Faster Kyber and Saber via a Generic Fujisaki-Okamoto Transform for Multi-User Security in the QROM

Julien Duman [1] Kathrin Hövelmanns [2] Eike Kiltz [1]
Vadim Lyubashevsky [3] Gregor Seiler [3,4]

April 23, 2021

[1] Ruhr-Universität Bochum
{julien.duman,eike.kiltz}@rub.de
[2] Eindhoven University of Technology
kathrin@hoevelmanns.net
[3] IBM Research Europe, Zurich
vad@zurich.ibm.com
[4] ETH Zürich
gseiler@inf.ethz.ch

Abstract

Constructing an efficient CCA-secure KEM is generally done by first constructing a passively-secure PKE scheme, and then applying the Fujisaki-Okamoto (FO) transformation. The original FO transformation was designed to offer security in a single user setting. A stronger notion, known as multi-user security, considers the attacker's advantage in breaking one of many user's ciphertexts. Bellare et al. (EUROCRYPT 2020) showed that standard single user security implies multi-user security with a multiplicative tightness gap equivalent to the number of users.

In order to achieve multi-user security in the random oracle model, it is a common design paradigm to "domain separate" the random oracles of each user by including his public key as an input to the hash function. We are not aware of any formal security analysis of this technique, but it was at least informally thought to be more secure. This design principle was carried over into the FO transformations used by several schemes in the NIST post-quantum standardization effort – notably the lattice-based schemes Kyber and Saber, which are two of the four third round KEM finalists.

In this work, we formally analyze domain separation in the context of the FO transformation in the multi-user setting. We first show that including the public key in the hash function is important for the tightness of the security reductions in the ROM and the QROM. At the same time, we show that including the *entire* public key into the hash function is unnecessarily wasteful – it is enough to include just a small (e.g. 32 byte) unpredictable part of the key to achieve the same security. Reducing the input of the hash function results in a very noticeable improvement in the running time of the lattice-based KEMs. In particular, using this generic transform results in a 2X - 3X speed-up over the current (Round 3) key generation and encapsulation procedures in Kyber, and up to a 40% improvement in the same functions in Saber.

Keywords: FO Transform, QROM, Key Exchange, Lattice Cryptography, Implementation

1 Introduction

Security definitions for public key encryption (PKE) schemes are generally stated in the *single-user* setting. In this setting, one party publishes its public key, which allows other parties to send it encrypted messages of their choice. For practical applications, however, this was shown to not be enough. In particular, Håstad showed [Hås88] that if there are multiple receivers, each with a different public key, and a sender encrypts the same message to all of them, then for certain RSA parameter settings everyone will be able to recover the message.

Håstad's simple attack against the basic RSA cryptosystem demonstrated that schemes can be secure in the single user setting, but completely broken in the multi-user one. Luckily, Bellare et al. [BBM00] showed that for CPA and CCA-secure schemes, security in the single-user setting implies security (with a multiplicative loss in the number of parties) in the multi-user setting as well .

Constructing an efficient Key Encapsulation Mechanism (KEM) with security against chosen-ciphertext attacks (CCA) is generally done by first constructing a passively-secure PKE scheme, and then applying the Fujisaki-Okamoto (FO) transformation [FO99, FO13, HHK17]. To achieve multi-user security in the random oracle model, it is a common design paradigm to use "domain separation" so that in the random oracle model, the parties all appear to be using different random functions. The simple way of achieving this in the context of the FO transformation is to always include the public key as an additional input to the cryptographic hash function that is being modeled as a random oracle. The other stated reason for including the public key in the hash is an informally-defined notion of making the KEM "contributory" – that is, both parties affect the shared key.

For classical schemes based on the hardness of the discrete logarithm problem, including the public key as an extra input to the hash function has virtually no effect on the running time of the full protocol. The reason is that the key is short (e.g. 32 bytes), and the hash function contributes a negligible amount of computation compared to the much more expensive group operations such as exponentiation. So even if unnecessary for practical security, adding the public key into the hash function does not have any measurable negative effects on the scheme.

The above-mentioned domain separation technique was carried over to schemes in the ongoing NIST post-quantum standardization process. For example, Kyber and Saber use the above-described method with the explicit purpose of protecting against multi-user attacks. Even though it was at least informally thought to be more secure, we are not aware of any formal security analysis of it.

1.1 Our Results

In this paper we formally analyze domain separation in the context of the FO transformation. Our results are twofold. First, we observe that there are good reasons for including the public key into the hash. We show that hashing the public key results in a tighter reduction when converting a CPA-secure scheme into a CCA-secure one, than if one were to directly apply the hybrid argument from [BBM00] to the FO transformation. And for schemes that additionally have a small correctness error, the tightness in the reduction is potentially even more improved. We additionally give a proof in the QROM which is significantly tighter that what one would trivially obtain from [BBM00]. This proof also appears to use (but in a different way) the fact that each party uses a different random function.

Our second, and main, result is that even though there are reasons to hash the public key, hashing the *entire* public key (as is currently done) is unnecessarily wasteful. The sizes of public keys in lattice-based schemes ($\approx$ 1KB) are noticeably larger than the 32 byte keys used in the discrete

log setting; so the hash function that takes the public key as input is now approximately an order of magnitude slower. At the same time, the underlying lattice-based CPA-secure scheme is significantly *more* computationally efficient than its discrete logarithm counterpart. When compounded, these two properties result in the hash function being a very significant contributor to the total running time of the resulting CCA-secure scheme.

Our new variant of the FO transformation, $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$, transforms a passively secure PKE scheme into an actively secure KEM. It resembles the FO variant $\mathsf{FO}^{\not\perp}_{m}$ of [HHK17]: It also uses "implicit rejection" and the only difference to $\mathsf{FO}^{\not\perp}_{m}$ is that it feeds a small (e.g. 32 byte), unpredictable part of the public key into the hash function. ($\mathsf{FO}^{\not\perp}_{m}$ does not include any part of pk in the hash function.) Compared to feeding the whole public-key into the hash function, this significantly reduces the running time of the scheme. In the ROM we prove that multi-user CPA-security of PKE tightly implies multi-user CCA-secure KEM. As an additional result, we give a reduction for multi-user security in the QROM which is tighter than the previously-known results which stem from the generic Bellare et al. result [BBM00].

Instantiating our transformation $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ with the CPA-secure Kyber PKE scheme, the key generation and encryption of Kyber is reduced by 30-56% and 47-66%, respectively; and the security of the CCA-scheme in the ROM and QROM is improved as in Figure 1. So we are now in the same situation with lattice schemes as we are in the discrete logarithm setting. There is no longer a theory vs. practice trade-off required for achieving multi-user security – hashing the (partial) public key leads to tighter security reductions and is computationally very cheap to implement.

1.2 Impact on Concrete Security

In the Fujisaki-Okamoto transform, a hash function F (modeled as a random oracle) is mainly used to derive the PKE randomness and the symmetric KEM key from a random message m. In the following table, we define three variants of the "implicit rejection" Fujisaki-Okamoto transformation, depending on which parts of pk it includes in the hash function F.

Transformation	Use of F
$\mathsf{FO}^{\not\perp}_{m}$ [HHK17]	$\mathsf{F}(m)$
$\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ (§ 3)	$\mathsf{F}(\mathsf{ID}(pk), m)$
$\mathsf{FO}^{\not\perp}_{pk,m}$ (§ 3)	$\mathsf{F}(pk, m)$

The formal definitions of $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ and $\mathsf{FO}^{\not\perp}_{pk,m}$ are given in Figure 5 of Section 3. $\mathsf{FO}^{\not\perp}_{m}$ was formally analyzed in [HHK17]. We remark that [HHK17] also considered a variant (called $\mathsf{FO}^{\not\perp}_{m,c}$) where the ciphertext c is included in F. Follow-up work [BHH+19] showed that hashing c is not necessary so we do not further consider it here.

Figure 1 compares multi-user security bounds in the ROM/QROM of $\mathsf{FO}^{\not\perp}_{m}$, $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$, and $\mathsf{FO}^{\not\perp}_{pk,m}$. The bounds for $\mathsf{FO}^{\not\perp}_{m}$ are obtained by applying the Bellare et al. hybrid argument to the single user CCA-bounds of [HHK17]. The bounds for $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ and $\mathsf{FO}^{\not\perp}_{pk,m}$ are new bounds obtained in this work. Our proofs in the ROM rely on techniques of [HHK17], while the ones in the QROM also rely on techniques of [BHH+19, HKSU20, JZM19].

First we note that the bounds for $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ and $\mathsf{FO}^{\not\perp}_{pk,m}$ are exactly the same (for prefixes with sufficient min-entropy l). Hence from a security perspective, it does not seem to make much sense to include the whole public key in the hash function. Hence in what follows we will only consider the prefix-hash variant $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$. How the bounds for $\mathsf{FO}^{\not\perp}_{m}$ and $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ compare to each other, depends on the relation between $\delta(n)$ and δ and $\mathsf{Adv}^{\mathsf{n\text{-}IND\text{-}CPA}}$ and $\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}$, respectively.

