CAKE requires programming - On the provable post-quantum security of (O)CAKE

Kathrin Hövelmanns[1], Andreas Hülsing[1,2], Mikhail Kudinov[1] and Silvia Ritsch[1]

[1] Eindhoven University of Technology, The Netherlands
[2] Sandbox AQ, Palo Alto, United States

Abstract. We revisit the post-quantum security of Password-Authenticated Key Exchange (PAKE) from Key Encapsulation Mechanisms (KEM), focusing on (O)CAKE. Despite extensive study, these protocols still lack a security proof against quantum adversaries. We pinpoint exactly why by identifying the precise remaining obstacle. Most game-hopping arguments extend to the quantum setting via standard QROM techniques. The difficulty lies with ideal ciphers: adaptive programming and query recording are well understood classically, but no analogous tools exist for ideal ciphers in quantum superposition. To isolate the issue, we formalize programming for PAKE via a *programming simulator*, which captures adaptive programming on a family of permutations from an arbitrary distribution. We prove this notion is both *necessary* and *sufficient* for a proof.

For necessity, we present a meta-reduction showing that any reduction-based proof for (O)CAKE must use programming in exactly this sense. We exhibit a class of KEMs – including practical schemes such as Kyber/ML-KEM – for which any successful reduction must either program or directly break the KEM challenge. The existence of this class implies that no fully generic KEM-based PAKE proof can avoid programming.

For sufficiency, we show that this is the only remaining barrier: assuming the existence of our simulator and using known QROM tools, we give a complete post-quantum security proof for OCAKE.

Together, our results explain why generic KEM-based PAKEs have so far resisted quantum security proofs and pinpoint the open problem whose resolution would complete the proof.

Keywords: Post-quantum cryptography, password-based authenticated key exchange, PAKE, CAKE, OCAKE, quantum ideal-cipher model, QIC, QROM, meta reduction.

K.H. was supported by an NWO VENI grant (Project No. VI.Veni.222.397). A.H. and M.K. were supported by an NWO VIDI grant (Project No. VI.Vidi.193.066). S.R. is part of the Quantum-Safe Internet (QSI) ITN which received funding from the European Union's Horizon-Europe programme as Marie Sklodowska-Curie Action (PROJECT 101072637 - HORIZON -MSCA-2021-DN-01). The authors would like to thank Manuel Barbosa, Anne Broadbent, Christian Majenz, and Marc Vostermanns, and Yu-Hsuan Huang for helpful discussions.

This work is licensed under a "CC BY 4.0" license.
Date of this document: 2026-08-03.

Contents

1 Introduction **2**

2 Preliminaries **4**

3 Protocols CAKE and OCAKE **10**

4 Definition of Programmability (SIM) **12**

5 Necessity of SIM for OCAKE **13**

6 Security of OCAKE in the QROM assuming SIM **23**

References **31**

A Honest robustness of FO-KEMs without prefix hashing **36**

1 Introduction

Security from Passwords. Passwords remain one of the most common authentication methods because they are easy to use across devices without extra hardware. However, their low entropy makes them easy to guess, and they can be intercepted during transmission. Password-authenticated key exchange (PAKE) avoids interception issues: the password is never sent, but is used locally by the parties to derive a shared session key. Beyond web authentication, PAKE is widely used whenever a session key must be established from a low-entropy shared secret. Prominent applications include WiFi authentication and contactless authentication of identity documents, where the machine-readable part of the document is read and used as a PAKE password (specifically PACE for ICAO passports).

These applications have spurred major recent efforts in PAKE design, including a competition within IETF's Crypto-Forum Research Group (CFRG) to select new PAKEs. The majority of existing designs, however, relies on group-based cryptography and its algebraic structure, making them vulnerable to future quantum attacks.

