bot$ then
    $\sigma := \bot$
else
    $\sigma := \left(w^{(k)}, z^{(k)}\right)$
$\mathcal{Q} := \mathcal{Q} \cup \{(m, \sigma)\}$                                        $/\mathsf{G}_0\text{-}\mathsf{G}_{4.0}$
return $\sigma$
```

```
$\underline{\mathsf{H} : |m, w\rangle|y\rangle \mapsto |m, w\rangle|y \oplus c'\rangle}$
return $c' := \mathsf{RF}_\mathsf{H}(m, w)$                                                    $/\mathsf{G}_0\text{-}\mathsf{G}_{4.1}$
if $\textsc{GetTrans}(m) = Ⅎ$ then return $Ⅎ$                                               $/\mathsf{G}_6\text{-}$
$\left\{\left(w^{(i)}, c^{(i)}, z^{(i)}\right)\right\}_{i \in [k]} \leftarrow \textsc{GetTrans}(m)$                                   $/\mathsf{G}_5\text{-}$
if $\exists i : w = w^{(i)}$ then $c' := c^{(i)}$ else $c' := \mathsf{RF}_\mathsf{H}(m, w)$                          $/\mathsf{G}_5\text{-}$

$\underline{\textsc{GetTrans}(m)}$
$k := 1; z^{(0)} := \bot$
while $z^{(k-1)} = \bot \land k \leq B$ do
    $(w^{(k)}, s) \leftarrow \mathsf{P}_1(sk; \mathsf{PRF}(K, (m, k)))$                                      $/\mathsf{G}_0$
    $(w^{(k)}, s) := \mathsf{P}_1(sk; \mathsf{RF}_\mathsf{P}(m, k))$                                        $/\mathsf{G}_1\text{-}\mathsf{G}_8$
    $c^{(k)} := \mathsf{RF}_\mathsf{H}(m, w^{(k)})$                                               $/\mathsf{G}_0\text{-}\mathsf{G}_6$
    $c^{(k)} := \mathsf{RF}'_\mathsf{H}(m, k)$                                                  $/\mathsf{G}_7\text{-}$
    $z^{(k)} := \mathsf{P}_2(sk, w^{(k)}, c^{(k)}, s)$                                           $/\mathsf{G}_0\text{-}\mathsf{G}_8$
    $(w^{(k)}, z^{(k)}) := \mathsf{Sim}(vk, c^{(k)}; \mathsf{RF}_{\mathsf{Sim}}(m, k))$                               $/\mathsf{G}_9$
    $k := k + 1$
$k := k - 1$                                       /cancel the last increment
if $\mathsf{Coll}(\{w^{(i)}\}_{i \in [k]}) = \mathsf{true}$ then return $Ⅎ$                                   $/\mathsf{G}_6\text{-}$
return $(w^{(k)}, c^{(k)}, z^{(k)})$                                                    $/\mathsf{G}_0\text{-}\mathsf{G}_1$
return $\left\{\left(w^{(i)}, c^{(i)}, z^{(i)}\right)\right\}_{i \in [k]}$                                           $/\mathsf{G}_2\text{-}$
```