FO variant	$\mathsf{Adv}^{\text{n-IND-CCA}}$ (ROM)	$\mathsf{Adv}^{\text{n-IND-CCA}}$ (QROM)
$\mathsf{FO}^{\not\perp}_{m}$ [HHK17, BBM00]	$n \cdot \mathsf{Adv}^{\text{IND-CPA}} + n \cdot q_{\mathsf{F}}\delta(1)$	$n \cdot \sqrt{q_{\mathsf{F}}\mathsf{Adv}^{\text{IND-CPA}}} + nq_{\mathsf{F}}^2\delta(1)$
$\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ (Th. 3.1+3.2)	$\mathsf{Adv}^{\text{n-IND-CPA}} + q_{\mathsf{F}}\delta(n) + \frac{n^2}{2^l}$	$\sqrt{q_{\mathsf{F}}\mathsf{Adv}^{\text{n-IND-CPA}}} + q_{\mathsf{F}}^2\delta(n) + \frac{n^2}{2^l}$
$\mathsf{FO}^{\not\perp}_{pk,m}$ (Th. 3.1+3.2)	$\mathsf{Adv}^{\text{n-IND-CPA}} + q_{\mathsf{F}}\delta(n)$	$\sqrt{q_{\mathsf{F}}\mathsf{Adv}^{\text{n-IND-CPA}}} + q_{\mathsf{F}}^2\delta(n)$

Figure 1: Multi-user security advantages $\mathsf{Adv}^{\text{n-IND-CCA}}$ in the ROM/QROM (simplified) for the Fujisaki-Okamoto variants $\mathsf{FO}^{\not\perp}_{m}$, $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$, and $\mathsf{FO}^{\not\perp}_{pk,m}$, as described in the text. Here $\delta(n)$ is the n-user correctness error of PKE, q_{F} is the number of (Q)ROM queries, and l is the min-entropy of $\mathsf{ID}(pk)$.

- **n-user correctness error $\delta(n)$.** $\delta(n)$ is defined as the n-user correctness error, i.e., the probability that one message induces a decryption error for one of n independent public-keys. (We refer to Section 2.1 for a formal definition.) Trivial bounds are $\delta(1) \leq \delta(n) \leq n\delta(1)$. Whereas there can exist schemes with $\delta(n) = \delta(1)$, for concrete applications like Kyber and Saber we believe that $\delta(n) \approx n\delta(1)$.

- **n-user CPA advantage $\mathsf{Adv}^{\text{n-IND-CPA}}$.** By a hybrid argument one obtains the trivial bounds

$$\mathsf{Adv}^{\text{IND-CPA}} \leq \mathsf{Adv}^{\text{n-IND-CPA}} \leq n\mathsf{Adv}^{\text{IND-CPA}}.$$

  For schemes based on prime-order groups (eg., ElGamal) we actually have $\mathsf{Adv}^{\text{DDH}} \approx \mathsf{Adv}^{\text{n-IND-CPA}}$ by the well known random self reducibility of DDH.

  For Kyber we have $\mathsf{Adv}^{\text{n-IND-CPA}} = \mathsf{Adv}^{n\text{-MLWE}}$, where n-MLWE is an n-instance variant of MLWE. While there is no known self-reduction for n-MLWE, it is quite unlikely that distinguishing $\{(\mathbf{A}_i, \mathbf{A}_i \cdot \vec{\mathbf{s}}_i + \vec{\mathbf{e}}_i)\}_{1 \leq i \leq n}$ from uniform is easier than just distinguishing one sample $(\mathbf{A}_1, \mathbf{A}_1 \cdot \vec{\mathbf{s}}_1 + \vec{\mathbf{e}}_1)$. In particular, there is a reduction to n-MLWE from Module-LWE where the number of Module-LWE samples (for the same secret) is $2n$.[1] And unless we are in a regime where the number of samples is so large that the Arora-Ge [AG11] attack applies, it is not known that seeing more samples makes the Module LWE problem any easier. Furthermore, asymptotically, the worst-case to average-case reductions do not restrict the number of samples of the average-case (Module)-LWE problem[Reg05, LPR10, LS15]. We therefore believe that in practice $\mathsf{Adv}^{n\text{-MLWE}} \approx \mathsf{Adv}^{\text{MLWE}}$, and then the bound in our reduction is significantly tighter than the previously known one. And independently of the relationship between MLWE and n-MLWE, the new QROM bound for $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ is also noticeably better than the one for $\mathsf{FO}^{\not\perp}_{m}$.

1.3 Impact on Efficiency

We now measure the effect of replacing the original FO transform used in the Kyber and Saber KEMs with our new transform $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ (see Figure 5). We implemented $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ on top of the underlying CPA-secure encryption schemes in the official AVX2-optmized implementations of Kyber and Saber, and performed benchmarks on an Intel Skylake CPU. Concretely, the numbers in Tables 1 and 2 are the medians of the cycle counts of 10000 executions of the key generation (**K**), encapsulation (**E**), and decpasulation (**D**) operations for either the original FO transform or the new one from this work.

[1] The reason that it's $2n$ samples instead of n is that we need an extra n samples for converting Module-LWE with random secrets to Module-LWE with small ones. [ACPS09]

Table 1: Median Skylake cycle counts of 10000 executions of Kyber and Saber using either the original CCA transform or the improved transform from this work. The "original" version of Kyber and Saber are the Round 3 submissions to the NIST post-quantum standardization process.

		Kyber			Saber		
NIST Level		Original	This Work	Speed-up	Original	This Work	Speed-up
1	**K**	23562	12883	45%	42169	36220	14%
	E	37144	16981	54%	57831	39232	32%
	D	28595	28529	0%	57780	57806	0%
3	**K**	40487	25272	38%	74577	64180	14%
	E	55726	27624	50%	95958	69304	28%
	D	43553	43442	0%	95388	95301	0%
5	**K**	55770	38815	30%	116178	102101	12%
	E	77011	40692	47%	142034	109203	23%
	D	61470	61473	0%	142957	143090	0%

Table 2: Median Skylake cycle counts of 10000 executions of the 90's variants of Kyber and Saber using either the original CCA transform or the improved transform from this work. The "original" version of Kyber and Saber are the Round 3 submissions to the NIST post-quantum standardization process.

		Kyber90s			uSaber90s		
NIST Level		Original	This Work	Speed-up	Original	This Work	Speed-up
1	**K**	13994	6224	56%	24557	17294	30%
	E	23069	7894	66%	36544	22363	39%
	D	16917	16959	0%	38156	38014	0%
3	**K**	21783	10995	50%	37511	29881	20%
	E	33534	13137	61%	55436	36282	35%
	D	25014	24957	0%	58395	58405	0%
5	**K**	31576	18834	40%	59169	50796	14%
	E	46881	21404	54%	81187	57920	29%
	D	36190	36165	0%	86142	86359	0%

For the 32-byte prefix $\mathsf{ID}(pk)$ of the public key in Kyber and Saber one can take the seed ρ that is already of size 32 bytes and uniformly random in these schemes. Alternatively, for Kyber, the first 33 bytes of the bitpacked representation of the polynomial vector $\vec{\mathbf{t}}$ in the public key can also be used. This is sufficient since $\vec{\mathbf{t}}$ is given in the NTT basis and contains the independently random short error vector $\vec{\mathbf{e}}$ as an additive term. It follows from a simple Fourier analysis computation as in [ALS20, CLS16] that the first few NTT coefficient of $\vec{\mathbf{e}}$ are close to uniform modulo $q = 3329$. Concretely, the first 22 coefficients have more than 256 bit of entropy and they occupy 33 bytes in the bitpacking that uses 12 bits per coefficient. Taking the prefix from $\vec{\mathbf{t}}$ instead of ρ has the advantage that this is still secure in the slightly modified but fully compatible variant of Kyber where users re-use ρ and hence the MLWE matrix $\mathbf{A}$.

We observe that one obtains significant speed improvements throughout all parameter sets and variants. For example, the key generation and encapsulation of Kyber-512-90s are 56% and 66% faster, respectively, when using $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$.[2] There is no speed-up in decryption since in the original Kyber and Saber CCA transforms avoid the expensive full public key hash needed for the re-encryption during decapsulation by pre-computing this hash during key generation and storing it in the secret key. Our new transform also has a noticeably larger effect on Kyber than Saber because the CPA-secure scheme underlying Kyber is more efficient than its Saber counterpart. Thus the running time of Kyber with the original FO transform was much more dominated by hashing.

1.4 Open Problems

One of the biggest performance gaps between theory and practice that remains in $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}$ is that Decaps always has to compute $\tilde{K} \leftarrow \mathsf{F}_1(\mathsf{ID}(pk), s, c)$ (cf. line 10 of Figure 5), independent of c being consistent or inconsistent. Only computing $\tilde{K}$ for inconsistent c would give rise to a simple timing attack that reveals whether a given c is consistent or not. It remains an open problem to prove the security of a corresponding FO transformation with "explicit rejection" in the QROM.

2 Preliminaries

For $n \in \mathbb{N}$, let $[n] := \{1, \ldots, n\}$. For a set S, $|S|$ denotes the cardinality of S. For a finite set S, we denote the sampling of a uniform random element x by $x \xleftarrow{\$} S$. The min entropy of a discrete random variable X is defined as $H_\infty(X) = -\log(\max_x \Pr[X = x])$.

2.1 Cryptographic Definitions

Public-Key Encryption. A public-key encryption scheme $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ consists of three algorithms, and a finite message space $\mathcal{M}$. The key generation algorithm Gen outputs a key pair (pk, sk), where pk also defines a finite randomness space $\mathcal{R} = \mathcal{R}(pk)$ as well as a ciphertext space $\mathcal{C}$. The encryption algorithm Enc, on input pk and a message $m \in \mathcal{M}$, outputs an encryption $c \xleftarrow{\$} \mathsf{Enc}(pk, m)$ of m under the public key pk. If necessary, we make the used randomness of encryption explicit by writing $c := \mathsf{Enc}(pk, m; r)$, where $r \in \mathcal{R}$. The decryption algorithm Dec, on input sk and a ciphertext c, outputs either a message $m = \mathsf{Dec}(sk, c) \in \mathcal{M}$ or a special symbol $\perp \notin \mathcal{M}$ to indicate that c is not a valid ciphertext.

[2] It should be mentioned again that another reason that encapsulation has a larger increase is that we removed an additional hash of the ciphertext. This was already shown to be intuitively unnecessary in [BHH+19], and in this work we show that it is also unnecessary in the multi-user setting.