Post-Quantum PAKE. For short-lived authentication decisions, the threat induced by quantum computers is limited because the quantum attack must occur during the protocol execution. But when PAKE establishes session keys, the protected data is vulnerable to *store-now-decrypt-later* attacks, so keys exchanged today can be broken by *future* quantum adversaries. Urgency also stems from PAKE's use in long-lived identity documents, whose lengthy specification and standardization cycles leave outdated schemes deployed for 15–20 years after transition. Thus, post-quantum secure PAKE is already needed.

Several works have been done on the topic of post-quantum PAKE. In general, these can be divided into two categories: First, there are proposals that build PAKE directly from specific hardness assumptions [DAL+17, SA23, DKBY24, TY19, SH19]. Second, there are generic proposals [BCP+23, PZ23] that can construct PAKE from generic Key Encapsulation Mechanisms (KEMs) that fulfill certain somewhat-standard properties ([Xag22, GMP22, MX23]). The former are mostly based on specific hardness assumptions on lattices. These proposals are usually quite efficient as they can take advantage of the available algebraic structure. The generic proposals do not have access to this structure, which can make them slightly less efficient. However, they have the advantage that they can be instantiated with a wide range of KEM. In short, assumption-specific constructions gain efficiency, while generic constructions gain long-term cryptographic agility.

Given that post-quantum schemes are often considered less mature and that applications often have different requirements, cryptographic agility with respect to choice of KEM'seems

like a strong advantage in practice. For that reason, in this work we study generic constructions. A comprehensive overview of post-quantum PAKE proposals and the models used in their analysis is given in [AHMW25], Table 1.
The main contenders in PAKE from KEM are the recently proposed protocols CAKE & OCAKE [BCP+23]. These are instances of the well-known Encrypted Key Exchange (EKE) blueprint that was also used in PACE, SPEKE, and many more. [LLH24] present a generic construction and provide a security proof that considers quantum adversaries. This proposal is particularly interesting as it does not require the missing proof technique outlined in this work. This is done by leveraging that the underlying lossy public-key encryption has ciphertexts that are statistically close to uniform, avoiding the reduction a computational assumption such as anonymity would require. However, to achieve this property requires very particular choices of parameters, limiting its applicability to currently standardized KEM, such as ML-KEM. Therefore in the following we study the case of (O)CAKE.

The quantum proof gap. It has been shown both in the UC-model [BCP+23] and the game-based setting [PZ23, AHHR24] that CAKE & OCAKE are secure against classical adversaries. However, a proof of security against *quantum* attackers is still missing so far.

It is known since [BDF+11] that we must model adversaries as having *quantum access* to both random oracles and ideal ciphers. In the classical setting, random oracles offer comparably simple, 'standard' arguments, however allowing quantum access can add significant technical overhead. Examples include query recording and (adaptive) reprogramming of oracles. Concerning random oracle arguments, the community has nonetheless made huge progress in finding suitable quantum counterparts. For ideal ciphers, however, the situation is different. Although this area of research is very active [Hos25, CHL+25, ACMT25, CP25, MMW24, Unr23a, Unr21, CMSZ19, HHM+26b], some techniques so far seem out of reach.
2026 update: In the years since the first submission of this document, there have been several developments in the field of quantum-accessible random permutations [ABMS26, MMW24, CHL+25, Car25, MMW25, HHM+26a]. New developments include compressed permutation oracles that allow for query-recording and some new lower-bounds for e.g. the double-sided-zero-search problem as well as reprogramming and resempling lemmas. However to the best of our knowledge no known technique fulfils strong adaptive programming requirement outlined in this work yet.

Our contribution. Our main contribution is a necessary-and-sufficient characterization of what is missing to enable a post-quantum proof. To do so, we

1. give a precise *definition* of the missing piece, an IC-like primitive called a programming simulator

2. prove the *necessity* of such a simulator when proving security of an EKE PAKE from an arbitrary KEM, using a meta-reduction technique, and

3. demonstrate *sufficiency* by giving a proof against quantum adversaries in the Quantum-accessible Random Oracle Model (QROM), assuming the existence of this simulator.