```
$\underline{\textsc{Forge}(m^*, \sigma^*) \text{ where } \sigma^* = (w^*, z^*)}$
if $\textsc{GetTrans}(m) = Ⅎ$ then return $Ⅎ$                                                            $/\mathsf{G}_6\text{-}$
$\left\{\left(\tilde{w}^{(i)}, \tilde{c}^{(i)}, \tilde{z}^{(i)}\right)\right\}_{i \in [k]} \leftarrow \textsc{GetTrans}(m^*)$                                                $/\mathsf{G}_3\text{-}$
if $\mathsf{V}(vk, \tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{false}$ then return $Ⅎ$                                            $/\mathsf{G}_3\text{-}$
$c^* := \mathsf{H}(m^*, w^*)$
if $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true} \land (m^*, \sigma^*) \notin \mathcal{Q}$ then                                     $/\mathsf{G}_0\text{-}\mathsf{G}_{4.0}$
    $\mathcal{Q} := \mathcal{Q} \cup \{(m^*, \sigma^*)\}$                                                     $/\mathsf{G}_0\text{-}\mathsf{G}_{4.0}$
if $\mathsf{V}(vk, w^*, c^*, z^*) = \mathsf{true} \land (w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ then                                $/\mathsf{G}_{4.0}\text{-}$
    $\mathsf{win} := \mathsf{true}$                                                               $/\mathsf{G}_{4.0}\text{-}\mathsf{G}_7$
    $\mathcal{L}_{m^*} := \{\tilde{w}^{(i)}\}_{i \in [k]}; \mathcal{L}'_{m^*} := \{\tilde{w}^{(i)}\}_{i \in [k-1]}$                                          $/\mathsf{G}_8\text{-}\mathsf{G}_{10}$
    if $(w^* \notin \mathcal{L}'_{m^*}) \lor (w^* \in \mathcal{L}'_{m^*} \land c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*))$ then $\mathsf{win} := \mathsf{true}$          $/\mathsf{G}_8\text{-}\mathsf{G}_9$
    if $(w^* \notin \mathcal{L}_{m^*}) \lor (w^* \in \mathcal{L}'_{m^*} \land c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*))$ then $\mathsf{win} := \mathsf{true}$            $/\mathsf{G}_{10}$
    if $w^* \neq \tilde{w}^{(k)} \land c^* = \mathsf{RF}_\mathsf{H}(m^*, w^*)$ then $\mathsf{win} := \mathsf{true}$                               $/\mathsf{G}_{11}\text{-}$
```

Fig. 11. G_i for $i \in \{0, 1, 2, 3, 4.0, 4.1, 5, 6, \dots, 12\}$ for PO security.

6.1 Proof of Theorem

We define fourteen games G_i for $i \in \{0, 1, 2, 3, 4.0, 4.1, 5, 6, \dots, 12\}$ to show our theorem. Let W_i denote the event that the experiment outputs true in G_i.

Roadmap: Before describing games, we briefly give intuitions for games.
In the PO security game, the game handles $\mathcal{Q}$ which contains distinct valid pairs of messages and signatures. The adversary wins if $\#\mathcal{Q} > q_S$ by submitting forgeries to the forgery-checking oracle Forge. As in the case of msEUF-CMA1 security, we start from the original game G_0 and go to G_3, in which the forgery-checking oracle outputs a special symbol if its own signature on m^* is invalid. We want to modify the winning condition that the adversary outputs at least $q_S + 1$ forgeries with the adversary succeeds in submitting $(m^*, (w^*, z^*))$ satisfying $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$. We manage the latter condition by a flag win as in msEUF-CMA1 security. In $\mathsf{G}_{4.0}$, we introduce this flag win and the game returns true if the adversary outputs at least $q_S + 1$ distinct valid pairs of message/signature ($\#\mathcal{Q} > q_S$) and the flag win is true. If a difference occurs between two games, then the adversary correctly guesses at least $q_S + 1$ signatures produced by GetTrans on *distinct* $q_S + 1$ messages. This means that the adversary correctly guesses $q_S + 1$ commitments w on distinct $q_S + 1$ messages. According to Proposition 4.1, the min-entropy of commitments is at least $\alpha - \lg(B)$ even if we know the whole table of the random oracle. Combining this with Lemma 2.3, we can show the bound on this event. In $\mathsf{G}_{4.1}$, we then remove $\mathcal{Q}$ by modifying the output of the game as win instead of $[\![\#\mathcal{Q} > q_S]\!] \land \mathsf{win}$. Since the adversary cannot detect this relaxation, the advantage in $\mathsf{G}_{4.1}$ at least the advantage in $\mathsf{G}_{4.0}$, i.e., $\Pr[W_{4.1}] \geq \Pr[W_{4.0}]$.
After that, we continue to modify the games as msEUF-CMA1 security. Since the adversary can access the *quantum* signing oracle, there are minor differences on the bound.

Game G_0*:* This is the original game. See Figure 11 for a concrete definition of G_0, where we expand the Sign algorithm. We have

$$\Pr[W_0] = \mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS}, \mathcal{A}}(\kappa).$$

Game G_1*:* We then replace PRF in P_1 of GetTrans with RF_P. The straightforward argument shows the following lemma. Unfortunately, this part is *memory-loose* because the reduction algorithm should maintain the forgery list Q.

Lemma 6.1. *There exists a quantum* $\mathcal{F}$*-oracle adversary* $\mathcal{A}_{\mathrm{prf}}$ *such that*

$$|\Pr[W_0] - \Pr[W_1]| \leq \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}_{\mathrm{prf}}}(\kappa),$$
$$\mathsf{Time}^*(\mathcal{A}_{\mathsf{PRF}}) = \mathsf{Time}(\mathcal{A}) + q_S \cdot O(B\mathsf{Time}(\mathsf{LID})) + q_F \cdot O(\mathsf{Time}(\mathsf{LID})),$$
$$\mathsf{Mem}^*(\mathcal{A}_{\mathsf{PRF}}) = \mathsf{Mem}(\mathcal{A}) + O(B\mathsf{Mem}(\mathsf{LID})) + q_F \cdot O(\mathsf{Mem}(\mathsf{LID})) + O(\lg(q_S)),$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C})$.

Game G_2*:* We next make GetTrans output all transcripts instead of the last one. This modification does not change anything, and we have

$$\mathsf{G}_1 = \mathsf{G}_2.$$

Game G_3*:* We next modify the forgery-checking oracle as follows: Before checking the validity of submitted query (m^*, σ^*), it generates its own signature $(\tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)})$ by using GetTrans(m^*). If the verification fails, that is, $\mathsf{V}(vk, \tilde{w}^{(k)}, \tilde{c}^{(k)}, \tilde{z}^{(k)}) = \mathsf{false}$, then the forgery-checking oracle returns the special symbol $\Finv$.
The adversary differentiates between the two games G_2 and G_3 if it submits such (m^*, σ^*) on which GetTrans(m^*) fails to output a valid signature. We can connect this event to the generic search problem with $\lambda = 1 - \rho'$. We here skip the proof and see the proof of sBU security (Lemma 5.2) for the detail.

Lemma 6.2. *Suppose that* LID *is* (γ, β)*-correct and has* (α, ϵ_m)*-commitment min-entropy. We have*

$$|\Pr[W_2] - \Pr[W_3]| \leq \Pr[\mathsf{Bad}_{m^*}] \leq 8(q_S + q_F + q_H + 1)^2(1 - \rho') + \epsilon_m.$$

Game $\mathsf{G}_{4.0}$*:* We replace the winning condition of $\mathcal{A}$ as follows: We introduce a flag win which is set true by Forge when the adversary queries $(w^*, z^*) \neq (\tilde{w}^{(k)}, \tilde{z}^{(k)})$. The challenger outputs $[\![\#Q > \mathsf{cnt}_s]\!] \wedge \mathsf{win}$. See $\mathsf{G}_{4.0}$ in Figure 11.

Lemma 6.3. *Suppose that* LID *has* (α, ϵ_m)*-commitment min-entropy. Then, we have*

$$|\Pr[W_3] - \Pr[W_{4.0}]| \leq (q_S + 1)/\lfloor 2^\alpha / B \rfloor + \epsilon_m.$$

Proof. The two games differ if the adversary queries at least $(q_S + 1)$ valid signatures on *distinct* messages to Forge such that $(w^*, z^*) = (\tilde{w}^{(k)}, \tilde{z}^{(k)})$ on each m^*. This is because if two valid signatures share the same message, then two signatures should be equivalent.
Let Bad_i be the event that the adversary in G_i queries $(q_S + 1)$ valid signatures on *distinct* messages to Forge such that $w^* = \tilde{w}^{(k)}$ on each m^*. By routine calculation, we have

$$|\Pr[W_3] - \Pr[W_{4.0}]| \leq \Pr[\mathsf{Bad}_3].$$

Proposition 4.1 shows that the min-entropy of $\tilde{w}^{(k)}$ on m^* is at least $\alpha - \lg(B)$ even if we know the whole table of the random oracle H with probability $1 - \epsilon_m$ over the choice of keys. Hence, we have $\Pr[\mathsf{Bad}_3] \leq (q_S + 1)/\lfloor 2^\alpha / B \rfloor + \epsilon_m$ by invoking Lemma 2.3. □

Game $\mathsf{G}_{4.1}$*:* We then replace the output of the game. In $\mathsf{G}_{4.1}$, the game just outputs the flag win instead of $[\![\#Q > \mathsf{cnt}_s]\!] \wedge \mathsf{win}$. See $\mathsf{G}_{4.1}$ in Figure 11. This modification allows us to forget Q.
Since we *relax* the condition and the adversary cannot detect this modification, we have

$$\Pr[W_{4.0}] \leq \Pr[W_{4.1}].$$

Game G_5*:* We next modify the random oracle as follows: On a query (m, w), the oracle first computes the transcripts $\{(w^{(i)}, c^{(i)}, z^{(i)})\}$ via GetTrans. If the input w is equivalent to one of $w^{(i)}$, then it returns $c' := c^{(i)}$; otherwise, it returns $c' := \mathsf{RF}_\mathsf{H}(m, w)$. See G_4 in Figure 11 for the details. Since $c^{(i)}$ is computed as $\mathsf{RF}_\mathsf{H}(m, w^{(i)})$, this modification changes nothing and we have

$$\mathsf{G}_{4.1} = \mathsf{G}_5.$$

Game G_6*:* The next game introduces a collision check for $w^{(i)}$'s in GetTrans. As Lemma 4.3, we have the following lemma.

Lemma 6.4. *Suppose that* LID *has* (α, ϵ_m)*-commitment min-entropy. Then, we have*

$$|\Pr[W_5] - \Pr[W_6]| \leq 2(q_S + q_H + q_F) \cdot B \cdot 2^{\frac{-\alpha-1}{2}} + \epsilon_m.$$

Game G_7*:* We next modify how to compute $c^{(k)}$ in GetTrans, in which it is computed as $\mathsf{RF}'_{\mathsf{H}}(m, k)$ instead of $\mathsf{RF}_{\mathsf{H}}(m, w^{(k)})$. We note that this does not change the adversary's view because $\mathsf{RF}'_{\mathsf{H}}$ is a random function, and if $w = w^{(i)}$ for the query (m, w), then consistent $c' = c^{(i)} = \mathsf{RF}'_{\mathsf{H}}(m, i)$ is output by H. (Note that excluding the collision is crucial [DFPS23].) By this argument, we have

$$\mathsf{G}_6 = \mathsf{G}_7.$$

Game G_8*:* We then modify the condition in Forge.
To ease the notation, let $\mathcal{L}_{m^*} := \{w^{(i)}\}_{i\in[k]}$ which are the w parts of the transcripts generated by GetTrans(m^*).
We additionally define $\mathcal{L}'_{m^*} := \{w^{(i)}\}_{i\in[k-1]}$.
In G_8, Forge additionally checks if $w^* \in \mathcal{L}'_{m^*}$ or not; if so, it also checks if $c^* = \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ or not as in Figure 11. As Lemma 4.4, we have the following lemma.

Lemma 6.5. *We have*

$$\Pr[W_7] = \Pr[W_8].$$

Game G_9*:* We next modify GetTrans to use Sim instead of P_1 and P_2. See G_9 in Figure 11 for the details. As Lemma 4.5 and Lemma 4.6 we have the following lemmas:

Lemma 6.6. *Assume that* LID *is* ϵ_{zk}*-HVZK. Then, we have*

$$|\Pr[W_8] - \Pr[W_9]| \le \sqrt{(6(q_S + q_H + q_F))^3 B\epsilon_{\mathrm{zk}}}.$$

Lemma 6.7. *Suppose that* LID *is* $(1 + \epsilon_{\mathrm{zk}})$*-divergence HVZK. Then, for any positive integer* ℓ*, we have*

$$\Pr[W_8] \le (1 + \epsilon_{\mathrm{zk}})^{B\ell}(\Pr[W_9] + 27q^3/\ell) + 27q^3/\ell.$$

Game G_{10}*:* We then treat the case $w^* = \tilde{w}^{(k)}$ as a special case. To do so, we replace the condition $w^* \notin \mathcal{L}'_{m^*}$ with $w^* \notin \mathcal{L}_{m^*}$ as in G_{10} in Figure 11. This is easily reduced to the CUR property.

Lemma 6.8. *There exists a quantum* $\mathcal{F}$*-oracle adversary* $\mathcal{A}_{\mathrm{cur}}$ *such that*

$$\begin{aligned}
&|\Pr[W_9] - \Pr[W_{10}]| \le \mathsf{Adv}^{\mathrm{cur}}_{\mathsf{LID},\mathcal{A}_{\mathrm{cur}}}(\kappa),\\
&\mathsf{Time}^*(\mathcal{A}_{\mathrm{cur}}) = O(\mathsf{Time}(\mathcal{A})) + (q_H + q_S + q_F) \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2),\\
&\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cur}}) = O(\mathsf{Mem}(\mathcal{A})) + O(B\mathsf{Mem}(\mathsf{LID})),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{Sim}})$*.*

Since the proof is the same as that of Lemma 4.7, we omit it.

Game G_{11}*:* We again modify the conditions in Forge: Forge checks if $(w^*, z^*) \ne (\tilde{w}^{(k)}, \tilde{z}^{(k)})$, $w^* \ne \tilde{w}^{(k)}$, and $c^* = \mathsf{RF}_{\mathsf{H}}(m^*, w^*)$ or not. If so, the flag is set as true. See G_{11} in Figure 11 for the details. As Lemma 4.8, this modification does not change anything and we have

$$\mathsf{G}_{10} = \mathsf{G}_{11}.$$

Game G_{12}*:* Finally, we replace a normal verification key with a lossy verification key. See G_{12} in Figure 11 for the details.
As Lemma 4.9, we have the following lemma:

Lemma 6.9. *There exists a quantum* $\mathcal{F}$*-oracle adversary* $\mathcal{A}_{\mathrm{ind}}$ *such that*

$$\begin{aligned}
&|\Pr[W_{11}] - \Pr[W_{12}]| \le \mathsf{Adv}^{\mathrm{indkey}}_{\mathsf{LID},\mathcal{A}_{\mathrm{ind}}}(\kappa),\\
&\mathsf{Time}^*(\mathcal{A}_{\mathrm{ind}}) = O(\mathsf{Time}(\mathcal{A})) + (q_H + q_S + q_F) \cdot O(B\mathsf{Time}(\mathsf{LID}) + B^2),\\
&\mathsf{Mem}^*(\mathcal{A}_{\mathrm{ind}}) = O(\mathsf{Mem}(\mathcal{A})) + O(B\mathsf{Mem}(\mathsf{LID})),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{W}, \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{C}) \times \mathsf{Func}(\mathcal{M} \times [B], \mathcal{R}_{\mathsf{Sim}})$*.*

We also have the following lemma as Kiltz et al. [KLS18]. See the proof of the case for sBU security (Lemma 5.11).

Lemma 6.10. *If* LID *is* ϵ_ℓ*-lossy, then we have*

$$\Pr[W_{13}] \le 8(q_S + q_H + q_F + 1)^2\epsilon_\ell.$$

Acknowledgements

The author thank the anonymous reviewers of EUROCRYPT 2024 for their constructive comments and suggestions.

References

AAC$^+$22. Gorjan Alagic, Daniel Apon, David Cooper, Quynh Dang, Thinh Dang, John Kelsey, Jacob Lichtinger, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, and Yi-Kai Liu. Status report on the third round of the NIST post-quantum cryptography standardization process. Technical report, NIST, 2022. 2

ACFK17. Benedikt Auerbach, David Cash, Manuel Fersch, and Eike Kiltz. Memory-tight reductions. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 101–132. Springer, Heidelberg, August 2017. 2, 8, 10

ADMP20. Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 411–439. Springer, Heidelberg, December 2020. 33, 34

AFLT12. Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-secure signatures from lossy identification schemes. In Pointcheval and Johansson [PJ12], pages 572–590. 2, 3, 4, 8, 14

AFLT16. Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly secure signatures from lossy identification schemes. *Journal of Cryptology*, 29(3):597–631, July 2016. 8, 9, 12

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, Heidelberg, August 2019. 16

AMRS18. Gorjan Alagic, Christian Majenz, Alexander Russell, and Fang Song. Quantum-secure message authentication via blind-unforgeability. Cryptology ePrint Archive, Report 2018/1150, 2018. https://eprint.iacr.org/2018/1150. 6, 37, 38

AMRS20. Gorjan Alagic, Christian Majenz, Alexander Russell, and Fang Song. Quantum-access-secure message authentication via blind-unforgeability. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 788–817. Springer, Heidelberg, May 2020. 3, 6, 10, 11, 37

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 Handschuh and Lysyanskaya [HL23], pages 358–389. 4, 6

BCD$^+$22. Markus Bläser, Zhili Chen, Dung Hoang Duong, Antoine Joux, Ngoc Tuong Nguyen, Thomas Plantard, Youming Qiao, Willy Susilo, and Gang Tang. On digital signatures based on isomorphism problems: QROM security, ring signatures, and applications. Cryptology ePrint Archive, Report 2022/1184, 2022. https://eprint.iacr.org/2022/1184. 33, 34