PKE has n-user correctness error $\delta(n)$ if for all (even unbounded) adversaries $\mathcal{A}$ we have

$$\Pr\left[\mathsf{n\text{-}COR}^{\mathcal{A}}_{\mathsf{PKE}} \Rightarrow 0\right] \leq \delta(n)\,, \tag{1}$$

where $\mathsf{n\text{-}COR}_{\mathsf{PKE}}$ is defined in Figure 2. For $n = 1$ we obtain the single-user correctness definition $\delta := \delta(1)$ of [HHK17]. Note that $\delta \leq \delta(n) \leq n\delta$.

GAME n-COR:
01 for $j \in [n]$
02 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{Gen}$
03 $\vec{sk} \leftarrow (sk_1, \ldots, sk_n)$
04 $\vec{pk} \leftarrow (pk_1, \ldots, pk_n)$
05 $(m, j) \xleftarrow{\$} \mathcal{A}(\vec{sk}, \vec{pk})$
06 $c \xleftarrow{\$} \mathsf{Enc}(pk_j, m)$
07 return $\mathsf{Dec}(sk_j, c) = m$

Figure 2: Correctness game n-COR for PKE.

In terms of PKE's security, we consider the following advantage functions:

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{n\text{-}OW\text{-}CPA}}(\mathcal{A}) &:= \Pr[\mathsf{n\text{-}OW\text{-}CPA}^{\mathcal{A}} \Rightarrow 1] \\
\mathsf{Adv}^{\mathsf{n\text{-}IND\text{-}CPA}}(\mathcal{A}) &:= \Pr[\mathsf{n\text{-}IND\text{-}CPA}^{\mathcal{A}} \Rightarrow 1] - \frac{1}{2},
\end{aligned}$$

where the games are defined in Figure 3.

n-OW-CPA
01 for $j \in [n]$
02 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{Gen}$
03 $\quad m_j \xleftarrow{\$} \mathcal{M}$
04 $\quad c^*_j \xleftarrow{\$} \mathsf{Enc}(pk_j, m_j)$
05 $\vec{pk} \leftarrow (pk_1, \ldots, pk_n)$
06 $\vec{c} \leftarrow (c^*_1, \ldots, c^*_n)$
07 $(j, m') \xleftarrow{\$} \mathcal{A}(\vec{pk}, \vec{c}^*)$
08 return $m' = m_j$

n-IND-CPA
09 $b \xleftarrow{\$} \{0, 1\}$
10 for $j \in [n]$
11 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{Gen}$
12 $\vec{pk} \leftarrow (pk_1, \ldots, pk_n)$
13 $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{CHAL}}(\vec{pk})$
14 return $b' = b$

$\mathsf{CHAL}(\vec{m}_0, \vec{m}_1)$ // one query
15 return $\mathsf{Enc}(\vec{pk}, \vec{m}_b)$

Figure 3: Games n-OW-CPA and n-IND-CPA for PKE.

Key Encapsulation Mechanisms. A key encapsulation mechanism $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ consists of three algorithms and a finite key space $\mathcal{K}$ similar to a PKE scheme, but Encaps does not take a message as input. The key generation algorithm Gen outputs a key pair (pk, sk), where pk also defines a finite randomness space $\mathcal{R} = \mathcal{R}(pk)$ as well as a ciphertext space $\mathcal{C}$. The encapsulation algorithm Encaps takes as input a public-key pk and outputs a key encapsulation ciphertext c and a key k, that is $(c, k) \xleftarrow{\$} \mathsf{Encaps}(pk)$. We write $c \leftarrow \mathsf{Encaps}_1(pk; r)$ and $k \leftarrow \mathsf{Encaps}_2(pk; r)$, if we want to seperate the two outputs. If necessary, we make the used randomness of encryption explicit by writing $(c, k) := \mathsf{Encaps}(pk; r)$, where $r \in \mathcal{R}$. The decapsulation algorithm Decaps, on input sk and

a ciphertext c, outputs either a key $k = \mathsf{Decaps}(sk, c) \in \mathcal{K}$ or a special symbol $\perp \notin \mathcal{K}$ to indicate that c is not a valid ciphertext.

We say KEM has correctness error δ if

$$\Pr[\mathsf{Decaps}(sk, \mathsf{Encaps}_1(pk; r)) \neq \mathsf{Encaps}_2(pk; r)] \leq \delta\,, \tag{2}$$

where the probability is taken over $r \xleftarrow{\$} \mathcal{R}$ and $(pk, sk) \xleftarrow{\$} \mathsf{Gen}$.

In terms of KEM's security, we consider the advantage function

$$\mathsf{Adv}^{\mathsf{n\text{-}IND\text{-}CCA}}(\mathcal{A}) \quad := \quad \Pr[\mathsf{n\text{-}IND\text{-}CCA}^{\mathcal{A}} \Rightarrow 1] - \frac{1}{2}$$

where the game is defined in Figure 4.

<u>n-IND-CCA</u>
01 $b \xleftarrow{\$} \{0,1\}$
02 for $j \in [n]$
03 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{Gen}$
04 $\quad (K^*_{j0}, c^*_j) \xleftarrow{\$} \mathsf{Encaps}(pk_j)$
05 $\quad K^*_{j1} \xleftarrow{\$} \mathcal{K}$
06 $\vec{pk} \leftarrow (pk_1, \ldots, pk_n)$
07 $\vec{c} \leftarrow (c^*_1, \ldots, c^*_n)$
08 $\vec{K}_b \leftarrow (K^*_{1b}, \ldots, K^*_{nb})$
09 $b' \xleftarrow{\$} \mathcal{A}(\vec{pk}, \vec{c}, \vec{K}_b)$
10 return $b = b'$

Figure 4: Game $\mathsf{n\text{-}IND\text{-}CCA}$ for KEM.

3 Fujisaki-Okamoto Transformation with Prefix Hashing

Let PKE be a public-key encryption scheme with message space $\mathcal{M}$ and randomness space $\mathcal{R}$ and let $\mathsf{ID} : \mathcal{PK} \to \{0,1\}^\gamma$ be a fixed-output length function. Let $\mathsf{F} : \{0,1\}^* \to \{0,1\}^k \times \mathcal{R}$ be a hash function and define $\mathsf{F}_1(X)$ as the first k bits of $\mathsf{F}(X)$. We build $\mathsf{KEM} = \mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}[\mathsf{PKE}, \mathsf{ID}, \mathsf{F}] = (\mathsf{Gen}', \mathsf{Encaps}, \mathsf{Decaps})$ as described in Figure 5. Our construction is essentially $\mathsf{FO}^{\not\perp}_m$ of [HHK17] with the difference that we feed $\mathsf{ID}(pk)$ into the hash function F for domains separation. Note that with the identity function $\mathsf{ID}(pk) = pk$ we recover $\mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m} = \mathsf{FO}^{\not\perp}_{pk,m}$ also mentioned in the introduction.

We remark that computing $\tilde{K}$ in line 10 of $\mathsf{Decaps}(sk, c)$ is only required in case c is inconsistent. We still recommend to always compute $\tilde{K}$ because otherwise the system might be prone to a simple side-channel attack.

We now state the two main theorems about KEM's security in the ROM and QROM, respectively. In the concrete security statements we will use the following terms

- n-user correctness error $\delta(n)$
- Min entropy l of $\mathsf{ID}(pk)$, i.e., $l := H_\infty(pk)$, where $(pk, sk) \xleftarrow{\$} \mathsf{Gen}$
- Bit-length λ of the secret seed $s \in \{0,1\}^\lambda$
- Max. number of (Q)ROM queries q_F

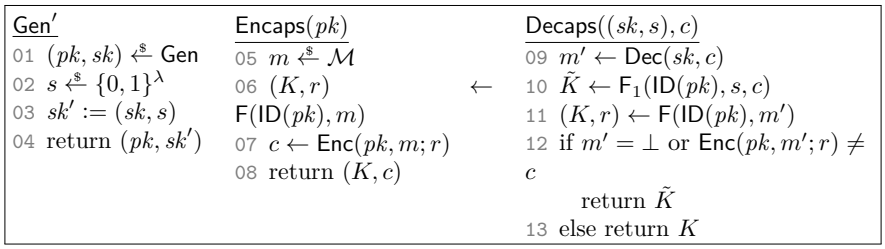

Figure 5: $\mathsf{KEM} = \mathsf{FO}^{\not\perp}_{\mathsf{ID}(pk),m}[\mathsf{PKE}, \mathsf{ID}, \mathsf{F}]$ built from PKE, F, and ID.

- Max. number of decapsulation queries q_{Decaps}

Theorem 3.1 (n-IND-CPA of PKE $\overset{\text{ROM}}{\Longrightarrow}$ n-IND-CCA of KEM). *For any adversary $\mathcal{A}$ against the* n-IND-CCA *security of KEM there exists an adversary $\mathcal{B}$ against* n-IND-CPA *of PKE (with roughly the same running time) such that*

$$\mathsf{Adv}^{\text{n-IND-CCA}}_{KEM}(\mathcal{A}) \leq 2\mathsf{Adv}^{\text{n-IND-CPA}}_{PKE}(\mathcal{B}) + \frac{2q_{\mathsf{F}}}{|\mathcal{M}|} + \frac{n^2}{2^l} + \frac{q_{\mathsf{F}}}{2^\lambda} + (q_{\mathsf{F}} + q_{\mathsf{Decaps}})\delta(n).$$

The proof of Theorem 3.1 is given in Section 4.