Technical overview. We formalize the notion of programmability as a 'simulator' SIM that provides an encryption and decryption oracle to an adversary in Definition 12. Using this definition of programmability, our main result can then be summarized by two (informal variants of the main) theorems:

Theorem 1 (Necessary: Any generic anonymity reduction must program.)**.** *There cannot exist any reduction $\mathcal{R}$ that reduces* ANO-PCA *to the* BPR *security of* OCAKE *generically for*

any KEM *without implementing a programming simulator* SIM*, unless it breaks anonymity itself.*

Theorem 2 (Sufficient: OCAKE is post-quantum secure, if programming is possible.)**.** *Assume that a programming simulator* SIM *exists. Then* OCAKE *is secure for any* KEM *that is* $(1 - \delta_{\mathsf{KEM}})$*-correct, multi-user-*IND-CPA- *and multi-user-*ANO-PCA*-securein the QROM.*

The formal statements of these theorems can be found in Proposition 1 and Theorem 4, respectively. We conduct our analysis in the game-based BPR model, however we would like to note that since UC-security implies BPR security [CHK+05, Appendix A], the impossiblity shown in BPR carries over to the UC setting.

Feasibility of instantiating the simulation. There has been some progress on simulations for random permutations that can support query-recording [Unr23b, MMW24, ACMT25], but to the best of our knowledge, there is currently none that can fulfill the programmability requirement we isolate. In fact, it may prove impossible to achieve an ideal cipher in the quantum setting. Although we phrase our results in some places with respect to an ideal cipher for ease of exposition, they can be easily generalized to families of bijections, where each bijection is sampled from an independent distribution over the set of bijections.

An alternative approach to ideal ciphers was presented in the NoIC protocol [ABJ25] that builds the password-encryption of the public keys using a two-round Feistel Cipher, in the ROM. Indeed, using this approach, it is possible to achieve the necessary reprogramming and query-recording properties (with a guessing loss square in the number of queries to the underlying hash functions) without the use of an ideal cipher, at least in the classical setting. However, it is an open problem to lift the random-oracle arguments used in that proof to the quantum setting. In particular, unlike classical methods, arguments relying on query direction or query ordering have no obvious analogue.

Organization. After stating necessary preliminaries in Section 2, we begin by stating protocols CAKE and OCAKE in Section 3. We introduce the programmability requirement by defining the simulator SIM in Section 4. We prove necessity of the programmability property in Section 5, using a meta-reduction technique in two parts: Meta-reduction 1 (Lemma 2) shows how non-programming reductions break anonymity (and thus OCAKE). Meta-reduction 2 (Lemma 3) shows that programming reductions implement SIM. The results make use of two auxiliary lemmas (Lemmas 4 and 5) and the fact that (O)CAKE is insecure when KEM is not anonymous (Theorem 3), which we show by giving a concrete attack. Lastly we show sufficiency of this programmability property in Section 6, by lifting OCAKE's security proof (in the Bellare-Pointcheval-Rogaway model) to the quantum random oracle model.

2 Preliminaries

Notation. We denote deterministic output y of an algorithm A on input x by $y := A(x)$. We denote algorithms with access to an oracle O by A^{O}. Unless stated otherwise, we assume all our algorithms to be probabilistic and denote the computation by $y \leftarrow\!\$\, A(x)$. We use the notational convention $\mathbf{Adv}_{\mathsf{i}} := |\Pr[\mathrm{G}_i(\mathcal{A}) \Rightarrow 1]|$. We use $[\![b = b']\!]$ to indicate a boolean return value.

Block Ciphers Existing proofs of security of the OCAKE and CAKE protocol are given in the ideal cipher model [Bla06]. Analogous to the ROM for hash functions, the ideal cipher (IC) is an idealized description of a block cipher.

Definition 1 (Block Cipher (BC))**.** A block cipher of block length n and key length k consists of two algorithms $\mathsf{BC} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ and $\mathsf{BC}^{-1} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ such that for every plaintext $m \in \{0,1\}^n$ and key $k \in \{0,1\}^k$, decryption undoes encryption: $\mathsf{BC}^{-1}(k, \mathsf{BC}(k,m)) = m$.