BDF$^+$11. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, December 2011. 2, 5, 39

Bel93. Eric David Belsley. *Rates of convergence of Markov chains related to association schemes.* PhD thesis, Harvard University, May 1993. 37

Ber11. Daniel J. Bernstein. Extending the Salsa20 nonce. In *SKEW 2011 (Symmetric Key Encryption Workshop 2011)*, 2011. See the authors' website or http://skew2011.mat.dtu.dk/program.html. 2

Bha20. Rishiraj Bhattacharyya. Memory-tight reductions for practical key encapsulation mechanisms. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 249–278. Springer, Heidelberg, May 2020. 2

BKV19. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Heidelberg, December 2019. 34

BLS01. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, December 2001. 2

BPS16. Mihir Bellare, Bertram Poettering, and Douglas Stebila. From identification to signatures, tightly: A framework and generic transforms. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 435–464. Springer, Heidelberg, December 2016. 12

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993. 2

BR96. Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 399–416. Springer, Heidelberg, May 1996. 2, 38

BZ13a. Dan Boneh and Mark Zhandry. Quantum-secure message authentication codes. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 592–608. Springer, Heidelberg, May 2013. 38

BZ13b. Dan Boneh and Mark Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 361–379. Springer, Heidelberg, August 2013. 2, 3, 4, 5, 7, 10, 32, 41, 42

CCLM22. Rohit Chatterjee, Kai-Min Chung, Xiao Liang, and Giulio Malavolta. A note on the post-quantum security of (ring) signatures. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part II*, volume 13178 of *LNCS*, pages 407–436. Springer, Heidelberg, March 2022. 3, 5, 38, 39, 42

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, Heidelberg. 3

CKMS16. Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar. Another look at tightness II: Practical issues in cryptography. In Raphael C.-W. Phan and Moti Yung, editors, *Mycrypt 2016*, volume 10311 of *LNCS*, pages 21–55. Springer, 2016. 2

CLM$^+$18. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Heidelberg, December 2018. 34

CMS12. Sanjit Chatterjee, Alfred Menezes, and Palash Sarkar. Another look at tightness. In Ali Miri and Serge Vaudenay, editors, *SAC 2011*, volume 7118 of *LNCS*, pages 293–319. Springer, Heidelberg, August 2012. 2

Cor00. Jean-Sébastien Coron. On the exact security of full domain hash. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 229–235. Springer, Heidelberg, August 2000. 2

DDKA21. Mina Doosti, Mahshid Delavar, Elham Kashefi, and Myrto Arapinis. A unified framework for quantum unforgeability. *CoRR*, abs/2103.13994, 2021. 3

DFPS22. Julien Devevey, Omar Fawzi, Alain Passelègue, and Damien Stehlé. On rejection sampling in lyubashevsky's signature scheme. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 34–64. Springer, Heidelberg, December 2022. 35

DFPS23. Julien Devevey, Pouria Fallahpour, Alain Passelègue, and Damien Stehlé. A detailed analysis of fiat-shamir with aborts. In Handschuh and Lysyanskaya [HL23], pages 327–357. 3, 4, 5, 6, 9, 10, 12, 14, 16, 22, 27

DGJL21. Denis Diemert, Kai Gellert, Tibor Jager, and Lin Lyu. Digital signatures with memory-tight security in the multi-challenge setting. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 403–433. Springer, Heidelberg, December 2021. 2, 3, 4, 5, 12, 13, 14, 39

Din20. Itai Dinur. Tight time-space lower bounds for finding multiple collision pairs and their applications. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 405–434. Springer, Heidelberg, May 2020. 2

dPRS23. Rafaël del Pino, Thomas Prest, Mélissa Rossi, and Markku-Juhani O. Saarinen. High-order masking of lattice signatures in quasilinear time. In *2023 IEEE Symposium on Security and Privacy*, pages 1168–1185. IEEE Computer Society Press, May 2023. 4, 6

DPS23. Julien Devevey, Alain Passelègue, and Damien Stehlé. G+G: A Fiat-Shamir lattice signature based on convolved Gaussians. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VII*, volume 14444 of *LNCS*, pages 37–64. Springer, 2023. 3, 4, 6, 35, 36

EKP20. Ali El Kaafarani, Shuichi Katsumata, and Federico Pintore. Lossy CSI-FiSh: Efficient signature scheme with tight reduction to decisional CSIDH-512. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 157–186. Springer, Heidelberg, May 2020. 33, 34, 35

FG15. Jason Fulman and Larry Goldstein. Stein's method and the rank distribution of random matrices over finite fields. *The Annals of Probability*, 43(3):1274–1314, May 2015. 37

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987. 2, 3

GGJT22. Ashrujit Ghoshal, Riddhi Ghosal, Joseph Jaeger, and Stefano Tessaro. Hiding in plain sight: Memory-tight proofs via randomness programming. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 706–735. Springer, Heidelberg, May / June 2022. 2, 8, 39

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, Heidelberg, December 2021. 6

GJT20. Ashrujit Ghoshal, Joseph Jaeger, and Stefano Tessaro. The memory-tightness of authenticated encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 127–156. Springer, Heidelberg, August 2020. 2

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 2, 38

GT20. Ashrujit Ghoshal and Stefano Tessaro. On the memory-tightness of hashed ElGamal. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 33–62. Springer, Heidelberg, May 2020. 2

HL23. Helena Handschuh and Anna Lysyanskaya, editors. *CRYPTO 2023, Part V*, volume 14085 of *LNCS*. Springer, Heidelberg, August 2023. 28, 29

HRS16. Andreas Hülsing, Joost Rijneveld, and Fang Song. Mitigating multi-target attacks in hash-based signatures. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 387–416. Springer, Heidelberg, March 2016. 7

JK22. Joseph Jaeger and Akshaya Kumar. Memory-tight multi-challenge security of public-key encryption. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part III*, volume 13793 of *LNCS*, pages 454–484. Springer, Heidelberg, December 2022. 2, 6

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, Heidelberg, April / May 2018. 3, 4, 5, 6, 7, 9, 10, 12, 14, 17, 23, 24, 27

KM07. Neal Koblitz and Alfred J. Menezes. Another look at "provable security". *Journal of Cryptology*, 20(1):3–37, January 2007. 2

KM15. Neal Koblitz and Alfred J. Menezes. The random oracle model: a twenty-year retrospective. *Des. Codes Cryptogr.*, 77:587–610, 2015. 12

KX24. Haruhisa Kosuge and Keita Xagawa. Probabilistic hash-and-sign with retry in the quantum random oracle model. In *PKC 2024*, 2024. To appear. See https://eprint.iacr.org/2022/1359. 5, 6

Lyu09. Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009. 3, 35

Lyu12. Vadim Lyubashevsky. Lattice signatures without trapdoors. In Pointcheval and Johansson [PJ12], pages 738–755. 35

MMO04. Theresa Migler, Kent E. Morrison, and Mitchell Ogle. Weight and rank of matrices over finite fields. [arXiv:math/0403314], 2004. https://arxiv.org/abs/math/0403314. 37

MMO21. Christian Majenz, Chanelle Matadah Manfouo, and Maris Ozols. Quantum-access security of the winternitz one-time signature scheme. In Stefano Tessaro, editor, *2nd Conference on Information-Theoretic Cryptography, ITC 2021, July 23-26, 2021, Virtual Conference*, volume 199 of *LIPIcs*, pages 21:1–21:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 3

MNPV99. David M'Raïhi, David Naccache, David Pointcheval, and Serge Vaudenay. Computational alternatives to random number generators. In Stafford E. Tavares and Henk Meijer, editors, *SAC 1998*, volume 1556 of *LNCS*, pages 72–80. Springer, Heidelberg, August 1999. 12

PJ12. David Pointcheval and Thomas Johansson, editors. *EUROCRYPT 2012*, volume 7237 of *LNCS*. Springer, Heidelberg, April 2012. 28, 30

RSA78. Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, February 1978. 2

vEH14. Tim van Erven and Peter Harremos. Rényi divergence and Kullback-Leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014. 7

WMHT18. Yuyu Wang, Takahiro Matsuda, Goichiro Hanaoka, and Keisuke Tanaka. Memory lower bounds of reductions revisited. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 61–90. Springer, Heidelberg, April / May 2018. 2

YTA23. Quan Yuan, Mehdi Tibouchi, and Masayuki Abe. Quantum-access security of hash-based signature schemes. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *ACISP 23*, volume 13915 of *LNCS*, pages 343–380. Springer, Heidelberg, July 2023. 3

Zha12a. Mark Zhandry. How to construct quantum random functions. In *53rd FOCS*, pages 679–687. IEEE Computer Society Press, October 2012. 7

Zha12b. Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, Heidelberg, August 2012. 32

A Lemma for the Rényi Divergence

To treat the divergence HVZK, we show a variant of Lemma 2.2 to treat the Rényi divergence.

Lemma A.1 (A variant of Lemma 2.2). *Let $\mathcal{X}$ and $\mathcal{Y}$ be two finite sets. Let $D = \{D_x\}$ and $D' = \{D'_x\}$ be two sets of efficiently sampleable distributions over $\mathcal{Y}$ indexed by $x \in \mathcal{X}$. Let $\mathcal{A}$ be a quantum algorithm making q queries to an oracle $H : \mathcal{X} \to \mathcal{Y}$. Suppose that there exists some rational $\epsilon \in [0, +\infty)$ such that for each $x \in X$, $R_\infty(D_x; D'_x) \leq (1+\epsilon)$ holds. Then, there exists a universal constant $C_0 < 27$ such that, for any positive ℓ, $\Pr[H \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D) : \mathcal{A}^{|H\rangle} = 1] \leq (1+\epsilon)^\ell (\Pr[H \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D') : \mathcal{A}^{|H\rangle} = 1] + C_0 q^3/\ell) + C_0 q^3/\ell$ holds.*

Before giving the proof of the lemma, we quickly review the proof of Lemma 2.2.

Quick review of the proof of Lemma 2.2: The strategy of Boneh and Zhandry [BZ13b] is summarized as follows: First, they showed that Lemma 2.2 for the sets D and D' of *rational* distributions and rational ϵ. Second, for general distributions, they considered the sets of sequences of *rational* distributions approximating the target sets of distributions D and D'. To show the former, they used the following definition and lemma to prove Lemma 2.2.

Definition A.1 ([BZ13b, Definition 2.3] taken from [Zha12b]). *Fix sets $\mathcal{X}$, $\mathcal{Y}$ and a distribution D on $\mathcal{Y}$. Fix an integer r. Let $\boldsymbol{y} = (y_1, \ldots, y_r)$ be a list of r samples from D and let P be a random function from $\mathcal{X}$ to $[r]$. The distributions on $\boldsymbol{y}$ and P induce a distribution on functions $H : \mathcal{X} \to \mathcal{Y}$ defined by $H(x) := \boldsymbol{y}[P(x)] = y_{P(x)}$. This distribution is called* a small-range distribution *with r samples of D.*

Lemma A.2 ([BZ13b, Lemma 2.4] taken from [Zha12b]). *There is a universal constant $C_0 = 27$ such that, for any sets $\mathcal{X}$ and $\mathcal{Y}$, distribution D on $\mathcal{Y}$, any integer ℓ, and any quantum algorithm $\mathcal{A}$ making q queries to an oracle $H : \mathcal{X} \to \mathcal{Y}$, the following two cases are indistinguishable, except with probability less than $C_0 q^3/\ell$:*

- *$H(x) = y_x$ where $\boldsymbol{y}$ is a list of samples of D of size $\#\mathcal{X}$.*
- *H is drawn from the small-range distribution with ℓ samples of D.*

To show Lemma 2.2 for the sets D and D' of *rational* distributions and rational ϵ, they defined two distributions E and E' over a set $[K]$ and a set of functions $f = \{f_x : [K] \to \mathcal{Y}\}$ such that $D_x = f_x \circ E$, $D'_x = f_x \circ E'$, and $|E - E'| \leq \epsilon$. Using them, they defined four games as follows:

- G_0: Choose $\boldsymbol{z} \leftarrow E^{\#\mathcal{X}}$; for each x, define $g(x) := \boldsymbol{z}[x]$ and $H(x) := f_x(g(x))$; return $\mathcal{A}^{|H\rangle}$.
- G_1: Choose $\boldsymbol{z} \leftarrow E^\ell$; choose $P \leftarrow \mathsf{Func}(\mathcal{X}, [\ell])$; for each x, define $g(x) := \boldsymbol{z}[P(x)]$ and $H(x) := f_x(g(x))$; return $\mathcal{A}^{|H\rangle}$.
- G_2: Choose $\boldsymbol{z} \leftarrow (E')^\ell$; choose $P \leftarrow \mathsf{Func}(\mathcal{X}, [\ell])$; for each x, define $g(x) := \boldsymbol{z}[P(x)]$ and $H(x) := f_x(g(x))$; return $\mathcal{A}^{|H\rangle}$.
- G_3: Choose $\boldsymbol{z} \leftarrow (E')^{\#\mathcal{X}}$; for each x, define $g(x) := \boldsymbol{z}[x]$ and $H(x) := f_x(g(x))$; return $\mathcal{A}^{|H\rangle}$.

By those definitions, we can see g in G_1 and G_2 are drawn from the small-range distribution with ℓ samples of E and E'. Applying Lemma A.2 to G_0 and G_1 (and G_2 and G_3, resp.), we have $|\Pr[E_0] - \Pr[E_1]| \leq C_0 q^3/\ell$ (and $|\Pr[E_2] - \Pr[E_3]| \leq C_0 q^3/\ell$, resp.). The distance between G_1 and G_2 is at most $\ell \cdot |E - E'| \leq \ell\epsilon$.
Taking $\ell = \sqrt{2C_0 q^3/\epsilon}$, the distance between G_0 and G_3 is at most $\ell\epsilon + 2C_0 q^3/\ell = \sqrt{2C_0 q^3 \epsilon} + \sqrt{2C_0 q^3 \epsilon} = \sqrt{8C_0 q^3 \epsilon}$.
This proved Lemma 2.2 for the sets D and D' of *rational* distributions and rational ϵ.

Our proof: Following their proof strategy, we proved Lemma A.1 for the sets D and D' of *rational* distributions and rational ϵ.

Proposition A.1 (Lemma A.1, Rational Distributions). *Suppose that the probabilities in each distribution in D and D' are* rational *and ϵ is* rational. *Then, there exists a universal constant $C_0 < 27$ such that, for any positive ℓ, we have* $\Pr[H \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D) : \mathcal{A}^{|H\rangle} = 1] \leq (1+\epsilon)^\ell (\Pr[H \leftarrow \mathsf{Func}_{\mathcal{X},\mathcal{Y}}(D') : \mathcal{A}^{|H\rangle} = 1] + C_0 q^3/\ell) + C_0 q^3/\ell$.

Proof. By the assumption, for any x and y, $D_x(y)$ and $D'_x(y)$ is rational.
We take large enough integers K and $K' := (1+\epsilon)K$ such that $K \cdot D_x(y)$ and $K' \cdot D'_x(y)$ are also integers for all x and y. For ease of notation, we let $p_{x,y} = K \cdot D_x(y)$ and $p'_{x,y} = K \cdot D'_x(y)$, which implies $(1+\epsilon)p'_{x,y} = K' \cdot D'_x(y) \in \mathbb{Z}$.
We design f_x, E, and E' as follows: Define E and E' as the uniform distributions over $[K]$ and $[K']$, respectively.
For each $x \in \mathcal{X}$, define $f_x : [K'] \to Y$ as follows:

- for $i \in [K]$: we assign $p_{x,y}$ elements for each $y \in \mathcal{Y}$.
- for $i \in \{K+1, \ldots, K'\}$: we assign $(1+\epsilon)p'_{x,y} - p_{x,y}$ elements for each $y \in \mathcal{Y}$.

We have $p_{x,y} \leq (1+\epsilon)p'_{x,y}$ since $R_\infty(D_x; D'_x) = \sup_{y \in \mathrm{Supp}(D_x)} D_x(y)/D'_x(y) = \sup(p_{x,y}/K)/(p'_{x,y}/K) = \sup p_{x,y}/p'_{x,y}$, which is at most $1+\epsilon$. Hence, f_x is well-defined. By the definition, $f_x \circ E$ and $f_x \circ E'$ are equivalent to D_x and D'_x, respectively, as we wanted. In addition, the definitions of E and E' yield that $R_\infty(E; E') = (1/K)/(1/K') = (1+\epsilon)$.
Using them, we define four games as follows:

- G_0: Choose $z \leftarrow E^{\#\mathcal{X}}$; define $H(x) := f_x(z[x])$; return $\mathcal{A}^{|H\rangle}$.
- G_1: Choose $z \leftarrow E^{\ell}$; choose $P \leftarrow \mathsf{Func}(\mathcal{X}, [\ell])$; define $H(x) := f_x(z[P(x)])$; return $\mathcal{A}^{|H\rangle}$.