Theorem 3.2 (n-IND-CPA of PKE $\overset{\text{QROM}}{\Longrightarrow}$ n-IND-CCA of KEM). *For any quantum adversary $\mathcal{A}$ against the* n-IND-CCA *security of KEM there exists a quantum adversary $\mathcal{B}$ against* n-IND-CPA *of PKE (with roughly the same running time) such that* $\mathsf{Adv}^{\text{n-IND-CCA}}_{KEM}(\mathcal{A}) \leq$

$$4\sqrt{q_{\mathsf{F}}\mathsf{Adv}^{\text{n-IND-CPA}}_{PKE}(\mathcal{B})} + 4q_{\mathsf{F}}\sqrt{\frac{n}{|\mathcal{M}|}} + \frac{n^2}{2^l} + 2q_{\mathsf{F}}\sqrt{\frac{n}{2^\lambda}} + 32(q_{\mathsf{F}} + q_{\mathsf{Decaps}})^2\delta(n)\,.$$

The proof of Theorem 3.2 is given in Section 5.

4 Proof of Theorem 3.1

Proof. Let $\mathcal{A}$ be an adversary and consider the games given in Figure 6.

GAME G_0. This is the original n-IND-CCA game.

$$\left|\Pr[G_0 \Rightarrow 1] - \frac{1}{2}\right| = \mathsf{Adv}^{\text{n-IND-CCA}}(\mathcal{A})\,.$$

GAME G_1. In game G_1 an abort condition COLL is introduced, which aborts if there is a collision in the public-key identifiers $\mathsf{ID}(pk_j)$. In case there no collision in the identifiers, we are able to identity each identifier $\mathsf{ID}(pk_j)$ with a unique index j. This allows us to internally simulate

$$\mathsf{F}(\mathsf{ID}(pk_j)), A) := \begin{cases} (\mathsf{H}_j(m), \mathsf{G}_j(m)) & A = m \\ \mathsf{K}_j(s, c) & A = (s, c) \end{cases},$$

where G_j and H_j are internal random oracles. (This implicitly assumes that the two spaces $\mathcal{M}$ and $\{0,1\}^k \times \mathcal{C}$ are disjoint.) Since the two games are identical until COLL happens, we have by the birthday bound

$$\left|\Pr[G_0 \Rightarrow 1] - \Pr[G_1 \Rightarrow 1]\right| \leq \Pr[\text{COLL}] \leq \frac{n^2}{2^l}\,,$$

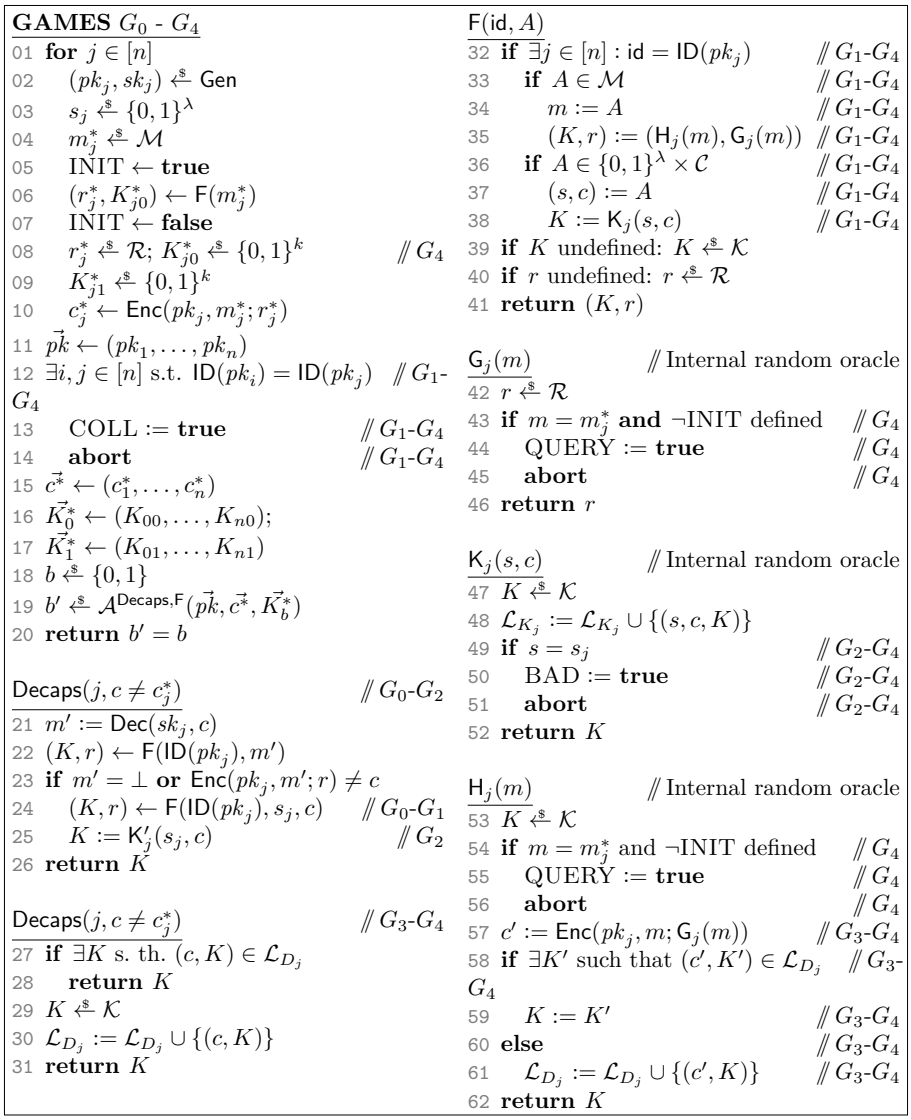

Figure 6: Games G_0 - G_4 for the proof of Theorem 3.1. $\mathsf{K}'_j, \mathsf{H}_j, \mathsf{G}_j, \mathsf{K}_j$ are internal random oracles not accessible by the adversary. We assume wlog that F is only queried once on each value A.

where $l = H_\infty(\mathsf{ID}(pk))$ is the min entropy of pk for $(pk, sk) \xleftarrow{\$} \mathsf{Gen}$.

Game G_2. In game G_2 we modify the Decaps oracle in lines 24 and 25 such that for an invalid ciphertext the key is defined as $\mathsf{K}'_j(s_j, c)$, where K'_j is an independent internal random oracle. This remains unnoticed to adversary $\mathcal{A}$ unless it queries

$$\mathsf{H}_j(s_j, \cdot)$$

for some $j \in [n]$. Since the seeds $s_j \in \{0,1\}^\lambda$ are uniformly random and information-theoretically hidden from the adversary, we have by the union bound

$$\Big|\Pr[G_1 \Rightarrow 1] - \Pr[G_2 \Rightarrow 1]\Big| \leq \frac{q_\mathsf{F}}{2^\lambda}\,.$$

GAME G_3. In game G_3 we simulate the decapsulation oracle $\mathsf{Decaps}(j,\cdot)$ without knowledge of the secret key by patching the random oracles H_j in lines 57-61. Note that if PKE was perfectly correct, then the random oracle patching is also perfectly correct and therefore two games would look identical in $\mathcal{A}$'s view.

The only bad case happens if G_j is queried on some m which induces a correctness error, that is $\mathsf{Dec}(sk_j, \mathsf{Enc}(pk_j; m; \mathsf{G}_j(m))) \neq m$. More concretely, define the sets

$$\mathrm{BAD}_j := \{m \in \mathcal{M} \mid m \neq m', \text{where } c \leftarrow \mathsf{Enc}(pk_j; m; \mathsf{G}_j(m)); m' \leftarrow \mathsf{Dec}(sk_j, c)\}$$

Define the event CORR to be the event that $\mathcal{A}$ makes an (implicit) query to $\mathsf{G}_j(m)$ for some $m \in \mathrm{BAD}_j$. Since there are at most $(q_{\mathsf{F}} + q_{\mathsf{Decaps}})$ explicit and implicit queries to G_j, we have

$$\Pr[\mathrm{CORR}] \leq (q_{\mathsf{F}} + q_{\mathsf{Decaps}})\delta(n)\,.$$

We now claim that

$$\Big|\Pr[G_2 \Rightarrow 1] - \Pr[G_3 \Rightarrow 1]\Big| \leq \Pr[\mathrm{CORR}]\,.$$

Let us analyze why G_2 and G_3 are identical conditioned on $\neg$CORR. Consider a query $\mathsf{Decaps}(j,c)$ and define $m' := \mathsf{Dec}(sk_j, c)$ and $c' := \mathsf{Enc}(pk_j, m'; \mathsf{G}_j(m'))$.

- **Case 1:** $m' = \perp$. H_j cannot be called on $m' = \perp$ and hence the KEM key of $K = \mathsf{Decaps}(sk_j, c) = \mathsf{K}'_j(c)$ in G_2 is identically distributed as $(c, K) \in \mathcal{L}_{D_j}$ in G_3.
- **Case 2:** $m' \neq \perp \wedge c \neq c'$**.** Both games return a uniform random key K. The only way for $\mathcal{A}$ to detect a difference between the two games is if it makes a query $\mathsf{H}_j(m)$ such that $\mathsf{Enc}(pk_j, m; \mathsf{G}_j(m)) = c$. (In G_3, $\mathsf{H}_j(m)$ would return the same key K as in $\mathsf{Decaps}(j,c)$, whereas in G_2 the two keys would be independent.) Since $c \neq c'$ we also have $m \neq m'$. But such a query $\mathsf{H}_j(m)$ would internally involve the query $\mathsf{G}_j(m)$ for $m \in \mathrm{BAD}_j$.
- **Case 3:** $m' \neq \perp \wedge c = c'$**.** In game G_2, $\mathsf{Decaps}(j,c)$ returns $K = \mathsf{H}_j(m')$, whereas G_3 first picks a uniform K and patches $\mathsf{H}_j(m)$ to match K for all m that deterministically encrypt to the same c, i.e., all m satisfying $\mathsf{Enc}(pk_j, m; \mathsf{G}_j(m)) = c$. The only way for $\mathcal{A}$ to detect a difference between the two games is to query H_j on some value $m \neq m'$ that also deterministically encrypts to the same c. But this also implies that $\mathsf{G}_j(m)$ was queried for some $m \in \mathrm{BAD}_j$.