Definition 2 (Ideal Cipher (IC))**.** An ideal cipher is a collection of random permutations indexed by a key, to which all parties (including the adversary) are given oracle access. I.e., it is a pair $\mathsf{E}, \mathsf{D} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$, such that $\mathsf{D}(k, \mathsf{E}(k,m)) = m$ and $\mathsf{E}(k, \mathsf{D}(k,m)) = m$ for all k, m in $\mathcal{K} \times \mathcal{M}$.

Key Encapsulation Mechanisms (KEM). We start with the functional definition of KEMs and then discuss their security. In the end, we introduce a new variant of robustness that is used for some supplementary proofs.

Definition 3 (Key Encapsulation Mechanisms (KEMs))**.** A KEM is a triple of algorithms $\mathsf{KEM} = (\mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$, together with a public key space $\mathcal{PK}$ and secret key space $\mathcal{SK}$.

- $\mathsf{KGen} \to (pk, sk)$: On empty input probabilistically **return** key pair (pk, sk), where pk also defines a finite key space $\mathcal{K}$ and a ciphertext space $\mathcal{C}$.
- $\mathsf{Encap}\ (pk) \to (c, K)$: On input pk probabilistically **return** a pair $(K, c) \in \mathcal{K} \times \mathcal{C}$. We call c the *encapsulation* of the key K.
- $\mathsf{Decap}\ (sk, c) \to K$: On input sk and ciphertext c deterministically **return** a key $K \in \mathcal{K}$.

Definition 4 ($\delta-$Correctness (average-case))**.** We say that KEM is average-case $(1-\delta)$-correct if

$$\Pr[\mathsf{Decap}(sk, c) = K | (c, K) \leftarrow\!\$\ \mathsf{Encap}(pk)] \geq 1 - \delta,$$

where the probability is taken over $(pk, sk) \leftarrow \mathsf{KGen}()$ and the random coins of Encap.

Definition 5 (γ-spreadness)**.** We say that PKE is γ-spread iff for all key pairs $(pk, sk) \in \mathit{supp}(\mathsf{KGen})$ and all messages $m \in \mathcal{M}$ it holds that

$$\max_{c \in \mathcal{C}} \Pr[\mathsf{PKE.Enc}(pk, m) = c] \leq 2^{-\gamma},$$

where the probability is taken over the internal randomness of $\mathsf{PKE.Enc}$.

Definition 6 (Anonymity for KEM)**.** Let KEM be a key encapsulation mechanism with public-key space $\mathcal{PK}$ and key space $\mathcal{K}$. We define the $\mathsf{ANO\text{-}PCA}$ (Anonymity under Plaintext-Checking Attack) game as depicted below, relative to challenge bit b, and the *advantage function of an adversary $\mathcal{A}$ against* KEM as

$$\mathbf{Adv}^{\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{A}) := |\Pr[\mathsf{ANO\text{-}PCA}^0(\mathcal{A}) \Rightarrow 0] - \Pr[\mathsf{ANO\text{-}PCA}^1(\mathcal{A}) \Rightarrow 0]|$$

Game $\mathsf{ANO\text{-}PCA}^b(\mathcal{A})$	$\underline{1\text{-}\mathsf{PCO}(c, K)}$
01 $(pk_0, sk_0) \leftarrow\!\$\ \mathsf{KGen}$	06 if $c \neq c^*$
02 $(pk_1, sk_1) \leftarrow\!\$\ \mathsf{KGen}$	07 $\quad K' \leftarrow \mathsf{Decap}(sk_0, c)$
03 $(c^*, K^*) \leftarrow \mathsf{Encap}(pk_b)$	08 $\quad$ **return** $[\![K = K']\!]$
04 $b' \leftarrow \mathcal{A}^{1\text{-}\mathsf{PCO}}(pk_0, pk_1, c^*)$	09 else **return** $\perp$
05 **return** $[\![b = b']\!]$	