- G_2: Choose $z \leftarrow (E')^{\ell}$; choose $P \leftarrow \mathsf{Func}(\mathcal{X}, [\ell])$; define $H(x) := f_x(z[P(x)])$; return $\mathcal{A}^{|H\rangle}$.
- G_3: Choose $z \leftarrow (E')^{\#\mathcal{X}}$; define $H(x) := f_x(z[x])$; return $\mathcal{A}^{|H\rangle}$.

For any event E_i in G_i, we have

- $|\Pr[E_0] - \Pr[E_1]| \leq C_0 q^3/\ell$ from Lemma A.2,
- $\Pr[E_1] \leq R_\infty(E; E')^{\ell} \Pr[E_2] \leq (1+\epsilon)^{\ell} \Pr[E_2]$ from Lemma 2.1, and
- $|\Pr[E_2] - \Pr[E_3]| \leq C_0 q^3/\ell$ from from Lemma A.2.

Combining them, we obtain

$$\begin{aligned}\Pr[E_0] &\leq \Pr[E_1] + C_0 q^3/\ell \\ &\leq (1+\epsilon)^{\ell} \Pr[E_2] + C_0 q^3/\ell \\ &\leq (1+\epsilon)^{\ell} (\Pr[E_3] + C_0 q^3/\ell) + C_0 q^3/\ell.\end{aligned}$$

□

For example, if we take $\epsilon = a/\ell$ for some positive constant a, we have $(1+\epsilon)^{\ell} = (1+a/\ell)^{\ell} \to e^a$ with $\ell \to +\infty$. By letting $\ell = C_0 q^3/\delta$ for some negligible δ, we have

$$\Pr[E_0] \leq (1+\epsilon)^{\ell}(\Pr[E_3] + C_0 q^3/\ell) + C_0 q^3/\ell \leq e^a(\Pr[E_3] + \delta) + \delta.$$

Let δ be a negligible function. We then set $\ell = C_0 q^3/\delta$ and $\epsilon = \delta \cdot a/C_0 q^3$, which is negligible in 1^κ.

B Instantiations of Lossy Identification

We review three instantiations of lossy identification schemes from post-quantum assumptions.

B.1 Lossy Identification Scheme based on Pseudorandom Group Action

Bläser et al. [BCD+22] gave a lossy identification scheme, which is an abstraction of $\mathsf{ID}_{\mathsf{ls}}^{\mathsf{enCh}}$ in [EKP20], based on pseudorandom group action. We briefly review cryptographic group action [ADMP20].

Definition B.1 (Group action). *Let G be a group with identity element 1_G. Let X be a set. A map $\star : G \times X \to X$ is* a group action *if, for all $g, h \in G$ and $x \in X$, $1_G \star x = x$ and $(gh) \star x = g \star (h \star x)$.*

In this section, we assume that G and X are finite. For the security of the LID scheme, we require the hardness of the following problem, which is an adapted version of [BCD+22, Definition 6].

Definition B.2 (S-pseudorandom problem, adapted). *Let S be a positive integer. Let $(G, X, \star)$ be a group action. The* S-pseudorandom *problem with parameter S asks to distinguish between the following two distributions:*

- *$(E, g_1 \star E, g_2 \star E, \dots, g_S \star E)$, where $E \leftarrow X$ and $g_1, \dots, g_S \leftarrow G$.*
- *$(E, E_1, E_2, \dots, E_S)$ where $E, E_1, \dots, E_S \leftarrow X$.*

For CUR property, we will use the following problem [BCD+22, Definition 7]:

Definition B.3 (Stabilizer problem). *Let $(G, X, \star)$ be a regular group action. The* stabilizer *problem is, given a random element $E \leftarrow X$, finding a non-trivial stabilizer $g \in G \setminus \{1_G\}$ satisfying $g \star E = E$. The $\mathsf{Stab}_{(G,X,\star)}$ assumption states that for any QPT adversary $\mathcal{A}$, its advantage*

$$\mathsf{Adv}_{\mathsf{Stab},\mathcal{A}}(\kappa) := \Pr[E \leftarrow X, g \leftarrow \mathcal{A}(G, X, \star, E) : g \star E = E \wedge g \in G \setminus \{1_G\}]$$

is negligible in κ.

The description of the scheme follows:

Public parameter: The public parameter is a cryptographic group action $(G, X, \star)$. We have $\mathcal{W} := X^t$ and $\mathcal{Z} := G^t$.

Key generation: $\mathsf{Gen}_{\mathsf{LID}}$ uniformly samples $E_0 \leftarrow X$ and $g_1, \dots, g_S \leftarrow G$, lets $g_0 := 1_G$, and outputs

$$vk = (E_0, E_1, \dots, E_S) \text{ and } sk = (vk, g_0, g_1, \dots, g_S),$$

where $E_i := g_i \star E_0$ for $i = 1, \dots, S$.

Lossy key generation: $\mathsf{LossyGen}_{\mathsf{LID}}$ uniformly samples $E_0, \dots, E_S \leftarrow X$ and outputs

$$vk = (E_0, E_1, \dots, E_S).$$

Challenge space: The challenge space is $C := \{0, 1, \dots, S\}^t$.

Prover: The prover's algorithms are defined as follows:

- $\mathsf{P}_1(sk)$ uniformly samples $r_1, \dots, r_t \leftarrow G$ and returns a commitment $w = (W_1, \dots, W_t)$, where $W_i := r_i \star E_0$ for $i = 1, \dots, t$, and outputs a state information $s := (r_1, \dots, r_t)$.
- $\mathsf{P}_2(sk, w, c, s)$, where $c = (c_1, \dots, c_t)$ and $s = (r_1, \dots, r_t)$, computes $z_i := r_i \cdot g_{c_i}^{-1} \in G$ for $i = 1, \dots, t$ and returns $z = (z_1, \dots, z_t)$.

Reconstruction: $\mathsf{Rec}(vk, c, z)$ computes $W_i = z_i \star E_{c_i}$ for $i = 1, \dots, t$ and returns $w = (W_1, \dots, W_t)$.

Verifier: $\mathsf{V}(vk, w, c, z)$ checks if $w = \mathsf{Rec}(vk, c, z)$ or not.

Simulator: $\mathsf{Sim}(vk, c)$ uniformly samples $z = (z_1, \dots, z_t) \leftarrow G^t$ and outputs $w = \mathsf{Rec}(vk, c, z)$.

The signature scheme is obtained by applying $\mathsf{DFS}_{1,cz}$ to the above lossy identification scheme.

The protocol is ϵ_ℓ-lossy with $\epsilon_\ell = \frac{1}{(S+1)^t} \cdot \prod_{i\in[S]} \frac{|X|-i|G|}{|X|} + (1 - \prod_{i\in[S]} \frac{|X|-i|G|}{|X|})$ [BCD+22, Lemma 4]. This ϵ_ℓ is negligible in κ when S is constant, $t = \omega(\lg(\kappa))$, and $|X| \gg |G|$. Key indistinguishability follows from the hardness of the S-pseudorandom problem of the underlying group action. In addition, parameters of the commitment min-entropy are $\alpha = t \cdot \lg(N)$ and $\epsilon_m = 0$. The protocol achieves perfect correctness and perfect HVZK.

We give the proof of CUR to check the memory usage of the reduction algorithm. The underlying problem is the stabilizer problem,

Lemma B.1. *The protocol has the CUR property under the* $\mathsf{Stab}_{(G,X,\star)}$ *assumption: Precisely speaking, for a quantum adversary $\mathcal{A}$ breaking the CUR property of the identification protocol, there exists a quantum adversary $\mathcal{A}_{\mathrm{stab}}$ against the* $\mathsf{Stab}_{(G,X,\star)}$ *assumption such that*

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{cur}}_{\mathsf{LID},\mathcal{A}}(\kappa) &\leq \mathsf{Adv}_{\mathsf{Stab},\mathcal{A}_{\mathrm{stab}}}(\kappa),\\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{stab}}) &= \mathsf{Time}(\mathcal{A}) + S \cdot O(|G|, |X|),\\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{stab}}) &= \mathsf{Mem}(\mathcal{A}) + S \cdot O(|G|, |X|).
\end{aligned}$$

Proof. We construct $\mathcal{A}_{\mathrm{stab}}$ as follows: Given E chosen from X, it samples $g_1, \dots, g_S \leftarrow G$, computes $E_i := g_i \star E$, and runs $\mathcal{A}$ on input $(E_0 := E, E_1, E_2, \dots, E_S)$. The adversary $\mathcal{A}$ outputs w, c, z and ζ. Since $z \neq \zeta$, there exists $i \in [t]$ satisfying $z_i \neq \zeta_i$. $\mathcal{A}_{\mathrm{stab}}$ outputs $f := g_{c_i}^{-1}\zeta_i^{-1}z_i g_{c_i}$ if it is not 1_G.

Let us check that f is a stabilizer. If (w, c, z) and (w, c, ζ) are valid, then we have $W_i = z_i \star E_{c_i} = \zeta_i \star E_{c_i}$. Putting $E_{c_i} = g_{c_i} \star E$, we have $(z_i g_{c_i}) \star E = (\zeta_i g_{c_i}) \star E$. Thus, $f \star E = \big((\zeta_i g_{c_i})^{-1} \cdot (z_i g_{c_i})\big) \star E = E$ and f is a stabilizer for E. It is easy to check that this f is non-trivial: Since $z_i \neq \zeta_i$, we have $z_i g_{c_i} \neq \zeta_i g_{c_i}$ and $f = (\zeta_i g_{c_i})^{-1} \cdot z_i g_{c_i} \neq 1_G$ as we wanted. Hence, the advantage of $\mathcal{A}_{\mathrm{stab}}$ is equivalent to that of $\mathcal{A}$. □

B.2 Lossy Identification Scheme based on CSIDH

We recall a lossy identification scheme $\mathsf{ID}^{\mathsf{denCh}}_{\mathsf{ls}}$ in Lossy CSI-FiSh proposed by El Kaafarani, Katsumata, and Pintore [EKP20], which is based on the hardness of the decisional Diffie-Hellman problems in the CSIDH setting.

We briefly review cryptographic group action [ADMP20] and quadratic twist.

Definition B.4 (Regulality of group action). *We say that a group action $(G, X, \star)$ is* regular *if the following two conditions hold: (transitive:) for every $x, x' \in X$, there exists $g \in G$ satisfying $x' = g \star x$. (free:) for each group element $g \in G$, $g = 1_G$ if and only if there exists some element $x \in X$ such that $x = g \star x$.*

In what follows, we assume that $G = \langle g \rangle$ of cardinality N. In the CSIDH setting, given $E = g^a \star E_0$, we can compute its quadratic twist $\mathsf{twist}(E)$, which is $g^{-a} \star E_0$ [CLM+18, BKV19, EKP20].

For the security of the LID scheme, we require the hardness of the following problem, which is an adapted version of [EKP20, Definition 4.1].

Definition B.5 (Fixed-Curve Multi-Decisional GADH problem). *Let S be a positive integer. Suppose that $G = \langle g \rangle$ of cardinarity N and X be a finite set. Let $(G, X, \star)$ be a regular group action. The* fixed-curve multi-decisional group-action Diffie-Hellman (FCMD-GADH) *problem with parameter S asks to distinguish between the following two distributions:*

- *$(E, H, g^{a_1} \star E, g^{a_1} \star H, \dots, g^{a_S} \star E, g^{a_S} \star H)$, where $E, H \leftarrow X$ and $a_1, \dots, a_S \leftarrow \mathbb{Z}_N$.*
- *$(E, H, E'_1, H'_1, \dots, E'_S, H'_S)$ where $E, H, E'_1, H'_1, \dots, E'_S, H'_S \leftarrow X$.*

If $S = 1$, the problem is said to be the decisional group-action Diffie-Hellman (D-GADH) problem.

The description of the scheme follows:

Public parameter: The public parameter is a cryptographic group action $(G, X, \star)$. Let $E_0 \in X$ be a fixed element in X. We have $\mathcal{W} := (X^2)^t$ and $\mathcal{Z} := \mathbb{Z}_N^t$.

Key generation: $\mathsf{Gen}_{\mathsf{LID}}$ uniformly samples $a_1, \dots, a_S, b_1, b_2 \leftarrow \mathbb{Z}_N$, lets $a_0 := 0$, and outputs

$$vk = \{(E_1^{(i)}, E_2^{(i)})\}_{i \in \{0,\dots,S\}} \text{ and } sk = (vk, a_0, a_1, \dots, a_S, b_1, b_2),$$

where $E_1^{(0)} := g^{b_1} \star E_0$, $E_2^{(0)} := g^{b_2} \star E_0$, $E_0^{(i)} = g^{a_i} \star E_0^{(0)}$, and $E_1^{(i)} = g^{a_i} \star E_1^{(0)}$ for $i = 1, \dots, S$. For ease of notation, for $\beta \in \{1, 2\}$ and $i \in [S]$, we define $E_\beta^{(-i)} := \mathsf{twist}(E_\beta^{(i)})$.

Lossy key generation: $\mathsf{LossyGen}_{\mathsf{LID}}$ uniformly samples $a_1, \dots, a_S$, $a'_1, \dots, a'_S$, $b_1, b_2 \leftarrow \mathbb{Z}_N$ and outputs

$$vk = \{(E_1^{(i)}, E_2^{(i)})\}_{i \in \{0,\dots,S\}},$$

where $E_1^{(0)} := g^{b_1} \star E_0$, $E_2^{(0)} := g^{b_2} \star E_0$, $E_0^{(i)} = g^{a_i} \star E_0^{(0)}$, and $E_1^{(i)} = g^{a'_i} \star E_1^{(0)}$ for $i = 1, \dots, S$.

Challenge space: The challenge space is $C := \{-S, -S+1, \dots, S-1, S\}^t$.

Prover: The prover's algorithms are defined as follows:

- $\mathsf{P}_1(sk)$ unformly samples $r_1, \dots, r_t \leftarrow \mathbb{Z}_N$ and returns a commitment $w = (w_1, \dots, w_t) = \{(F_1^{(k)}, F_2^{(k)})\}_{i \in [t]}$, where $(F_1^{(k)}, F_2^{(k)}) := (g^{r_k} \star E_1^{(0)}, g^{r_k} \star E_2^{(0)})$ and outputs a state information $s := (r_1, \dots, r_t)$.
- $\mathsf{P}_2(sk, w, c, s)$, where $c = (c_1, \dots, c_t)$ and $s = (r_1, \dots, r_t)$, computes, for $k \in [t]$, $z_k = r_k - a_{c_k} \in \mathbb{Z}_N$ if $c_k \geq 0$ and $z_k = r_k + b_1 + b_2 + a_{|c_k|}$ otherwise and returns $z = (z_1, \dots, z_t)$.

Reconstruction: $\mathsf{Rec}(vk, c, z)$ computes, for $k \in [t]$, $w_k = (g^{z_k} \star E_1^{(c_k)}, g^{z_k} \star E_2^{(c_k)})$ if $c \geq 0$ and $(g^{z_k} \star E_2^{(c_k)}, g^{z_k} \star E_1^{(c_k)})$ otherwise and returns $w = (w_1, \dots, w_t)$.

Verifier: $\mathsf{V}(vk, w, c, z)$ checks if $w = \mathsf{Rec}(vk, c, z)$ or not.

Simulator: $\mathsf{Sim}(vk, c)$ uniformly samples $z = (z_1, \dots, z_t) \leftarrow \mathbb{Z}_N^t$ and outputs $w = \mathsf{Rec}(vk, c, z)$.

The signature scheme Lossy CSI-FiSh is obtained by applying $\mathsf{DFS}_{1,cz}$ to the above lossy identification scheme. El Kaafarani et al. showed that the protocol is ϵ_ℓ-lossy with $\epsilon_\ell = \frac{1}{(2S+1)^t} \cdot \prod_{i \in [S]} \frac{N-i}{N} + (1 - \prod_{i \in [S]} \frac{N-i}{N})$ [EKP20, Lemma 4.7]. This ϵ_ℓ is negligible in κ when S is constant and $t = \omega(\lg(\kappa))$. Key indistinguishability follows from the hardness of the FCMD-GADH problem. The protocol achieves perfect correctness, perfect HVZK, and perfect unique response. In addition, parameters of the commitment min-entropy are $\alpha = t \cdot \lg(N)$ and $\epsilon_m = 0$.

B.3 Lossy Identification Scheme based on Lattices

As an example of lossy identification based on lattices, we take a new scheme G+G proposed by Devevey, Passelègue, and Stehlé [DPS23] instead of [Lyu09, Lyu12, DFPS22]. We first define the Gaussian function with covariance parameter $\Sigma \in \mathbb{R}^{k \times k}$, which is a positive-definite symmetric matrix, and center parameter $c \in \mathbb{R}^k$ as

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

For a lattice $\Lambda \subseteq \mathrm{Span}(\Sigma)$, the Gaussian distribution over Λ with covariance parameter Σ and center parameter c is defined by a probability mass function

$$D_{\Lambda,\Sigma,c}(x) = \frac{\rho_{\Sigma,c}(x)}{\sum_{y \in \Lambda} \rho_{\Sigma,c}(y)} \text{ for } x \in \Lambda.$$

Definition B.6 (Learning With Errors (LWE), Hermite Normal Form). *Let $m, k, q \in \mathbb{Z}^+$ with $q \geq 2$. Let χ be a distribution over $\mathbb{Z}$. The $\mathsf{LWE}_{m,k,\ell,q,\chi}$ assumption states that for any QPT adversary $\mathcal{A}$, the following two distributions are computationally indistinguishable:*

$$D_1 : A \leftarrow \mathbb{Z}_q^{m \times k}; S \leftarrow \chi^{k \times \ell}; E \leftarrow \chi^{m \times \ell};\ \textit{return}\ (A, AS + E),$$
$$D_2 : A \leftarrow \mathbb{Z}_q^{m \times k}; U \leftarrow \mathbb{Z}_q^{m \times \ell};\ \textit{return}\ (A, U).$$

Definition B.7 (Short Integer Solution). *Let $m, k, q \in \mathbb{Z}^+$ with $q \geq 2$. Let $\gamma > 0$. The $\mathsf{SIS}_{m,k,q,\gamma}$ assumption states that for any QPT adversary $\mathcal{A}$, its advantage*

$$\mathsf{Adv}_{\mathsf{SIS},\mathcal{A}}(\kappa) := \Pr[A \leftarrow \mathbb{Z}_q^{m \times k}, x \leftarrow \mathcal{A}(A) : Ax \equiv 0 \pmod{q} \wedge x \neq 0 \wedge \|x\| \leq \gamma]$$

is negligible in κ.

The description of the LID scheme follows:

Public parameter: The public parameters are $m \geq \ell > 0$, $k > m + \ell$, and $C \subseteq \mathbb{Z}_2^\ell$, odd modulus q, a bound $\gamma \in \mathbb{R}^+$, a distribution χ over $\mathbb{Z}$, Gaussian parameters s and σ. Define $\Sigma : \mathbb{Z}^{k \times \ell} \to \mathbb{R}^{k \times k}$ as $S \mapsto \sigma^2 I_k - s^2 S S^\top$. Let $J := [I_m \mid 0^{m \times (k-m)}]^T \in \mathbb{Z}^{k \times m}$. Let $\mathcal{W} := \mathbb{Z}_{2q}^m$ and $\mathcal{Z} := \{z \in \mathbb{Z}^k \mid \|z\| \leq \gamma\}$.

Key generation: $\mathsf{Gen}_{\mathsf{LID}}$ computes vk and sk as follows: $A_1 \leftarrow \mathbb{Z}_q^{m \times (k-m-\ell)}$; $(S_1, S_2) \leftarrow \chi^{(k-m-\ell) \times \ell} \times \chi^{m \times \ell}$; $B := A_1 S_1 + S_2 \bmod q$; $A := [qJ - 2B \mid 2A_1 \mid 2I_m] \in \mathbb{Z}_{2q}^{m \times k}$; $S := [I_\ell \mid S_1^\top \mid S_2^\top]^\top \in \mathbb{Z}^{k \times \ell}$; $vk := A$; $sk := S$; outputs vk and sk.

Lossy key generation: $\mathsf{LossyGen}_{\mathsf{LID}}$ compute vk as follows: $A_1 \leftarrow \mathbb{Z}_q^{m\times(k-m-\ell)}$; $B \leftarrow \mathbb{Z}_q^{m\times\ell}$; $A := [qJ - 2B \mid 2A_1 \mid 2I_m] \in \mathbb{Z}_{2q}^{m\times k}$; outputs $vk := A$.