GAME G_4. In game G_4 we abort on queries of the form $\mathsf{H}_j(m_j^*)$ or $\mathsf{G}_j(m_j^*)$, in which case the event QUERY holds true. We use the flag INIT to avoid triggering QUERY in Line 06. We have by the difference lemma,

$$\Big|\Pr[G_3 \Rightarrow 1] - \Pr[G_4 \Rightarrow 1]\Big| \leq \Pr[\mathrm{QUERY}]\,. \tag{3}$$

Note that in G_4 bit b is independent of the view of the adversary. We thus have

$$\Big|\Pr[G_4] \Rightarrow 1\Big| = \frac{1}{2}\,.$$

We claim that

$$\Pr[\mathrm{QUERY}] \leq 2 \cdot \left(\mathsf{Adv}^{\mathsf{n\text{-}IND\text{-}CPA}}_{\mathsf{PKE}}(\mathcal{B}) + \frac{q_{\mathsf{F}}}{|\mathcal{M}|}\right)\,. \tag{4}$$

Summing up the inequalities yields the claimed bound, concluding the proof of the theorem.

We show (4) by giving an adversary $\mathcal{B}$ against the $\mathsf{n\text{-}IND\text{-}CPA}$ security of PKE. Adversary $\mathcal{B}$ from the $\mathsf{n\text{-}IND\text{-}CPA}$ challenger n public-keys and a left-or-right challenge encryption oracle. The

reduction samples $\vec{K}^* \xleftarrow{\$} (\{0,1\}^k)^n$ and message vectors $\vec{m}_0 \xleftarrow{\$} \mathcal{M}^n$ and $\vec{m}_1 \xleftarrow{\$} \mathcal{M}^n$ and sends them to the LoR challenge oracle to obtain $\vec{c}^* \leftarrow \mathsf{Enc}(\vec{pk}, \vec{m}_b)$. The reduction calls $\mathcal{A}$ on $(\vec{pk}, \vec{c}^*, \vec{K}^*)$ simulating its view in G_4. On a query $\mathsf{H}_j(m_{j,b'})$ the reduction sends the n-IND-CPA challenge the bit b', if no such query happens, the reduction just samples a uniform bit b' instead.

Since $\vec{m}_{1-b}$ is information-theoretically hidden from $\mathcal{A}$ the probability that it queries either $\mathsf{G}_j(m_{j,1-b})$ or $\mathsf{H}_j(m_{j,1-b})$ is bounded by $q_\mathsf{F}/|\mathcal{M}|$. Assume that this is not the case. We have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{n\text{-}IND\text{-}CPA}}_{\mathsf{PKE}}(\mathcal{B}) + \frac{q_\mathsf{F}}{|\mathcal{M}|} &\geq \left|\Pr[b = b'] - \frac{1}{2}\right| \\ &= \left|\Pr[\text{QUERY}] + \frac{1}{2}\Pr[\neg\text{QUERY}] - \frac{1}{2}\right| \\ &= \frac{1}{2}\Pr[\text{QUERY}]\,,\end{aligned}$$

which proves (4). □

5 Proof of Theorem 3.2

Before we give the proof of Theorem 3.2, we recall some standard quantum notation for the reader's convenience. The presentation of notation and already known results closely follows the presentations given in [HKSU20] and [Höv21, Section 1.3]. Readers that are already familiar with standard quantum notation and helper theorems in the context of the FO transformation can skip to Lemma 5.5.

Qubits. For simplicity, we will treat a *qubit* as a vector $|\varphi\rangle \in \mathbb{C}^2$, i.e., a linear combination $|\varphi\rangle = \alpha \cdot |0\rangle + \beta \cdot |1\rangle$ of the two *basis states* (vectors) $|0\rangle$ and $|1\rangle$ with the additional requirement to the probability amplitudes $\alpha, \beta \in \mathbb{C}$ that $|\alpha|^2 + |\beta|^2 = 1$. The basis $\{|0\rangle, |1\rangle\}$ is called *standard orthonormal computational basis*. The qubit $|\varphi\rangle$ is said to be *in superposition*. Classical bits can be interpreted as quantum bits via the mapping $(b \mapsto 1 \cdot |b\rangle + 0 \cdot |1-b\rangle)$.

Quantum Registers. We will treat a quantum register as a collection of multiple qubits, i.e. a linear combination $|\varphi\rangle := \sum_{x\in\{0,1\}^n} \alpha_x \cdot |x\rangle$, where $\alpha_x \in \mathbb{C}$, with the additional restriction that $\sum_{x\in\{0,1\}^n} |\alpha_x|^2 = 1$. As in the one-dimensional case, we call the basis $\{|x\rangle\}_{x\in\{0,1\}^n}$ the *standard orthonormal computational basis*. We say that $|\varphi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x \cdot |x\rangle$ *contains the classical query* x if $\alpha_x \neq 0$.

Measurements. Qubits can be measured with respect to a basis. In this paper, we will only consider measurements in the standard orthonormal computational basis, and denote this measurement by Measure$(\cdot)$, where the outcome of Measure$(|\varphi\rangle)$ for a single qubit $|\varphi\rangle = \alpha \cdot |0\rangle + \beta \cdot |1\rangle$ will be 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$, and the outcome of measuring a qubit register $|\varphi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x \cdot |x\rangle$ will be x with probability $|\alpha_x|^2$. Note that the amplitudes *collapse* during a measurement, this means that by measuring $\alpha \cdot |0\rangle + \beta \cdot |1\rangle$, α and β are switched to one of the combinations in $\{\pm(1,0), \pm(0,1)\}$. Likewise, in the n-dimensional case, all amplitudes are switched to 0 except for the one that belongs to the measurement outcome and which will be switched to 1.

Quantum oracles and quantum Adversaries. Following [BDF+11, BBC+98], we view a quantum oracle $|\mathsf{O}\rangle$ as a mapping

$$|x\rangle|y\rangle \mapsto |x\rangle|y \oplus \mathsf{O}(x)\rangle \ ,$$

where $\mathsf{O} : \{0,1\}^n \to \{0,1\}^m$, and model quantum adversaries A with access to O by a sequence U_1, $|\mathsf{O}\rangle$, U_2, $\cdots$, $|\mathsf{O}\rangle$, U_N of unitary transformations. We write $\mathsf{A}^{|\mathsf{O}\rangle}$ to indicate that the oracles are quantum-accessible (contrary to oracles which can only process classical bits).

Quantum random oracle model. We consider security games in the quantum random oracle model (QROM) as their counterparts in the classical random oracle model, with the difference that we consider quantum adversaries that are given **quantum** access to the (offline) random oracles involved, and **classical** access to all other (online) oracles.

We will now recall some QROM theorems that we will use during our proof of Theorem 3.2.

Lemma 5.1 (Simulating Quantum Random Oracles [Zha12]). *A quantum random oracle can be perfectly simulated by a 2q-wise independent function for any adversary making at most q (quantum) queries.*

Lemma 5.2 (Generic Distinguishing Problem with Bounded Probabilities [HKSU20]). *Let X be a finite set, and let $\lambda \in [0,1]$. For any (unbounded, quantum) algorithm $\mathcal{A}$ issuing at most q quantum queries to F,*

$$|\Pr[\mathsf{GDPB}^{\mathcal{A}}_{\lambda,0} \Rightarrow 1] - \Pr[\mathsf{GDPB}^{\mathcal{A}}_{\lambda,1} \Rightarrow 1]| \leq 8 \cdot \lambda \cdot (q+1)^2 \quad , \tag{5}$$

where games $\mathsf{GDPB}_{\lambda,b}$ (for bit $b \in \{0,1\}$) are defined in Figure 7.

GAME $\mathsf{GDPB}_{\lambda,b}$

$(\lambda_x)_{x\in X} \leftarrow \mathcal{A}_1$
if $\exists x \in X$ s.t. $\lambda_x > \lambda$
  return 0
if $b = 1$
  for all $x \in X$
    $F(x) \leftarrow B_{\lambda_x}$
else
  $F := 0$
$b' \leftarrow \mathcal{A}_2^{|\mathsf{F}\rangle}$
return b'

Figure 7: Generic distinguishing games $\mathsf{GDPB}_{\lambda,b}$ with bounded maximal Bernoulli parameter $\lambda \in [0,1]$.

Oneway to Hiding with semi-classical oracles. In [AHU19], Ambainis et al. defined semi-classical oracles that return a state that was measured with respect to one of the input registers. To any subset $S \subset X$, one can associate the following "semi-classical" oracle $\mathsf{O}^{\mathsf{SC}}_S$: Intuitively, $\mathsf{O}^{\mathsf{SC}}_S$ collapses states taken from $\mathcal{H}_{X\times Y}$ to a state that contains only elements of either S or $X \setminus S$. To be more precise, $\mathsf{O}^{\mathsf{SC}}_S$ takes as input a quantum state $|\psi, 0\rangle$ such that $|\psi\rangle \in \mathcal{H}_X \otimes \mathcal{H}_Y$. $\mathsf{O}^{\mathsf{SC}}_S$ first measures the X-register with respect to the projectors $M_1 := \sum_{x\in S} |x\rangle\langle x|$ and $M_0 := \sum_{x\notin S} |x\rangle\langle x|$, and then initialises the last register to $|b\rangle$ for the measured bit b. Consequently, $|\psi, 0\rangle$ collapses to either a state $|\psi', 1\rangle$ such that the X-register of $|\psi'\rangle$ only contains elements of S, or a state $|\psi', 0\rangle$ such that the X-register of $|\psi'\rangle$ only contains elements of $X \setminus S$.