Definition 7 (Multi-user security notions for KEM)**.** Let KEM be a key encapsulation mechanism with public-key space $\mathcal{PK}$ and key space $\mathcal{K}$. For integers n and q_C, we define

the $(n, q_C) - \mathsf{IND\text{-}CPA}$ game and the $(n, q_C) - \mathsf{ANO\text{-}PCA}$ game as below, each relative to challenge bit b, and the respective *advantage function of an adversary* $\mathcal{A}$ *against* KEM as

$$\begin{aligned}\mathbf{Adv}^{(n,q_C)-\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{A}) :=& |\Pr[(n,q_C)-\mathsf{ANO\text{-}PCA}^0(\mathcal{A})\Rightarrow 0]-\\ &\Pr[(n,q_C)-\mathsf{ANO\text{-}PCA}^1(\mathcal{A})\Rightarrow 0]|\\ \mathbf{Adv}^{(n,q_C)-\mathsf{IND\text{-}CPA}}_{\mathsf{KEM}}(\mathcal{A}) :=& |\Pr[(n,q_C)-\mathsf{IND\text{-}CPA}^0(\mathcal{A})\Rightarrow 0]-\\ &\Pr[(n,q_C)-\mathsf{IND\text{-}CPA}^1(\mathcal{A})\Rightarrow 0]| \ .\end{aligned}$$

Game $(n, q_C) - \mathsf{IND\text{-}CPA}^b$

01 for $i \in [n]$
02 $\quad (pk_i, sk_i) \leftarrow\!\$\ \mathsf{KGen}$
03 $\quad \vec{pk}\,.\mathtt{append}(pk_i)$
04 $b' \leftarrow \mathcal{A}^{\ \mathtt{Chall}^b_{q_C}(\vec{pk})}$
05 **return** b'

$\mathtt{Chall}^b_{q_C}(j)$ $\quad //q_C$ times per j

06 $(c, K_0) \leftarrow\!\$\ \mathsf{Encap}(pk_j)$
07 $K_1 \xleftarrow{unif}\!\$\ \mathcal{K}$
08 **return** (c, K_b)

Game $(n, q_C) - \mathsf{ANO\text{-}PCA}^b(\mathcal{A})$

01 for $j \in [n]$
02 $\quad (pk_{0,j}, sk_{0,j}) \leftarrow\!\$\ \mathsf{KGen}$
03 $\quad (pk_{1,j}, sk_{1,j}) \leftarrow\!\$\ \mathsf{KGen}$
04 $\quad \vec{pk}_0\,.\mathtt{append}(pk_{0,j})$
05 $\quad \vec{pk}_1\,.\mathtt{append}(pk_{1,j})$
06 $b' \leftarrow \mathcal{A}^{1\text{-}\mathsf{PCO},\mathtt{Chall}^b_{q_C}}(\vec{pk}_0, \vec{pk}_1)$
07 **return** b'

$\mathtt{Chall}^b_{q_C}(j)$ $\quad //q_C$ times per j

08 $(c_b, K_b) \leftarrow\!\$\ \mathsf{Encap}(pk_{b,j})$
09 $\mathcal{L}^c_j \leftarrow \mathcal{L}^c_j \cup \{c_b\}$
10 **return** (c_b, K_b)

$1\text{-}\mathsf{PCO}(j, c, K)$ $\quad //$ once per j, for $c \notin \mathcal{L}^c_j$

11 $K' \leftarrow \mathsf{Decap}(sk_{0,j}, c)$
12 **return** $[\![K = K']\!]$

Finally, we introduce an extremely weak notion of robustness that we call honest robustness. One may think of it as a form of weak robustness (WROB) [GMP22] where all values are randomly chosen instead of given by an adversary. Consequently, the notion is implied by WROB.