Challenge space: The challenge space is $C \subseteq \mathbb{Z}_2^{\ell}$. See [DPS23] for the parameter choices.
Prover: The prover's algorithms are defined as follows:

- $\mathsf{P}_1(sk)$ samples $y \leftarrow D_{\mathbb{Z}^k,\Sigma(S),0}$ and computes $w := Ay \bmod 2q$. It also samples seed for a seed of P_2. It outputs w and $\mathsf{st} := (y, \mathsf{seed})$.
- $\mathsf{P}_2(sk, w, c, \mathsf{st})$ samples $k \leftarrow D_{\mathbb{Z}^\ell,s^2 I_\ell,-c/2}$ (with seed seed) and computes $z := y + 2Sk + Sc$. It outputs z.

Reconstruction: $\mathsf{Rec}(vk, c, z)$ returns $w = Az - qJc \bmod 2q$. (Note: Rec implicitly checks whether $z \in \mathcal{Z}$ or not.)
Verifier: $\mathsf{V}(vk, w, c, z)$ checks if $w = \mathsf{Rec}(vk, c, z)$.
Simulator: $\mathsf{Sim}(vk, c)$ samples $z \leftarrow D_{\mathbb{Z}^k,\sqrt{2}\sigma}$ and outputs $w = \mathsf{Rec}(vk, c, z)$.
The protocol achieves $(1-\mathsf{negl}(\kappa), 0)$-correctness [DPS23, Theorem 2], statistical HVZK and the high commitment min-entropy [DPS23, Theorem 3] with careful choice of parameters. It also ϵ_ℓ-lossy with appropriate parameter settings, and its key indistinguishability follows from the decisional LWE assumption [DPS23, Theorem 4]. The computational unique response follows from the SIS assumption and the LWE assumption as follows:

Lemma B.2. *Let $m \geq \ell > 0$ and $k > m + \ell$. Let a be a positive integer. Let q be the odd modulus and let $\gamma \in \mathbb{R}^+$ be bound. Let χ be a distribution over $\mathbb{Z}$. The protocol has the CUR property under the $\mathsf{SIS}_{m,k+a,q,2\gamma}$ assumption and the $\mathsf{LWE}_{k-m-\ell,m,\ell,\chi,q}$ assumption.*
Precisely speaking, for a quantum adversary $\mathcal{A}$ breaking the CUR property of the identification protocol, there exist a quantum adversary $\mathcal{A}_{\text{lwe}}$ against the $\mathsf{LWE}_{k-m-\ell,m,\ell,\chi,q}$ assumption and a quantum adversary $\mathcal{A}_{\text{sis}}$ against the $\mathsf{SIS}_{m,k+a,q,2\gamma}$ assumption such that

$$\begin{aligned}
&\mathsf{Adv}^{\text{cur}}_{\mathsf{LID},\mathcal{A}}(\kappa) \leq \mathsf{Adv}_{\mathsf{LWE},\mathcal{A}_{\text{lwe}}}(\kappa) + \mathsf{Adv}_{\mathsf{SIS},\mathcal{A}_{\text{sis}}}(\kappa) + 2mq^{-(a+1)},\\
&\mathsf{Time}^*(\mathcal{A}_{\text{lwe}}) = \mathsf{Time}(\mathcal{A}) + O(\mathsf{Time}(\mathsf{LID})),\\
&\mathsf{Mem}^*(\mathcal{A}_{\text{lwe}}) = \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{LID})),\\
&\mathsf{Time}^*(\mathcal{A}_{\text{sis}}) = \mathsf{Time}(\mathcal{A}) + O((k+a)^3 \log^3 q),\\
&\mathsf{Mem}^*(\mathcal{A}_{\text{sis}}) = \mathsf{Mem}(\mathcal{A}) + O(m(k+a)\log q).
\end{aligned}$$

Proof. Let us consider two games: The first one is G_0 in which the challenger generates $(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{LID}}(1^\kappa)$, runs the adversary on input vk and receives (w, c, z, z') from the adversary, and returns $[\![z \neq z' \wedge \mathsf{V}(vk, w, c, z) \wedge \mathsf{V}(vk, w, c, z')]\!]$. The second one G_1 is the same game G_0 except that $vk \leftarrow \mathsf{LossyGen}_{\mathsf{LID}}(1^\kappa)$.
By definition, we have

$$\Pr[\mathsf{G}_0 \Rightarrow \mathsf{true}] = \mathsf{Adv}^{\text{cur}}_{\mathsf{LID},\mathcal{A}}(\kappa).$$

It is easy to construct an adversary $\mathcal{A}_{\text{lwe}}$ such that

$$\begin{aligned}
&|\Pr[\mathsf{G}_0 \Rightarrow \mathsf{true}] - \Pr[\mathsf{G}_1 \Rightarrow \mathsf{true}]| \leq \mathsf{Adv}_{\mathsf{LWE},\mathcal{A}_{\text{lwe}}}(\kappa),\\
&\mathsf{Time}^*(\mathcal{A}_{\text{lwe}}) = \mathsf{Time}(\mathcal{A}) + O(\mathsf{Time}(\mathsf{LID})),\\
&\mathsf{Mem}^*(\mathcal{A}_{\text{lwe}}) = \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{LID})).
\end{aligned}$$

By using an adversary $\mathcal{A}$ in G_1, we construct an adversary $\mathcal{A}_{\text{sis}}$ as follows: Given $\tilde{A} = [\tilde{A}_1 \mid \tilde{A}_2] \leftarrow \mathbb{Z}_q^{m\times(k+a)}$ with $\tilde{A}_1 \in \mathbb{Z}_q^{m\times(k-m)}$ and $\tilde{A}_2 \in \mathbb{Z}_q^{m\times(m+a)}$, $\mathcal{A}_{\text{sis}}$ tries to find a set of m linearly independent vectors $\tilde{a}_{i_1}, \dots, \tilde{a}_{i_m}$ from $\tilde{A}_2$. This set exists with probability at least $1 - 2mq^{-(a+1)}$ (see Lemma B.3 below). Let $\hat{A} = [\hat{A}_1 \mid \hat{A}_2] := [\tilde{A}_1 \mid \tilde{a}_{i_1} \dots \tilde{a}_{i_m}] = \tilde{A} \cdot P$, where $\hat{A}_1 \in \mathbb{Z}_q^{(k-m)\times m}$, $\hat{A}_2 = [\tilde{a}_{i_1} \dots \tilde{a}_{i_m}] \in \mathbb{Z}_q^{m\times m}$, and P is a corresponding matrix in $\{0,1\}^{(k+a)\times k}$. Notice that the Hamming weight of the columns of P is 1, and the Hamming weight of the rows of P is at most 1. It then computes $A := 2((2\hat{A}_2)^{-1} \cdot \hat{A} \bmod q) + [qJ \mid O] \bmod 2q$ and feeds it to $\mathcal{A}$. The distribution of this lossy verification key is perfect. $\mathcal{A}$ outputs (w, c, z, z'). If $z \neq z'$ and $\mathsf{V}(vk, w, c, z) = \mathsf{V}(vk, w, c, z') = \mathsf{true}$, then $\mathcal{A}_{\text{sis}}$ output $P(z - z')$ as the solution of the SIS problem.
If we have (w, c, z, z') such that $z \neq z'$ and $\mathsf{V}(vk, w, c, z) = \mathsf{V}(vk, w, c, z') = \mathsf{true}$, then we have the relations

$$\|z\| \leq \gamma \wedge \|z'\| \leq \gamma \wedge w \equiv Az - qJc \equiv Az' - qJc \pmod{2q}.$$

The bounds on the norms imply $\|z - z'\| \leq 2\gamma$ and the condition $z \neq z'$ implies $z - z' \neq 0$. The last equation with the fact that q is odd implies $A(z - z') \equiv 0 \pmod{q}$ (instead of $2q$). Therefore, we have

$$A(z - z') \equiv 2(2\hat{A}_2)^{-1}\hat{A} \cdot (z - z') \equiv \hat{A}_2^{-1} \cdot \tilde{A} \cdot P(z - z') \equiv 0 \pmod{q}$$

Multiplying $\hat{A}_2$ to the both sides, we have

$$\tilde{A} \cdot P(z - z') \equiv 0 \pmod{q}.$$

Due to the property of P, we have $P(z - z') \neq 0$ and $\|P(z - z')\| = \|z - z'\| \leq 2\gamma$. Thus, $P(z - z')$ is the solution of an instance $\tilde{A}$ of the SIS problem.
Thus, we have

$$\begin{aligned}
\Pr[\mathsf{G}_1 \Rightarrow \mathsf{true}] &\leq \mathsf{Adv}_{\mathsf{SIS},\mathcal{A}_{\mathrm{sis}}}(\kappa) + O(q^{-m}), \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{sis}}) &= \mathsf{Time}(\mathcal{A}) + O(k^3 \log^3 q), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{sis}}) &= \mathsf{Mem}(\mathcal{A}) + O(mk \log q).
\end{aligned}$$

Lemma B.3. *Let m, a be a positive inteters. We have*

$$\Pr_{D \leftarrow \mathbb{Z}_q^{m \times (m+a)}}[\mathrm{rank}(D) < m] \geq 2mq^{-(a+1)}.$$

While the above bound is well-known, we include the proof for completeness.

Proof. By the formula in [Bel93, MMO04, FG15], we have

$$\Pr_{D \leftarrow \mathbb{Z}_q^{m \times (m+a)}}[\mathrm{rank}(D) = m - r] = \frac{1}{q^{r(a+r)}} \cdot \frac{\prod_{i=1}^{m+a}(1 - q^{-i}) \prod_{i=r+1}^{m}(1 - q^{-i})}{\prod_{i=1}^{m-r}(1 - q^{-i}) \prod_{i=1}^{a+r}(1 - q^{-i})}.$$

Since we consider the case $r = 0$, the probability is

$$\begin{aligned}
\Pr_{D \leftarrow \mathbb{Z}_q^{m \times (m+a)}}[\mathrm{rank}(D) = m] &= \frac{\prod_{i=1}^{m+a}(1 - q^{-i}) \prod_{i=1}^{m}(1 - q^{-i})}{\prod_{i=1}^{m}(1 - q^{-i}) \prod_{i=1}^{a}(1 - q^{-i})} = \frac{\prod_{i=1}^{m+a}(1 - q^{-i})}{\prod_{i=1}^{a}(1 - q^{-i})} \\
&= \prod_{i=a+1}^{m+a}(1 - q^{-i}) \geq (1 - q^{-(a+1)})^m \geq 1 - 2mq^{-(a+1)}.
\end{aligned}$$

Thus, the lemma follows. □

C Relation Between Blind Unforgeability and Plus-One Unforgeability

Alagic et al. showed that there exists a PO-secure but BU-insecure MAC scheme by assuming a random function or qPRF [AMRS20, Section 8.1]. In the original version of [AMRS20], Alagic et al. insisted that BU security implies PO security (for MAC), but this claim was retracted in Apr. 2023. They weakened their claim as that their BU security implies quadratic PO security, where an adversary is required to output ck^2 forgeries with probability 1 by making k quantum queries for a fixed constant c [AMRS18, Section 5.2.3].
Here, we give a simple example of BU-secure but PO-insecure signature assuming the existence of BU-secure signature. Our example exploits the fact that PO security is a quantum version of *strong existential unforgeability*, but BU security does not.

Lemma C.1 (BU $\not\Rightarrow$ PO). *Suppose that there exists a* BU*-secure MAC/signature scheme. We then have a* BU*-secure but* PO*-insecure MAC/signature scheme.*

In the proof, we only consider the signature schemes. The lemma for MAC is obtained similarly.

Proof. Suppose that we have a BU-secure $\mathsf{SIG} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ whose signature space is $\mathcal{S} \subseteq \{0,1\}^\lambda$ for some $\lambda = \lambda(\kappa)$. We construct a new BU-secure signature scheme $\mathsf{SIG}' = (\mathsf{Gen}, \mathsf{Sign}', \mathsf{Vrfy}')$ as follows:

- $\mathsf{Sign}'(sk, m)$: Generate $\sigma \leftarrow \mathsf{Sign}(sk, m)$, and output $\sigma' := (\sigma, 0)$.
- $\mathsf{Vrfy}'(vk, m, \sigma')$: Parse $\sigma' = (\sigma, b)$ with $b \in \{0,1\}$ and output $\mathsf{dec} := \mathsf{Vrfy}(vk, m, \sigma)$.

Note that the new signature space is $\mathcal{S}' \subseteq \{0,1\}^{\lambda+1}$.

BU *security:* This new signature scheme is still BU-secure because we can construct an adversary $\mathcal{A}$ breaking the BU security of SIG if there exists an adversary $\mathcal{A}'$ breaking the BU security of SIG′. $\mathcal{A}$ is defined as follows: On input vk, it runs $\mathcal{A}'$ on input vk. For a hash query to the random oracle, it passes the query to its random oracle and returns the result. It also implements the blinded signing oracle $|B_\epsilon \textsc{Sign}'\rangle$ for $\mathcal{A}'$ as follows: For a signing query $|m\rangle |y\rangle |y'_0\rangle |y'_1\rangle$ to $B_\epsilon \textsc{Sign}'$, where $y \in \{0,1\}^\lambda$, $y'_0, y'_1 \in \{0,1\}$,

1. query $|m\rangle |y\rangle |y'_1\rangle$ to its signing oracle $|B_\epsilon \textsc{Sign}\rangle$
2. receive $|m\rangle |y \oplus \sigma\rangle |y'_1 \oplus b_\sigma\rangle$, where $\sigma \| b_\sigma$ is $\sigma \| 0$ or $\perp = 0^\lambda \| 1$
3. return $|m\rangle |y \oplus \sigma\rangle |y'_0\rangle |y'_1 \oplus b_\sigma\rangle$.

This perfectly simulates $B_\epsilon \textsc{Sign}'$. If $\mathcal{A}'$ outputs m and (σ, b) with $b \in \{0,1\}$, $\mathcal{A}$ outputs m and σ as a forgery.

PO *insecurity:* On the other hand, this new signature scheme is PO-insecure: If we obtain a signature $\sigma' = (\sigma, 0)$ on a message m by querying to $|\textsc{Sign}\rangle$, we can output *two valid distinct pairs* of message and siganture, $(m, (\sigma, 0))$ and $(m, (\sigma, 1))$. □

Remark C.1. On their security definition of MAC, Boneh and Zhandry [BZ13a] wrote that "After issuing q quantum chosen message queries the adversary wins the game if it can generate $q+1$ valid classical message-tag pairs." just before Definition 1 (EUF-qCMA). While there is an ambiguity of distinctness, we treat it as $q + 1$ distinct pairs, since they reviewed sEUF-CMA security of MAC as the classical security definition.

Refuting that BU implies quadratic PO: The above example can be used to show that there exists BU-seucre but quadratic PO-insecure MAC, while Alagic et al. showed that BU implies quadratic PO [AMRS18].
Let $a = \omega(\lg(\kappa))$. Suppose we have a BU-secure MAC scheme $\mathsf{MAC} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$. We then construct a new BU-secure MAC scheme $\mathsf{MAC}' = (\mathsf{Gen}, \mathsf{Sign}', \mathsf{Vrfy}')$ as follows:

- $\mathsf{Sign}'(sk, m)$: Generate $\sigma \leftarrow \mathsf{Sign}(sk, m)$, and output $\sigma' := (\sigma, 0)$.
- $\mathsf{Vrfy}'(sk, m, \sigma')$: Parse $\sigma' = (\sigma, b)$ with $b \in \{0, 1\}^a$ and output $\mathsf{dec} := \mathsf{Vrfy}(sk, m, \sigma)$.

This new signature scheme is still BU-secure because we can construct an adversary breaking the BU security of MAC if there exists an adversary breaking the BU security of MAC'. On the other hand, given k pairs of distinct messages and corresponding tags, it is easy to construct ck^2 ($\leq k2^a$) distinct valid pairs of messages and tags when $ck \leq 2^a = 2^{\omega(\lg(\kappa))}$.

Summary: As we exemplified, BU security does not imply PO security. What we should ask is the relation between sBU security and PO security and the relation between BU security and weakened PO security, where the adversary is required to output $q + 1$ distinct messages and their corresponding signatures/tags.

D Memory-Tight Proofs for PSF-(P)FDH

D.1 Preimage Sampleable Functions

We review the syntax of preimage sampleable functions.

Definition D.1 (Preimage sampleable function [GPV08]). *A family of preimage sampleable functions consists* PSF *of the following quadruple of PPT algorithms* $(\mathsf{Gen}_{\mathsf{PSF}}, \mathsf{F}, \mathsf{Inv}, \mathsf{Sample})$*:*

- $\mathsf{Gen}_{\mathsf{PSF}}(1^\kappa) \to (vk, sk)$*: a key-generation algorithm that on input* 1^κ *outputs a pair of keys* (vk, sk).
- $\mathsf{F}(vk, x) \to y$*: a* deterministic *evaluation algorithm that takes as input* vk *and* $x \in \mathcal{X}$ *and outputs* $y \in \mathcal{Y}$.
- $\mathsf{Sample}(vk) \to x$*: a sampling algorithm that takes as input* vk *and outputs* $x \in \mathcal{X}$.
- $\mathsf{Inv}(sk, y) \to x$*: a preimage-sampling algorithm that takes as input* sk *and* $y \in \mathcal{Y}$ *and outputs* $x \in \mathcal{X}$.

We then define properties of PSF.

Definition D.2 (Simulatability [CCLM22]). *We say that* PSF *is* ϵ-simulatable *if the following two distributions are* ϵ*-close:*

$$D_1 : y \leftarrow \mathcal{Y}; x \leftarrow \mathsf{Inv}(sk, y); \mathit{return}\ (x, y)$$
$$D_2 : x \leftarrow \mathsf{Sample}(vk); y := \mathsf{F}(vk, x); \mathit{return}\ (x, y).$$

Definition D.3 (Preimage min-entropy). *We say that* PSF *has* α-preimage min-entropy *if for each* $y \in \mathcal{Y}$*, the conditional min-entropy of* $x \leftarrow \mathsf{Sample}(vk)$ *given* $\mathsf{F}(vk, x) = y$ *is at least* α.

Definition D.4 (Collision resistance). *We say that* PSF *is* collision-resistant *if for any QPT adversary* $\mathcal{A}$*, the following advantage is negligible in* κ*:*

$$\mathsf{Adv}^{\mathrm{cr}}_{\mathsf{PSF},\mathcal{A}}(\kappa) := \Pr\left[\begin{array}{c}(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{PSF}}(1^\kappa); (x, x') \leftarrow \mathcal{A}(vk) : \\ x \neq x' \wedge \mathsf{F}(vk, x) = \mathsf{F}(vk, x')\end{array}\right].$$

D.2 Signature based on PSF

We review a signature scheme constructed from preimage-sampleable functions (PSF) [BR96, GPV08]. Let $\mathsf{PSF} = (\mathsf{Gen}_{\mathsf{PSF}}, \mathsf{F}, \mathsf{Inv}, \mathsf{Sample})$ be a family of preimage sampleable functions. The signature scheme obtained by applying the Full-Domain Hash FDH is depicted in Figure 12. If Inv is derandomized by PRF, then we call this conversion as DFDH and denote $\mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$. If we use RF instead of PRF, then we denote it as $\mathsf{DFDH}^+[\mathsf{PSF}, \mathsf{H}, \mathsf{RF}]$. If we apply RDS in subsection 3.1 to the obtained scheme, then we call the conversion as PFDH and denote $\mathsf{PFDH}[\mathsf{PSF}, \mathsf{H}, \lambda]$.