To a quantum-accessible oracle O and a subset $S \subset X$, one can furthermore associate oracle $\mathsf{O} \setminus \mathsf{S}$ which first queries $\mathsf{O}^{\mathsf{SC}}_S$ and then O. Let FIND denote the event that $\mathsf{O}^{\mathsf{SC}}_S$ ever returns a state $|\psi', 1\rangle$. Unless FIND occurs, the outcome of $\mathsf{O} \setminus \mathsf{S}$ is independent of the values $\mathsf{O}(x)$ for all $x \in S$, which is why $\mathsf{O} \setminus \mathsf{S}$ is also called a "punctured" oracle.

We will now restate several "semi-classical one-way to hiding" theorems from [AHU19]. While [AHU19] consider adversaries that might execute parallel oracle invocations, and therefore differentiate between query depth d and number of queries q, we use the upper bound $q \geq d$ for the

sake of simplicity. Theorem 5.3 is a simplification of [AHU19, Thm. 1: "Semi-classical O2H"], and Equation (7) (Equation (8)) of Theorem 5.4 is a simplification of [AHU19, Thm. 2: "Search in semi-classical oracle"] ([AHU19, Cor. 1]).

Theorem 5.3 *Let $S \subset X$ be random. Let $\mathsf{O}_1, \mathsf{O}_2 \in Y^X$ be random functions such that $\mathsf{O}_1(x) = \mathsf{O}_2(x)$ for all $x \in X \setminus S$, and let z be a random bitstring. (S, O_1, O_2, and z may have an arbitrary joint distribution.) For $i \in \{1, 2\}$, let*

$$p_i := \Pr[1 \leftarrow \mathsf{A}^{|\mathsf{O}_i\rangle}(z)] \ ,$$

and let

$$p_{\text{FIND}} := \Pr[b \leftarrow \mathsf{A}^{|\mathsf{O}_1 \setminus \mathsf{S}\rangle}(z) : \text{FIND}] \ .$$

For all quantum algorithms A with binary output, issuing at most q queries, we have that

$$|p_1 - p_2| \leq 2 \cdot \sqrt{(q+1) \cdot p_{\text{FIND}}} \ . \tag{6}$$

Theorem 5.4 *Let $S \subset X$ be random, let O be a random function, and let z be a random bitstring. (S, O, and z may have an arbitrary joint distribution.) Let*

$$p_{\text{FIND}} := \Pr[b \leftarrow \mathsf{A}^{|\mathsf{O} \setminus \mathsf{S}\rangle}(z) : \text{FIND}] \ .$$

Then, for all quantum algorithms A with binary output issuing at most q queries, we have that

$$p_{\text{FIND}} \leq 4q \cdot \Pr[x \leftarrow \mathsf{B}(z) : x \in S] \ , \tag{7}$$

where B is the algorithm that, on input z, chooses $i \xleftarrow{\$} \{1, \cdots, q\}$, runs $\mathsf{A}^{|\mathsf{O}\rangle}$ until (just before) the i-th query, measures its query input register in the computational basis and outputs the measurement outcome.

If $S := \{x_1, \ldots, x_n\}$ for $x_1, \ldots, x_n \xleftarrow{\$} X$, and S and z are independent, we have that

$$p_{\text{FIND}} \leq \frac{4qn}{|X|} \ . \tag{8}$$

We will now prove an additional helper lemma.

Lemma 5.5 (QROM Multi-User PRF). *For $i, j \in [n]$, let $p_i, p_j \in \{0,1\}^\gamma$ arbitrarily subject to $p_i \neq p_j$, when $i \neq j$. Define $\vec{p} := (p_1, \ldots, p_n)$. Let $\mathsf{H}, \mathsf{H}_1, \ldots, \mathsf{H}_n$ be independent random oracles with $\mathsf{H}\colon \{0,1\}^\gamma \times \{0,1\}^\lambda \times \mathcal{X} \to \mathcal{Y}'$ and $\mathsf{H}_i : \mathcal{X} \to \mathcal{Y}'$, then for all quantum algorithms $\mathcal{A}$ issuing at most q quantum queries to H and arbitrarily many queries to H_i with $i \in [n]$, we have*

$$\left| \Pr\left[\mathcal{A}^{|\mathsf{H}\rangle, \mathsf{H}(p_1, s_1, \cdot), \ldots, \mathsf{H}(p_n, s_n, \cdot)}(\vec{p}) = 1\right] - \Pr\left[\mathcal{A}^{|\mathsf{H}\rangle, \mathsf{H}_1, \ldots, \mathsf{H}_n}(\vec{p}) = 1\right] \right| \leq 4q\sqrt{\frac{n}{2^\lambda}}$$

where the probabilities are taken over $\mathsf{H}, \mathsf{H}_1, \ldots, \mathsf{H}_n$ and $s_1, \ldots s_n \xleftarrow{\$} \{0,1\}^\lambda$, and the internal randomness of $\mathcal{A}$.

Proof. The proof follows from combining [AHU19, Theorem 1] with [AHU19, Corollary 1], the overall proof idea is very similar to the one of [BHH+19, Corollary 1]. The adversary's goal is to distinguish quantum access to H and additional classical access to $(\mathsf{H}(p_1, s_1, \cdot), ..., \mathsf{H}(p_n, s_n, \cdot))$ from quantum access to H and additional classical access to a collection $(\mathsf{H}_1, ..., \mathsf{H}_N)$ of independent random oracles. This is the same as distinguishing $(\mathsf{H}, \mathsf{H}^{(p_1, s_1, \cdot) \mapsto \mathsf{H}_1(\cdot)}, \ldots, \mathsf{H}^{(p_n, s_n, \cdot) \mapsto \mathsf{H}_n(\cdot)})$ from $(\mathsf{H}, \mathsf{H}_1, ..., \mathsf{H}_n)$, and the set of differences between these oracles is $S := \{(p_j, s_j)\}_{j \in [n]} \times \mathcal{X}$. According to [AHU19, Theorem 1], the distinguishing advantage is at most $2\sqrt{q \cdot p_{\text{FIND}}}$, and since S is independent of the collection $(\mathsf{H}_1, ..., \mathsf{H}_n)$, we can apply [AHU19, Corollary 1] to obtain that $p_{\text{FIND}} \leq 4q \cdot \frac{n}{2^\lambda}$.

□

With these results at hand, we can finally proceed to the proof of Theorem 3.2. Its overall structure is very similar to recent QROM proofs of IND-CCA security: We first change the game such that the decapsulation oracle can be simulated without knowledge of the collection of secret keys, which is achieved as usual by plugging encryption into the random oracle, and then apply one-way to hiding to argue key indistinguishability. For the sake of completeness, we also show how to modify the proof such that it accommodates schemes that are only n-OW-CPA secure, at the price of an additional loss of $\sqrt{q}$.

We want to add a remark: As usual, the simulation of the decapsulation oracle deals with potential decryption failure by replacing random oracle G with a modification G' such that $\mathsf{G}'(\mathsf{ID}(pk_j),\cdot)$ only samples "good" randomness with respect to pk_j. (The notion of "good" randomness will be explained during the proof.) Remark that this step might turn out to be difficult to prove without the hashing of public-key identifiers, as we would then have to modify G such that it only samples "good" randomness with respect to *all* public keys at once, which would diminish the potential range further with each user.

Proof.

GAMES G_0 - G_6

11 **for** $j \in [n]$
12 $\quad(pk_j, sk_j) \xleftarrow{\$} \mathsf{Gen}$
13 $\quad m_j^* \xleftarrow{\$} \mathcal{M}$
14 $\quad c_j^* \leftarrow \mathsf{Enc}(pk_j, m_j^*; \mathsf{G}(\mathsf{ID}(pk_j), m_j^*))$
15 $\quad K_{j0}^* := \mathsf{H}(\mathsf{ID}(pk_j), m_j^*)$
16 $\quad K_{j1}^* \xleftarrow{\$} \{0,1\}^n$
17 $\vec{pk} \leftarrow (pk_1, \ldots, pk_n)$
18 $\exists i, j \in [n]$ s.t. $\mathsf{ID}(pk_i) = \mathsf{ID}(pk_j)$ // G_1-G_6
19 $\quad$ COLL := **true** // G_1-G_6
20 $\quad$ **abort** // G_1-G_6
21 $\vec{c^*} \leftarrow (c_1^*, \ldots, c_n^*)$
22 $\vec{K_0^*} \leftarrow (K_{00}, \ldots, K_{n0})$
23 $\vec{K_1^*} \leftarrow (K_{01}, \ldots, K_{n1})$
24 $b \xleftarrow{\$} \{0,1\}$
25 $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{Decaps}_m^{\not\perp}, |\mathsf{G}\rangle, |\mathsf{H}\rangle}(\vec{pk}, \vec{c^*}, \vec{K_b^*})$
26 **return** $b' = b$

$\mathsf{G}(\mathsf{id}, m)$
27 **if** $\exists j \in [n] : \mathsf{id} = \mathsf{ID}(pk_j)$
28 $\quad$ **return** $\mathsf{G}_j(m)$ // G_1-G_2,G_6
29 $\quad$ **return** $\mathsf{G}'_j(m)$ // G_3-G_5
30 **return** $\mathsf{G}_0(\mathsf{id}, m)$

$\mathsf{H}(\mathsf{id}, m)$
31 **if** $\exists j \in [n] : \mathsf{id} = \mathsf{ID}(pk_j)$ // G_1-G_6
32 $\quad$ **return** $\mathsf{H}_j(m)$ // G_1-G_6
33 **return** $\mathsf{H}_0(\mathsf{id}, m)$

$\mathsf{H}_j(m)$ // Internal Random Oracle
34 **return** $\mathsf{H}_j^1(m)$ // G_0-G_3
35 **return** $\mathsf{H}_j^2(\mathsf{Enc}(pk_j, m; \mathsf{G}_j(m)))$ // G_4-G_6

$\mathsf{Decaps}_m^{\not\perp}(j, c \neq c_j^*)$ // G_5-G_6
36 **return** $\mathsf{H}_j^2(c)$

$\mathsf{Decaps}_m^{\not\perp}(j, c \neq c_j^*)$ // G_0-G_4
37 $m' := \mathsf{Dec}(sk_j, c)$
38 **if** $\mathsf{Enc}(pk_j, m'; \mathsf{G}(\mathsf{ID}(pk_j), m')) = c$
39 $\quad K := \mathsf{H}(\mathsf{ID}(pk_j), m')$
40 **else**
41 $\quad K := \mathsf{H}(\mathsf{ID}(pk_j), s_j, c)$
42 $\quad K := \mathsf{H}_j^3(c)$ // G_2-G_4
43 **return** K

Figure 8: Games G_0 - G_6 for the proof of Theorem 3.2. $(\mathsf{H}_j^i)_{i\in[1,3],j\in[0,n]}$ and $(\mathsf{G}_j)_{j\in[0,n]}$ are internal random oracles not directly accesible to the adversary. $|\mathsf{G}\rangle$ and $|\mathsf{H}\rangle$ denotes that the adverary $\mathcal{A}$ has quantum access to the random oracles. G'_j denotes the modification of random oracle G_j which only samples "good" randomness (a more detailed explanation is given during the proof).