Definition 8 (hROB)**.** For any given key encapsulation mechanism KEM, we define the advantage against honest robustness as

$$\mathbf{Adv}^{\mathsf{hROB}}_{\mathsf{KEM}}\mathcal{A} := \Pr\left[k \neq \perp \;\middle|\; \begin{array}{l}(pk_1, .), (., sk_2) \leftarrow\!\$\ \mathsf{KGen}()\\ (c, .) \leftarrow\!\$\ \mathsf{Encap}(pk_1)\\ k := \mathsf{Decap}(sk_2, c)\end{array}\right]$$

The Fujisaki-Okamoto (FO) transform Many post-quantum KEM proposals leverage the Fujisaki-Okamoto transform [FO99]. The modular variant [HHK17] consists of two transforms: T-transform derandomizes a public-key encryption (PKE) scheme (and adds a reencryption check), while the U-transform turns this derandomized PKE scheme into an IND-CCA secure key encapsulation mechanism. More formally, the **T** transforms an OW-CPA secure public-key encryption scheme into an OW-PCA secure one. To a public-key encryption scheme $\mathsf{PKE} = (\mathsf{PKE.KGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ with message space $\mathcal{M}$ and randomness space $\mathcal{R}$, and random oracle $\mathtt{G}{:}\mathcal{M} \to \mathcal{R}$, we associate $\mathsf{PKE}_1 = \mathbf{T}[\mathsf{PKE}, \mathtt{G}\,]$. The algorithms of $\mathsf{PKE}_1 = (\mathsf{PKE.KGen}, \mathsf{PKE.Enc}_1, \mathsf{PKE.Dec}_1)$ are defined in Fig. 1. Note that Enc1 deterministically computes the ciphertext as $c := \mathsf{PKE.Enc}(pk, m; \mathtt{G}(m))$.

Extractable Compressed Random Oracles Here we introduce an extension of Zhandry's Compressed Random Oracle [Zha19] called extractable Compressed Random Oracle [DFMS22] (eCO). Using $\mathtt{eCO}$ represents a quantum-accessible Random Oracle with an additional mechanism that allows to check if there was a query that satisfies some

$\mathsf{PKE.Enc}_1(pk, m)$	$\mathsf{PKE.Dec}_1(sk, c)$
13 $c := \mathsf{PKE.Enc}(pk, m; \mathtt{G}(m))$	15 $m' := \mathsf{PKE.Dec}(sk, c)$.
14 return c	16 if $m' = \perp$ or $\mathsf{PKE.Enc}(pk, m'; \mathtt{G}(m')) \neq c$ return $\perp$
	17 else return m'

Figure 1: The algorithms of the T-Transform.

function f. This gives a form of observable QROM.
The eCO developed in [DFMS22] has two interfaces: one that mimics the behavior of a random oracle ($\mathtt{eCO.RO}$) and another that allows extraction of information ($\mathtt{eCO.E}$). The $\mathtt{eCO.RO}_f$ interface defined relative to a function f takes a classical value t as input and simulates a quantum measurement that "collapses" the oracle database, allowing it to yield a specific outcome x. After the measurement, the database is in a state where all the values for $\mathtt{eCO.RO}$ collapse to the values y that satisfy the equation $f(x, y) = t$. Whenever the function f used is clear from context we write $\mathtt{eCO.E}$ instead of $\mathtt{eCO.E}_f$. To clarify, consider the classical analogue where the Random Oracle is simulated via lazy-sampling. Then for an extraction query we go through the database and output the first pair (x, y) such that x is the smallest value for which $f(x, y) = t$.
Previously, in [DFMS22, HM24] the eCO was used to simulate Decapsulation queries during the runtime of the adversary. In our case we use it to check if the adversarially generated tags are valid.
Now lets give a more formal treatment of the eCO. We will closely follow the description from [DFMS22, HM24]. The description represents an inefficient version of the eCO which can be made efficient using sparse encoding (c.f., [DFMS22]). The simulator eCO for a random function O: $\{0,1\}^m \to \{0,1\}^n$ is a stateful oracle with a state stored in a quantum register $D = D_{0^m} \dots D_{1^m}$, where for each input value $x \in \{0,1\}^m$, register D_x has $n+1$ qubits used to store superpositions of n-bit output strings y, encoded as $0y$, and an additional symbol $\perp$, encoded as 10^n. We adopt the convention that an operator expecting n input qubits acts on the last n qubits when applied to one of the registers D_x. The compressed oracle has the following three components.