```
$\underline{\mathsf{Gen}(1^\kappa)}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{PSF}}(1^\kappa)$
return $(vk, sk)$
```

```
$\underline{\mathsf{Sign}(sk, m)}$
$h := \mathsf{H}(m)$
$\sigma \leftarrow \mathsf{Inv}(sk, h)$
return $\sigma$
```

```
$\underline{\mathsf{Vrfy}(vk, m, \sigma)}$
$h := \mathsf{H}(m)$
$h' := \mathsf{F}(vk, \sigma)$
return $[\![h = h']\!]$
```

```
$\underline{\mathsf{Gen}(1^\kappa)}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{PSF}}(1^\kappa)$
$K \leftarrow \{0,1\}^\kappa$
return $(vk, (sk, K))$
```

```
$\underline{\mathsf{Sign}((sk, K), m)}$
$h := \mathsf{H}(m)$
$r := \mathsf{PRF}(K, m)$
$\sigma := \mathsf{Inv}(sk, h; r)$
return $\sigma$
```

```
$\underline{\mathsf{Vrfy}(vk, m, \sigma)}$
$h := \mathsf{H}(m)$
$h' := \mathsf{F}(vk, \sigma)$
return $[\![h = h']\!]$
```

Fig. 12. $\mathsf{FDH}[\mathsf{PSF}, \mathsf{H}]$ (upper) and $\mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$ (lower).

D.3 Multi-Challenge Security for PSF-(P/D)FDH

While we can use both approaches of Diemert et al. [DGJL21] and Ghoshal et al. [GGJT22], we here use the approach of Diemert et al. [DGJL21]: We show the MsEUF-CMA1 security of $\mathsf{FDH}[\mathsf{PSF}, \mathsf{H}]$ by slightly modifying the sEUF-CMA proof of $\mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$ in Boneh et al. [BDF+11] or the BU proof of $\mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$ in Chatterjee et al. [CCLM22] and apply Lemma 3.1 to show the MsEUF-CMA security of $\mathsf{PFDH}[\mathsf{PSF}, \mathsf{H}] = \mathsf{RDS}[\mathsf{FDH}[\mathsf{PSF}, \mathsf{H}], \lambda]$ via memory-tight reductions.

Theorem D.1 (MsEUF-CMA1 **security of** $\mathsf{FDH}[\mathsf{PSF}, \mathsf{H}]$). *Let* $\mathsf{H} \colon \mathcal{M} \to \mathcal{Y}$ *be a random oracle. Let* PSF *be a family of preimage-sampleable functions that is* ϵ*-simulatable and has* α*-preimage min-entropy. Let* $\mathsf{DS} := \mathsf{FDH}[\mathsf{PSF}, \mathsf{H}]$*. Then, for a quantum adversary* $\mathcal{A}$ *breaking the* MsEUF-CMA1 *security of* DS *taht issues at most* q_H *quantum queried to* H*,* q_S *classical queries to the signing oracle, and* q_F *classical queries to the forgery-checking oracle, there exists a quantum* $\mathcal{F}$*-oracle adversary* $\mathcal{A}_{\mathrm{cr}}$ *such that*

$$\begin{aligned}
&\mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS},\mathcal{A}}(\kappa) \leq \mathsf{Adv}^{\text{cr}}_{\mathsf{PSF},\mathcal{A}_{\text{cr}}}(\kappa) + \sqrt{(6(q_S + q_H + q_F))^3 \epsilon} + q_F \cdot 2^{-\alpha},\\
&\mathsf{Time}^*(\mathcal{A}_{\text{cr}}) = \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})),\\
&\mathsf{Mem}^*(\mathcal{A}_{\text{cr}}) = \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF})),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$*.*

Applying Lemma 3.1, we obtain the following corollary.

Corollary D.1 (MsEUF-CMA **security of** $\mathsf{PFDH}[\mathsf{PSF}, \mathsf{H}, \lambda]$). *Let* $\mathsf{H} \colon \mathcal{M} \times \{0,1\}^\lambda \to \mathcal{Y}$ *be a random oracle. Let* PSF *be a family of preimage-sampleable functions that is* ϵ*-simulatable and has* α*-preimage min-entropy. Let* $\mathsf{DS} := \mathsf{PFDH}[\mathsf{PSF}, \mathsf{H}, \lambda]$*. Then, for a quantum adversary* $\mathcal{A}$ *breaking the* MsEUF-CMA *security of* DS *that issues at most* q_H *quantum queried to* H*,* q_S *classical queries to the signing oracle, and* q_F *classical queries to the forgery-checking oracle, there exists a quantum* $\mathcal{F}$*-oracle adversary* $\mathcal{A}_{\mathrm{cr}}$ *such that*

$$\begin{aligned}
&\mathsf{Adv}^{\text{mseuf-cma}}_{\mathsf{DS},\mathcal{A}}(\kappa) \leq \mathsf{Adv}^{\text{cr}}_{\mathsf{PSF},\mathcal{A}_{\text{cr}}}(\kappa) + \sqrt{(6(q_S + q_H + q_F))^3 \epsilon} + q_F \cdot 2^{-\alpha} + q_S^2 \cdot 2^{-\lambda},\\
&\mathsf{Time}^*(\mathcal{A}_{\text{cr}}) = \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})),\\
&\mathsf{Mem}^*(\mathcal{A}_{\text{cr}}) = \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF})),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$*.*

Game G_0*:* This is the original game of the MsEUF-CMA1 security. See G_0 in Figure 13. By definition, we have

$$\Pr[W_0] = \mathsf{Adv}^{\text{mseuf-cma1}}_{\mathsf{DS},\mathcal{A}}(\kappa).$$

Game G_1*:* Next, we derandomize the signing oracle using a random function RF_{I}. By this modification, we do not need to maintain the list in the signing oracle. We have

$$\mathsf{G}_0 = \mathsf{G}_1.$$