GAME G_0. This is the original n-IND-CCA game. We have

$$\left|\Pr[G_0 \Rightarrow 1] - \frac{1}{2}\right| = \mathsf{Adv}^{\mathsf{n\text{-}IND\text{-}CCA}}_{\mathsf{KEM}}(\mathcal{A})\,.$$

GAME G_1. Game G_1 differs from G_0 in that the game aborts on a public-key identifier collision. We can now identify every public-key identifier $\mathsf{ID}(pk_j)$ with its index j (as in the ROM proof). This allows us to simulate the random oracles, when called on such an identifier, via

$$\mathsf{H}(\mathsf{ID}(pk_j), m) := \mathsf{H}_j(m)$$

and

$$\mathsf{G}(\mathsf{ID}(pk_j), m) := \mathsf{G}_j(m)\,,$$

where H_j and G_j are fresh independent oracles.

Since the latter is only a conceptual change, we have

$$\left|\Pr[G_0 \Rightarrow 1] - \Pr[G_1 \Rightarrow 1]\right| \leq \Pr[\mathrm{COLL}] \leq \frac{n^2}{2^l}\,.$$

GAME G_2. This game differs from G_1 in that we substitute $\mathsf{H}(\mathsf{ID}(pk_j), s_j, c)$ with $\mathsf{H}^3_j(c)$, where H^3_j is an internal independent random oracle for every $j \in [n]$. With a straightforward reduction, we can apply Lemma 5.5 to obtain

$$\left|\Pr[G_1 \Rightarrow 1] - \Pr[G_2 \Rightarrow 1]\right| \leq 2q_{\mathsf{H}}\sqrt{\frac{n}{2^\lambda}}\,.$$

GAME G_3. This game switches the random oracle G to an oracle which samples only "good" randomness, meaning that no decryption failure of proper encryptions can possibly occur anymore. For fixed $(pk, sk) \in \mathsf{supp}\,(\mathsf{Gen})$ and $m \in \mathcal{M}$, let

$$\mathcal{R}_{\mathrm{bad}}(pk, sk, m) := \{r \in \mathcal{R} \colon \mathsf{Dec}(sk, \mathsf{Enc}(pk, m; r)) \neq m\}$$

and

$$\delta(pk, sk, m) := \frac{|\mathcal{R}_{\mathrm{bad}}(pk, sk, m)|}{|\mathcal{R}|}\,.$$

The modified oracle G can now be defined as follows: We will still let G coincide with G_0 anywhere but on $(\mathsf{ID}(pk_j))_j \times \mathcal{M}$. Whereas for any index j, however, $\mathsf{G}(\mathsf{ID}(pk_j), \cdot)$ was defined until this game by as an oracle G_j which has range $\mathcal{R}$, we now replace each G_j with a random oracle G'_j that has range $\mathcal{R} \setminus \mathcal{R}_{\mathrm{bad}}(pk_j, sk_j, m))$ instead.

We will now argue that distinguishing game G_2 from G_3 can be reduced to distinguishing the GDPB games (see Lemma 5.2). Consider the quantum distinguisher $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ given in Figure 9, running either in game $\mathsf{GDPB}_{\lambda,0}$ or in game $\mathsf{GDPB}_{\lambda,1}$, where $\lambda := \delta(n)$. $\mathcal{B}$ perfectly simulates game G_2 if run in game $\mathsf{GDPB}_{\lambda,1}$, and game G_3 if run in $\mathsf{GDPB}_{\lambda,0}$.

Since for any $i \in [n]$ and $m \in \mathcal{M}$, we have that $\delta(pk_i, sk_i, m) \leq \delta(\vec{pk}, \vec{sk})$, applying Lemma 5.2 yields

$$\left|\Pr[G_2 \Rightarrow 1] - \Pr[G_3 \Rightarrow 1]\right| \leq 8(q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{Decaps}} + 1)^2\delta(n)\,.$$

GAME G_4. In this game, we modify the random oracle $\mathsf{H}_j(m)$ such that it returns $\mathsf{H}^2_j(\mathsf{Enc}(pk_j, m; \mathsf{G}_j(m)))$ instead of $\mathsf{H}^1_j(m)$. Here, H^1_j and H^2_j are independent internal random oracles ($j \in [n]$). Since we use the modified G that only samples "good" randomness, the mapping $\mathsf{Enc}(pk_j, \cdot; \mathsf{G}_j(\cdot))$ is injective for

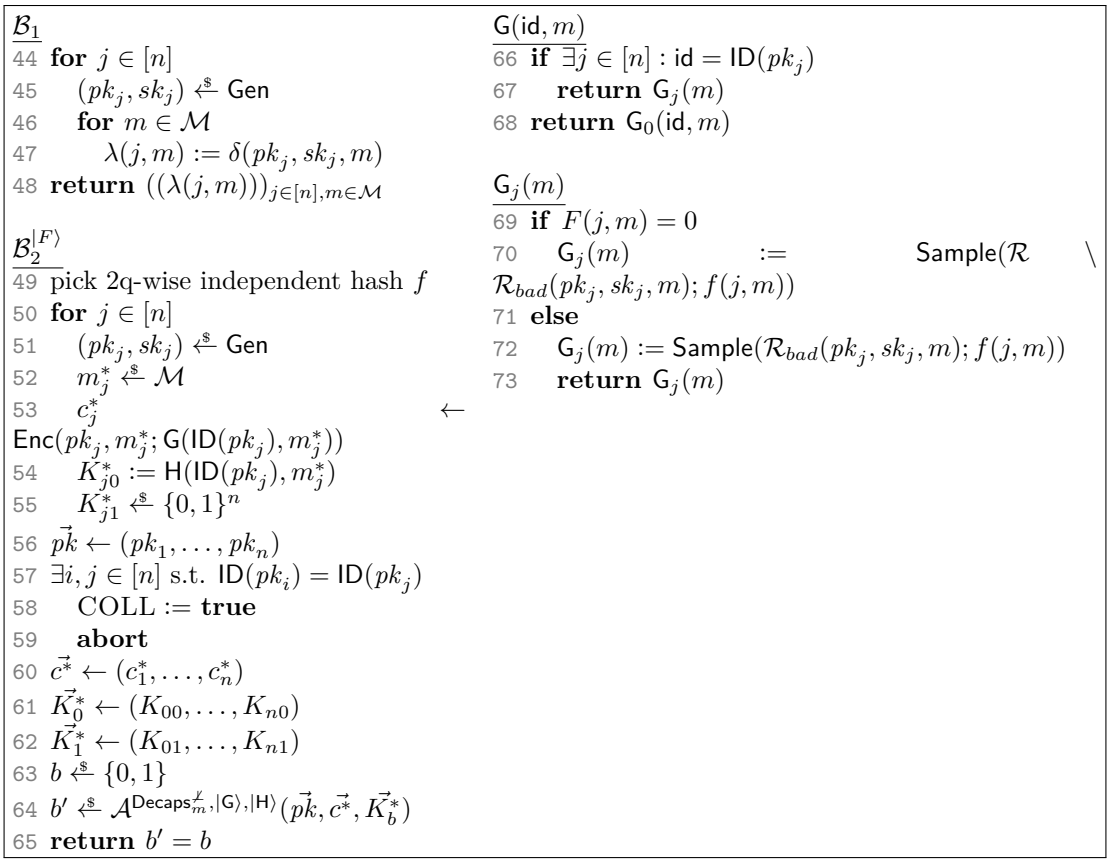

Figure 9: Distinguisher $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ for the generic distinguishing problem with bounded probabilities. Oracles Decaps and H are simulated as in G_2 (or equivalently G_3) of Figure 8. Variable q represents the number of all explicit and implicit queries to G.

all $j \in [n]$. Furthermore, since $\mathsf{ID}(pk_i) \neq \mathsf{ID}(pk_j)$ for all $i \neq j$, we have that the function h, defined as

$$h(\mathsf{id}, m) := \begin{cases} (\mathsf{id}, \mathsf{Enc}(pk_j, m; \mathsf{G}_j(m))) \text{ if exists } j \in [n] \text{ s.t. } \mathsf{id} = \mathsf{ID}(pk_j) \\ (\mathsf{id}, m) \text{ else} \end{cases} , \tag{9}$$

is injective.