```
$\mathsf{G}_i$ for $i \in \{0,1,2,3\}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{PSF}}(1^\kappa)$
$K \leftarrow \{0,1\}^\kappa$                                   $/\mathsf{G}_0$
$\mathsf{RF}_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{Y})$                 $/\mathsf{G}_0$-$\mathsf{G}_1$
$\mathsf{RF}_\mathsf{I} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Inv}})$                  $/\mathsf{G}_1$
$\mathsf{RF}_\mathsf{S} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$               $/\mathsf{G}_2$-
$Q := \emptyset$                                    $/\mathsf{G}_0$-$\mathsf{G}_2$
$\mathsf{win} := \mathsf{false}$
run $\mathcal{A}^{\textsc{Sign},\textsc{Forge},|\mathsf{H}\rangle}(vk)$
return $\mathsf{win}$

$\textsc{Sign}(m)$
if $\exists(m, \sigma) \in Q$ then                       $/\mathsf{G}_0$
    return $\sigma$                                  $/\mathsf{G}_0$
$\sigma \leftarrow \mathsf{Inv}(sk, \mathsf{H}(m))$                         $/\mathsf{G}_0$
$\sigma := \mathsf{Inv}(sk, \mathsf{H}(m); \mathsf{RF}_\mathsf{I}(m))$                 $/\mathsf{G}_1$
$\sigma := \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m))$                  $/\mathsf{G}_2$-
$Q := Q \cup \{(m, \sigma)\}$                            $/\mathsf{G}_0$-$\mathsf{G}_2$
return $\sigma$

$\mathsf{H} : |m\rangle|y\rangle \mapsto |m\rangle|y \oplus h\rangle$
return $h := \mathsf{RF}_\mathsf{H}(m)$                          $/\mathsf{G}_0$-$\mathsf{G}_1$
return $h := \mathsf{F}(vk, \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m)))$      $/\mathsf{G}_2$-

$\textsc{Forge}(m^*, \sigma^*)$
$h' := \mathsf{H}(m^*)$                                 $/\mathsf{G}_0$-$\mathsf{G}_2$
$\sigma' := \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*))$                   $/\mathsf{G}_3$
$h' := \mathsf{F}(vk, \sigma')$                               $/\mathsf{G}_3$
$h^* := \mathsf{F}(vk, \sigma^*)$
if $h^* = h'$ then                          $/\mathsf{Vrfy}$ passed
    if $(m^*, \sigma^*) \notin Q$ then                    $/\mathsf{G}_0$-$\mathsf{G}_2$
        $\mathsf{win} := \mathsf{true}$                           $/\mathsf{G}_0$-$\mathsf{G}_2$
    if $\sigma^* \neq \sigma'$ then                              $/\mathsf{G}_3$
        $\mathsf{win} := \mathsf{true}$                              $/\mathsf{G}_3$
```

Fig. 13. G_i for $i \in \{0,1,2,3\}$ for MSEUF-CMA1 security.

Game G_2*:* We next modify the signing oracle and the random oracle. In this game, the signing oracle given m returns $\mathsf{Sample}(vk, \mathsf{RF}_\mathsf{S}(m))$ and the random oracle given m returns $\mathsf{F}(vk, \mathsf{Sample}(vk, \mathsf{RF}_\mathsf{S}(m)))$. By applying Lemma 2.2 with ϵ-simulatability, we have

$$|\Pr[W_1] - \Pr[W_2]| \leq \sqrt{(6(q_S + q_H + q_F))^3 \epsilon}.$$

Game G_3*:* We finally modify how to update the flag win. In G_3, the flag is set true if the submitted forgery σ^* differs from the expected one σ'.

Lemma D.1. *Suppose that* PSF *has α-preimage min-entropy. We have*

$$|\Pr[W_2] - \Pr[W_3]| \leq q_F \cdot 2^{-\alpha}.$$

Proof. Suppose that an adversary $\mathcal{A}$ submits a valid pair (m^*, σ^*). let $\sigma' := \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*))$. Let us consider two cases:

1. If m^* is already queried to SIGN, then (m^*, σ') should be contained in Q. Thus the condition $(m^*, \sigma^*) \notin Q$ is equivalent to $\sigma^* \neq \sigma'$ and the flags win are the same in the both games.
2. If m^* is not queried to SIGN, then the two games differ if the adversary submits (m^*, σ') in G_2. Since the min-entropy of $\sigma' = \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*))$ given $h^* = \mathsf{F}(vk, \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*)))$ is at least α, this event happens with probability at most $q_F \cdot 2^{-\alpha}$.

Thus, we obtain the bound. □

Now, making a memory-tight reduction for collision-resistance of PSF is easy.

Lemma D.2. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{cr}}$ such that*

$$\begin{aligned}
&\Pr[W_3] \leq \mathsf{Adv}^{\mathrm{cr}}_{\mathsf{PSF},\mathcal{A}_{\mathrm{cr}}}(\kappa),\\
&\mathsf{Time}^*(\mathcal{A}_{\mathrm{cr}}) = \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})),\\
&\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cr}}) = \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF})),
\end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$.

Since the reduction is straightforward, we omit it.

```
$\mathsf{G}_i$ for $i \in \{0,1,2,3,4\}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{PSF}}(1^\kappa)$
$K \leftarrow \{0,1\}^\kappa$                                  /$\mathsf{G}_0$
$\mathsf{RF}_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{Y})$                  /$\mathsf{G}_0$-$\mathsf{G}_1$
$\mathsf{RF}_\mathrm{I} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Inv}})$                /$\mathsf{G}_1$
$\mathsf{RF}_\mathsf{S} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$             /$\mathsf{G}_2$-
$\mathcal{Q} := \emptyset$                                    /$\mathsf{G}_0$-$\mathsf{G}_3$
$\mathsf{win} := \mathsf{false}$                              /$\mathsf{G}_3$-
run $\mathcal{A}^{|\textsc{Sign}\rangle, \textsc{Forge}, |\mathsf{H}\rangle}(vk)$
return $[\![\#\mathcal{Q} > q_S]\!]$                          /$\mathsf{G}_2$
return $[\![\#\mathcal{Q} > q_S]\!] \land \mathsf{win}$                    /$\mathsf{G}_3$
return $\mathsf{win}$                                     /$\mathsf{G}_4$

$\textsc{Sign}: |m\rangle |y\rangle \mapsto |m\rangle |y \oplus \sigma\rangle$
$\sigma := \mathsf{Inv}(sk, \mathsf{H}(m); \mathsf{PRF}(K, m))$         /$\mathsf{G}_0$
$\sigma := \mathsf{Inv}(sk, \mathsf{H}(m); \mathsf{RF}_\mathrm{I}(m))$           /$\mathsf{G}_1$
$\sigma := \mathsf{Sample}(vk; \mathsf{RF}_{\mathsf{Sim}}(m))$              /$\mathsf{G}_2$
return $\sigma$
```

```
$\mathsf{H}: |m\rangle |y\rangle \mapsto |m\rangle |y \oplus h\rangle$
return $h := \mathsf{RF}_\mathsf{H}(m)$                              /$\mathsf{G}_0$-$\mathsf{G}_1$
return $h := \mathsf{F}(vk, \mathsf{Sample}(vk; \mathsf{RF}_{\mathsf{Sim}}(m)))$   /$\mathsf{G}_2$-

$\textsc{Forge}(m^*, \sigma^*)$
$h' := \mathsf{H}(m^*)$                                       /$\mathsf{G}_0$-$\mathsf{G}_2$
$h^* := \mathsf{F}(vk, \sigma^*)$
if $h^* = h'$ then                                 /Vrfy passed
    if $(m^*, \sigma^*) \notin \mathcal{Q}$ then                      /$\mathsf{G}_0$-$\mathsf{G}_3$
        $\mathcal{Q} := \mathcal{Q} \cup \{(m^*, \sigma^*)\}$                     /$\mathsf{G}_0$-$\mathsf{G}_3$
    if $\sigma^* \neq \mathsf{Sample}(vk; \mathsf{RF}_{\mathsf{Sim}}(m^*))$ then     /$\mathsf{G}_3$-$\mathsf{G}_4$
        $\mathsf{win} := \mathsf{true}$                             /$\mathsf{G}_3$-$\mathsf{G}_4$
```

Fig. 14. G_i for $i \in \{0,1,2,3,4\}$ for PO security.

D.4 Plus-One Security for PSF-DFDH

The following theorem is obtained by modifying the proof of Boneh and Zhandry [BZ13b, Theorem 3.19].

Theorem D.2 (**PO security of** $\mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$**).** *Let* $\mathsf{H}: \mathcal{M} \to \mathcal{Y}$ *be a random oracle. Let* PSF *be a family of preimage-sampleable functions that is* ϵ*-simulatable and has* α*-preimage min-entropy. Let* $\mathsf{DS} := \mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$*. Then, for a quantum adversary* $\mathcal{A}$ *breaking the* PO *security of* DS *that issues at most* q_H *quantum queried to* H*,* q_S *classical queries to the signing oracle, and* q_F *classical queries to the forgery-checking oracle, there exist a quantum* $\mathcal{F}_{\mathrm{prf}}$*-oracle adversary* $\mathcal{A}_{\mathrm{prf}}$ *and a quantum* $\mathcal{F}_{\mathrm{cr}}$*-oracle adversary* $\mathcal{A}_{\mathrm{cr}}$ *and such that*

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(\kappa) &\leq \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PSF},\mathcal{A}_{\mathrm{prf}}}(\kappa) + \mathsf{Adv}^{\mathrm{cr}}_{\mathsf{PSF},\mathcal{A}_{\mathrm{cr}}}(\kappa) \\
&\quad + \sqrt{(6(q_S + q_H + q_F))^3 \epsilon} + (q_S + 1)/\lfloor 2^\alpha \rfloor, \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{prf}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{prf}}) &= \mathsf{Mem}(\mathcal{A}) + O(q_F \mathsf{Mem}(\mathsf{PSF})), \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF})),
\end{aligned}$$

where $\mathcal{F}_{\mathrm{prf}} = \mathsf{Func}(\mathcal{M}, \mathcal{Y})$ *and* $\mathcal{F}_{\mathrm{cr}} = \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$*.*

The proof becomes memory-tight if we derandomize with the random function RF.

Corollary D.2 (**PO security of** $\mathsf{DFDH}^+[\mathsf{PSF}, \mathsf{H}, \mathsf{RF}]$**).** *Let* $\mathsf{H}: \mathcal{M} \to \mathcal{Y}$ *be a random oracle. Let* PSF *be a family of preimage-sampleable functions that is* ϵ*-simulatable and has* α*-preimage min-entropy. Let* $\mathsf{DS} := \mathsf{DFDH}^+[\mathsf{PSF}, \mathsf{H}, \mathsf{RF}]$*. Then, for a quantum adversary* $\mathcal{A}$ *breaking the* PO *security of* DS *that issues at most* q_H *quantum queried to* H*,* q_S *classical queries to the signing oracle, and* q_F *classical queries to the forgery-checking oracle, there exists a quantum* $\mathcal{F}_{\mathrm{cr}}$*-oracle adversary* $\mathcal{A}_{\mathrm{cr}}$ *and such that*

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(\kappa) &\leq \mathsf{Adv}^{\mathrm{cr}}_{\mathsf{PSF},\mathcal{A}_{\mathrm{cr}}}(\kappa) + \sqrt{(6(q_S + q_H + q_F))^3 \epsilon} + (q_S + 1)/\lfloor 2^\alpha \rfloor, \\
\mathsf{Time}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})), \\
\mathsf{Mem}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF})),
\end{aligned}$$

where $\mathcal{F}_{\mathrm{cr}} = \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$*.*

Game G_0*:* This is the original game of the PO security. See G_0 in Figure 14. By definition, we have

$$\Pr[W_0] = \mathsf{Adv}^{\mathrm{po}}_{\mathsf{DS},\mathcal{A}}(\kappa).$$

Game G_1*:* We next replace PRF with $\mathsf{RF}_{\mathcal{I}}$ in G_1. The straightforward reduction shows the following lemma, which is *memory-loose* since we need to maintain $\mathcal{Q}$.

Lemma D.3. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{prf}}$ such that*

$$\begin{aligned} |\Pr[W_0] - \Pr[W_1]| &\leq \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}_{\mathrm{prf}}}(\kappa), \\ \mathsf{Time}^*(\mathcal{A}_{\mathrm{prf}}) &= O(\mathsf{Time}(\mathcal{A})) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})), \\ \mathsf{Mem}^*(\mathcal{A}_{\mathrm{prf}}) &= O(\mathsf{Mem}(\mathcal{A})) + q_F \cdot O(\mathsf{Mem}(\mathsf{PSF})), \end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M}, \mathcal{Y})$.

Game G_2*:* We next modify the signing oracle and the random oracle. In this game, the signing oracle given m returns $\sigma = \mathsf{Sample}(vk, \mathsf{RF}_\mathsf{S}(m))$ and the random oracle given m returns $\mathsf{F}(vk, \mathsf{Sample}(vk, \mathsf{RF}_\mathsf{S}(m)))$. By applying Lemma 2.2 with ϵ-simulatability, we have

$$|\Pr[W_1] - \Pr[W_2]| \leq \sqrt{(6(q_S + q_H + q_F))^3 \epsilon}.$$

Game G_3*:* We next modify the winning condition as follows: We introduce a flag win which is set true if (m^*, σ^*) is valid and $\sigma^* \neq \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*))$ in the oracle FORGE. The challenger outputs $[\![\#\mathcal{Q} > \mathsf{cnt}_s]\!] \wedge \mathsf{win}$ instead of $[\![\#\mathcal{Q} > \mathsf{cnt}_s]\!]$. We have the following lemma as Lemma 6.2 by following the argument in the proof of [BZ13b, Theorem 3.19].

Lemma D.4. *Suppose that* PSF *has α-preimage min-entropy. We have*

$$|\Pr[W_2] - \Pr[W_3]| \leq (q_S + 1)/\lfloor 2^\alpha \rfloor.$$

Proof. The two games differ if the adversary submits at least $(q_S + 1)$ distinct pairs of message/signature $\{(m_i^*, \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m_i^*)))\}_i$. Since PSF has α-preimage min-entropy, even if the adversary knows $h^* = \mathsf{H}(m^*) = \mathsf{F}(vk, \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*)))$, the min-entropy of $\mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*))$ is at least α. Thus, applying Lemma 2.3, this event happens with probability at most $(q_S + 1)/\lfloor 2^\alpha \rfloor$. □

Game G_4*:* The challenger outputs the flag win in this game. We can remove the list $\mathcal{Q}$. Since we relax the condition and this relaxation cannot be detected by the adversary, we have

$$\Pr[W_3] \leq \Pr[W_4].$$

Constructing an adversary finding a collision for $\mathsf{F}(vk, \cdot)$ is easy.

Lemma D.5. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{cr}}$ such that*

$$\begin{aligned} \Pr[W_4] &\leq \mathsf{Adv}^{\mathrm{cr}}_{\mathsf{PSF},\mathcal{A}_{\mathrm{cr}}}(\kappa), \\ \mathsf{Time}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF})), \\ \mathsf{Mem}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF})), \end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$.

Since the reduction is straightforward, we omit it.

D.5 Strong Blind Unforgeability for PSF-DFDH

While Chatterjee et al. [CCLM22] showed the BU security of PSF-DFDH via memory-loose reductions, we here show stronger security (sBU security) with memory-tight reduction. The proof is obtained by slightly modifying their proof for the BU security.

Theorem D.3 (sBU security of $\mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$). *Let* $\mathsf{H} : \mathcal{M} \to \mathcal{Y}$ *be a random oracle. Let* PSF *be a family of preimage-sampleable functions that is ϵ-simulatable and has α-preimage min-entropy. Let* $\mathsf{DS} := \mathsf{DFDH}[\mathsf{PSF}, \mathsf{H}, \mathsf{PRF}]$. *Then, for a quantum adversary $\mathcal{A}$ breaking the* sBU *security of* DS *that issues at most q_H quantum queried to* H, *q_S classical queries to the signing oracle, and q_F classical queries to the forgery-checking oracle, there exist a quantum $\mathcal{F}_{\mathrm{prf}}$-oracle adversary $\mathcal{A}_{\mathrm{prf}}$ and a quantum $\mathcal{F}_{\mathrm{cr}}$-oracle adversary $\mathcal{A}_{\mathrm{cr}}$ such that*

$$\begin{aligned} \mathsf{Adv}^{\mathrm{sbu}}_{\mathsf{DS},\mathcal{A}}(\kappa) &\leq \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}_{\mathrm{prf}}}(\kappa) + \mathsf{Adv}^{\mathrm{cr}}_{\mathsf{PSF},\mathcal{A}_{\mathrm{cr}}}(\kappa) \\ &\quad + \sqrt{(6(q_S + q_H + q_F))^3 \epsilon} + q_F \cdot 2^{-\alpha}, \\ \mathsf{Time}^*(\mathcal{A}_{\mathrm{prf}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF}) + \mathsf{Time}(B_\epsilon)), \\ \mathsf{Mem}^*(\mathcal{A}_{\mathrm{prf}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF})) + \mathsf{Mem}(B_\epsilon)), \\ \mathsf{Time}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF}) + \mathsf{Time}(B_\epsilon)), \\ \mathsf{Mem}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF}) + \mathsf{Mem}(B_\epsilon)), \end{aligned}$$

where $\mathcal{F}_{\mathrm{prf}} = \mathsf{Func}(\mathcal{M} \times \mathcal{X}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M}, \mathcal{Y})$ *where* $\mathcal{F}_{\mathrm{cr}} = \mathsf{Func}(\mathcal{M} \times \mathcal{X}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$.

```
$\mathsf{G}_i$ for $i \in \{0,1,2,3\}$
$(vk, sk) \leftarrow \mathsf{Gen}_{\mathsf{PSF}}(1^\kappa)$
$K \leftarrow \{0,1\}^\kappa$                                   $/\mathsf{G}_0$
$\mathsf{RF}_B \leftarrow \mathsf{Func}(\mathcal{M} \times \mathcal{X}, \mathcal{P})$
$\mathsf{RF}_\mathsf{H} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{Y})$                        $/\mathsf{G}_0\text{-}\mathsf{G}_1$
$\mathsf{RF}_\mathrm{I} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Inv}})$                        $/\mathsf{G}_1$
$\mathsf{RF}_\mathsf{S} \leftarrow \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$                     $/\mathsf{G}_2\text{-}$
$\mathsf{win} := \mathsf{false}$
run $\mathcal{A}^{|\textsc{Sign}\rangle, \textsc{Forge}, |\mathsf{H}\rangle}(vk)$
return $\mathsf{win}$

$B_\epsilon\textsc{Sign}: |m\rangle|y\rangle \mapsto |m\rangle|y \oplus \sigma\rangle$
$\sigma := \mathsf{Inv}(sk, \mathsf{H}(m); \mathsf{PRF}(K, m))$          $/\mathsf{G}_0$
$\sigma := \mathsf{Inv}(sk, \mathsf{H}(m); \mathsf{RF}_\mathrm{I}(m))$            $/\mathsf{G}_1$
$\sigma := \mathsf{Sample}(vk; \mathsf{RF}_{\mathsf{Sim}}(m))$              $/\mathsf{G}_2$
if $(m, \sigma) \in B_\epsilon$ then
    return $\sigma := \bot$
else
    return $\sigma$
```

```
$\mathsf{H}: |m\rangle|y\rangle \mapsto |m\rangle|y \oplus h\rangle$
return $h := \mathsf{RF}_\mathsf{H}(m)$                                  $/\mathsf{G}_0\text{-}\mathsf{G}_1$
return $h := \mathsf{F}(vk, \mathsf{Sample}(vk; \mathsf{RF}_{\mathsf{Sim}}(m)))$   $/\mathsf{G}_2\text{-}$

$\textsc{Forge}(m^*, \sigma^*)$
$h' := \mathsf{H}(m^*)$                                       $/\mathsf{G}_0\text{-}\mathsf{G}_2$
$\sigma' := \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*))$                       $/\mathsf{G}_3$
$h' := \mathsf{F}(vk, \sigma')$                                    $/\mathsf{G}_3$
$h^* := \mathsf{F}(vk, \sigma^*)$
if $h^* = h'$ then                               /Vrfy passed
    if $(m^*, \sigma^*) \in B_\epsilon$ then                    $/\mathsf{G}_0\text{-}\mathsf{G}_2$
        $\mathsf{win} := \mathsf{true}$                              $/\mathsf{G}_0\text{-}\mathsf{G}_2$
    if $(m^*, \sigma^*) \in B_\epsilon \wedge \sigma^* \neq \sigma'$ then            $/\mathsf{G}_3$
        $\mathsf{win} := \mathsf{true}$                                  $/\mathsf{G}_3$
```

Fig. 15. G_i for $i \in \{0,1,2,3\}$ for sBU security.

Game G_0*:* This is the original game of the sBU security. See G_0 in Figure 15. By definition, we have

$$\Pr[W_0] = \mathsf{Adv}^{\mathrm{sbu}}_{\mathsf{DS},\mathcal{A}}(\kappa).$$

Game G_1*:* We next replace PRF with RF_I in G_1. The straightforward reduction shows the following lemma.

Lemma D.6. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{prf}}$ such that*

$$\begin{aligned} |\Pr[W_0] - \Pr[W_1]| &\leq \mathsf{Adv}^{\mathrm{pr}}_{\mathsf{PRF},\mathcal{A}_{\mathrm{prf}}}(\kappa), \\ \mathsf{Time}^*(\mathcal{A}_{\mathrm{prf}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF}) + \mathsf{Time}(B_\epsilon)), \\ \mathsf{Mem}^*(\mathcal{A}_{\mathrm{prf}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF}) + \mathsf{Mem}(B_\epsilon)), \end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{X}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M}, \mathcal{Y})$.

Game G_2*:* We next modify the signing oracle and the random oracle. In this game, the signing oracle given m returns $\sigma = \mathsf{Sample}(vk, \mathsf{RF}_\mathsf{S}(m))$ (on unfiltered m) and the random oracle given m returns $\mathsf{F}(vk, \mathsf{Sample}(vk, \mathsf{RF}_\mathsf{S}(m)))$. By applying Lemma 2.2 with ϵ-simulatability, we have

$$|\Pr[W_1] - \Pr[W_2]| \leq \sqrt{(6(q_S + q_H + q_F))^3 \epsilon}.$$

Game G_3*:* We finally modify how to update the flag win. In G_3, the flag is set true if the submitted forgery σ^* is different from the expected one σ' and $(m^*, \sigma^*) \notin B_\epsilon$.

Lemma D.7. *Suppose that* PSF *has α-preimage min-entropy. We have*

$$|\Pr[W_2] - \Pr[W_3]| \leq q_F \cdot 2^{-\alpha}.$$

Proof. Let (m^*, σ^*) be a query to Forge the adversary made. Let $\sigma' := \mathsf{Sample}(vk; \mathsf{RF}_\mathsf{S}(m^*))$. The two games differ if the adversary submits a valid pair $(m^*, \sigma^*) \in B_\epsilon$ but $\sigma^* = \sigma'$. Let us consider two cases:

1. If $(m^*, \sigma') \notin B_\epsilon$, then this contradicts with $(m^*, \sigma') = (m^*, \sigma^*) \in B_\epsilon$. Thus, we do not need to consider this case.
2. If $(m^*, \sigma') \in B_\epsilon$, then the adversary cannot know σ' from B_ϵSign. Due to α-preimage min-entropy of PSF, the probability that $\sigma^* = \sigma'$ is at most $q_F \cdot 2^{-\alpha}$.

Thus, we obtain the bound. □

Now, making a memory-tight reduction for collision-resistance of PSF is easy.

Lemma D.8. *There exists a quantum $\mathcal{F}$-oracle adversary $\mathcal{A}_{\mathrm{cr}}$ such that*

$$\begin{aligned} \Pr[W_3] &\leq \mathsf{Adv}^{\mathrm{cr}}_{\mathsf{PSF},\mathcal{A}_{\mathrm{cr}}}(\kappa), \\ \mathsf{Time}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Time}(\mathcal{A}) + (q_H + q_S + q_F) \cdot O(\mathsf{Time}(\mathsf{PSF}) + \mathsf{Time}(B_\epsilon)), \\ \mathsf{Mem}^*(\mathcal{A}_{\mathrm{cr}}) &= \mathsf{Mem}(\mathcal{A}) + O(\mathsf{Mem}(\mathsf{PSF}) + \mathsf{Mem}(B_\epsilon)), \end{aligned}$$

where $\mathcal{F} = \mathsf{Func}(\mathcal{M} \times \mathcal{X}, \mathcal{P}) \times \mathsf{Func}(\mathcal{M}, \mathcal{R}_{\mathsf{Sample}})$.

Since the reduction is straightforward, we omit it.