We can now rewrite H in Game G_4 as

$$\mathsf{H} = \mathsf{H}^2 \circ h \,,$$

where we define the internal random oracle H^2 as

$$\mathsf{H}^2(\mathsf{id}, m) := \begin{cases} (\mathsf{H}_j^2(m) \text{ if exists } j \in [n] \text{ s.t. } \mathsf{id} = \mathsf{ID}(pk_j) \\ (\mathsf{H}_0(\mathsf{id}, m) \text{ else} \end{cases} .$$

After these changes, $\mathsf{H} = \mathsf{H}^2 \circ h$ for an internal random oracle H^2 and an injective function h, therefore H is still a random oracle in G_4 and hence perfectly indistinguishable from the one of G_3.

$$\Pr[G_3 \Rightarrow 1] = \Pr[G_4 \Rightarrow 1] \ .$$

GAME G_5. In this game, we start to simulate the decapsulation oracle without knowledge of the secret keys. We modify $\mathsf{Decaps}(j, c)$ to return $\mathsf{H}_j^2(c)$. Let $m' := \mathsf{Dec}(sk_j, c)$ and consider the following two cases:

- Case 1: The ciphertext c lies in the image of encryption, that is $\mathsf{Enc}(pk_j, m'; \mathsf{G}_j(m')) = c$. In this case we have

$$\begin{aligned}\mathsf{H}_j^2(c) &= \mathsf{H}_j^2(\mathsf{Enc}(pk_j, m'; \mathsf{G}_j(m'))) \\ &= (\mathsf{H}^2 \circ h)(\mathsf{ID}(pk_j), m') \\ &= \mathsf{H}(\mathsf{ID}(pk_j), m') \\ &= \mathsf{Decaps}(j, c)\,.\end{aligned}$$

  Therefore the oracles in G_4 and G_5 are identical.

- Case 2: The ciphertext c lies not in the image of encryption, that is $\mathsf{Enc}(pk_j, m'; \mathsf{G}_j(m')) \neq c$. Here the adversary gets to see $H_j^3(c)$ in G_4 and $H_j^2(c)$ in G_5 for internal random oracles H_j^3 and H_j^2. In both games the adversary gets access to H^2 indirectly through $H(\mathsf{ID}(pk_j), m)$ forcing m through the encryption algorithm. Therefore, the adversary can not access H^2 for ciphertexts lying outside the image space of encryption. Therefore no adversary can distinguish between $H^3(\mathsf{ID}(pk_j), c)$ and $H^2(\mathsf{ID}(pk_j), c)$.

We just have shown

$$\Pr[G_4] = \Pr[G_5]\,.$$

GAME G_6. In game G_6, we switch back to using the original G. With essentially the same argument as in the game-hop from G_2 to G_3, we have that

$$\left|\Pr[G_5 \Rightarrow 1] - \Pr[G_6 \Rightarrow 1]\right| \leq 8(q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{Decaps}} + 1)^2\delta(n)\,.$$

What we have achieved so far is that the decapsulation oracle is simulatable by a reduction that does not know the secret keys, we can now apply the one-way to hiding theorems in order to reduce key indistinguishability to an attacker against the underlying encryption scheme.

GAME G_7. In this game, we decouple the oracle values on the challenge plaintexts from the values that are used to compute the challenge ciphertexts and the challenge keys, see Figure 10. In game G_7, the adversary's view is independent of b and

$$\Pr[G_7 \Rightarrow 1] = \frac{1}{2}\,.$$

We can now first apply Theorem 5.3 in order to argue that

$$\left|\Pr[G_6 \Rightarrow 1] - \Pr[G_7 \Rightarrow 1]\right| \leq 2\sqrt{(q+1) \cdot \Pr[\mathrm{FIND}]},$$

where $\Pr[\mathrm{FIND}]$ denotes the probability that one of the punctured oracles $G \setminus S$ or $H \setminus S$ ever collapses the input register of one of the random oracle queries to an element of S, i.e., to an element of the form $(\mathsf{ID}(pk_j), m_j^*)$, when used to replace oracles G and H in game G_6.

It remains to upper bound $\Pr[\mathrm{FIND}]$, which we can do either with a reduction to n-OW-CPA or with a reduction to n-IND-CPA. For the sake of completeness, we do both. Since a n-OW-CPA adversary does not know the challenge plaintexts, it cannot simulate the punctured oracle on its

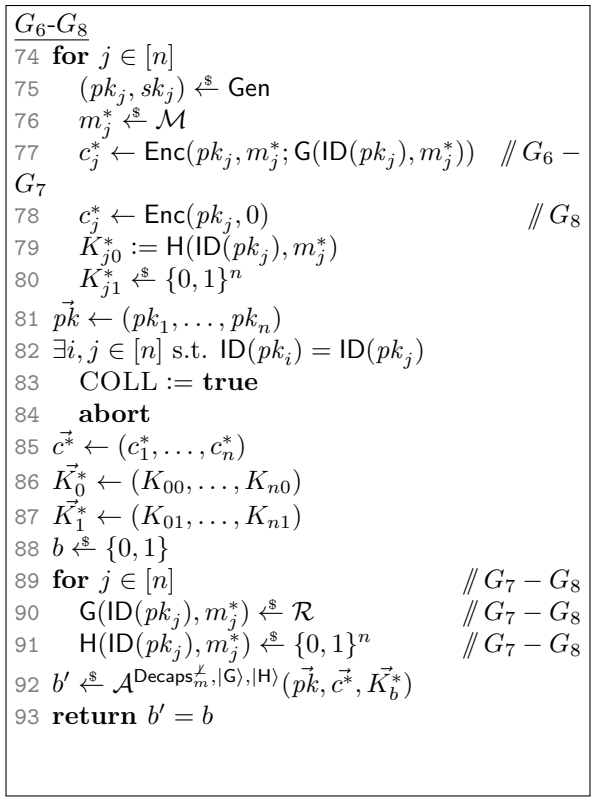

Figure 10: Games G_6 - G_8. Apart from the reprogramming in lines 92-94, all oracles remain defined as in game G_6 (see Figure 8).

own. We therefore have to resort to Equation (7) of Theorem 5.4, thereby losing a factor linear in q, the number of random oracle queries:

$$p_{\text{FIND}} \leq 4q \cdot \mathsf{Adv}^{\mathsf{n\text{-}OW\text{-}CPA}}_{\mathsf{PKE}}(\mathcal{B}_{\mathsf{OW}}) \ , \tag{10}$$

where $\mathcal{B}_{\mathsf{OW}}$ is the adversary that chooses $i \xleftarrow{\$} \{1, \cdots, q\}$, simulates game G_6 to A until (just before) its i-th query, measures its query input register in the computational basis to obtain a tuple $(\mathsf{ID}(pk_j), m_j^*)$, and outputs (j, m_j^*).

Contrarily, a n-IND-CPA reduction knows the challenge plaintexts, it can hence perfectly simulate the puncturing. Our respective reduction $\mathcal{B}_{\mathsf{IND}}$ queries its challenge oracle CHAL on the vector $\vec{m}_0 := (m_j^*)_{j \in [n]}$, where the m_j^* are defined as in game G_6, and the constant zero vector $\vec{m}_1 := (0)_{j \in [n]}$. It uses the response of CHAL to simulate either game G_6 with punctured oracles (if the n-IND-CPA bit is 0) or game G_8 with punctured oracles (if the n-IND-CPA bit is 1), we hence have

$$p_{\text{FIND}} \leq \Pr[\text{FIND in } G_8] + \mathsf{Adv}^{\mathsf{n\text{-}IND\text{-}CPA}}_{\mathsf{PKE}}(\mathcal{B}_{\mathsf{IND}}) \ . \tag{11}$$

Since in game G_8, the set S is independent of A's view, we can now apply Equation (8) of Theorem 5.4 to argue that

$$\Pr[\text{FIND in } G_8] \leq \frac{4qn}{|X|} \ . \tag{12}$$

Adding the bounds yields the claimed bound in the theorem, concluding our proof. □

References

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Heidelberg, August 2009. (Cited on page 4.)

[AG11] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *ICALP 2011, Part I*, volume 6755 of *LNCS*, pages 403–415. Springer, Heidelberg, July 2011. (Cited on page 4.)

[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. (Cited on page 13, 14.)

[ALS20] Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 470–499. Springer, 2020. (Cited on page 6.)

[BBC+98] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. In *39th FOCS*, pages 352–361. IEEE Computer Society Press, November 1998. (Cited on page 12.)

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, Heidelberg, May 2000. (Cited on page 2, 3, 4.)

[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. (Cited on page 12.)

[BHH+19] Nina Bindel, Mike Hamburg, Kathrin Hövelmanns, Andreas Hülsing, and Edoardo Persichetti. Tighter proofs of CCA security in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 61–90. Springer, Heidelberg, December 2019. (Cited on page 3, 6, 14.)

[CLS16] Hao Chen, Kristin E. Lauter, and Katherine E. Stange. Security considerations for galois non-dual RLWE families. In Roberto Avanzi and Howard M. Heys, editors, *SAC 2016*, volume 10532 of *LNCS*, pages 443–462. Springer, Heidelberg, August 2016. (Cited on page 6.)

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Heidelberg, August 1999. (Cited on page 2.)

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013. (Cited on page 2.)

[Hås88] Johan Håstad. Solving simultaneous modular equations of low degree. *SIAM J. Comput.*, 17(2):336–341, 1988. (Cited on page 2.)

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Heidelberg, November 2017. (Cited on page 2, 3, 4, 7, 8.)

[HKSU20] Kathrin Hövelmanns, Eike Kiltz, Sven Schäge, and Dominique Unruh. Generic authenticated key exchange in the quantum random oracle model. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 389–422. Springer, Heidelberg, May 2020. (Cited on page 3, 12, 13.)

[Höv21] Kathrin Hövelmanns. *Generic constructions of quantum-resistant cryptosystems*. doctoralthesis, Ruhr-Universität Bochum, Universitätsbibliothek, 2021. (Cited on page 12.)

[JZM19] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 227–248. Springer, Heidelberg, 2019. (Cited on page 3.)

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Heidelberg, May / June 2010. (Cited on page 4.)

[LS15] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, 75(3):565–599, 2015. (Cited on page 4.)

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005. (Cited on page 4.)

[Zha12] 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. (Cited on page 13.)