1. The initial state of the oracle, $|\phi\rangle = |\perp\rangle^{2^m}$
2. A quantum query with query input register X and output register Y is answered using the oracle unitary O_{XYD} defined by $O_{XYD}\,|X\rangle_X = |X\rangle_X \otimes \left(F_{D_x}\mathrm{CNOT}^{\otimes n}_{D_x;Y}F_{D_x}\right)$, where $F\,|\perp\rangle = |\phi_0\rangle\,, F\,|\phi_0\rangle = |\perp\rangle$ and $F\,|\psi\rangle = |\psi\rangle$ for all $|\psi\rangle$ such that $\langle\psi|\perp\rangle = \langle\psi|\phi_0\rangle = 0$, with $|\phi_0\rangle = |+\rangle^{\otimes n}$ being the uniform superposition.
3. A recovery algorithm that recovers a standard QRO O: apply $F^{\otimes 2^m}$ to D and measure it to obtain the function table of O.

The formal description of the extraction interface is the following. Given a random oracle $\mathrm{O} : \{0,1\}^m \to \{0,1\}^n$, let $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^\ell$ be a function. We define a family of measurements $(\mathcal{M}^t)_{t\in\{0,1\}^\ell}$. The measurement $\mathcal{M}^t$ has measurement projectors $\{\Sigma^{t,x}\}_{x\in\{0,1\}^m\cup\{\emptyset\}}$.
Let $\Pi^{t,x} = \sum_{\substack{y\in\{0,1\}^n:\\ f(x,y)=t}} |y\rangle\langle y|$. Then, for $x \in \{0,1\}^m$, the projector selects the case where D_x is the first (in lexicographical order) register that contains y such that $f(x, y) = t$, i.e. $\Sigma^{t,x} = \bigotimes_{x'<x} \bar{\Pi}^{t,x'}_{D'_x} \otimes \Pi^{t,x}_{D_x}$, and $\bar{\Pi} = \mathbb{1} - \Pi$. The remaining projector corresponds to the case where no register contains such a y, i.e. $\Sigma^{t,\emptyset} = \bigotimes_{x'\in\{0,1\}^m} \bar{\Pi}^{t,x'}_{D'_x}$ eCO is initialized with the initial state of the compressed oracle. $\mathtt{eCO.RO}$ is quantum-accessible, $\mathtt{eCO.E}$ is a classical oracle interface that, on input t, applies $\mathcal{M}^t$ to eCO's internal state. The following properties of eCO are required for the main lemma that will be used in our proof. We define a relationship R based on function f and a target value t as all input output pairs that satisfy function $f(x, y) = t$. $R_{f,t}(x, y) :\Leftrightarrow f(x, y) = t$. Next, lets define

$\Gamma_R := \max_x |\{y \mid R(x,y)\}|$. Γ_R outputs the maximum number values y that satisfy a relationship R for some input x. And finally we define $\Gamma(f) = \max_t \Gamma_{R_{f,t}}$. $\Gamma(f)$ outputs the maximum number of outputs y that satisfy some relationship $R_{f,t}$ (maximizing over target values t) for some input x. The $\Gamma(f)$ value determines whether we are able to use extractions without disturbing the random oracle too much. This is stated in the following lemma.

Lemma 1 (Part of theorem 3.4 in [DFMS22], formulated in [HM24])**.** *The extractable RO simulator* `eCO` *described above, with interfaces* `eCO.RO` *and* `eCO.E`*, satisfies the following properties.*

1. *If* `eCO.E` *is unused,* `eCO` *is perfectly indistinguishable from a random oracle.*
2. *Any two subsequent independent queries to* `eCO.RO` *commute. In particular, two subsequent classical* `eCO.RO`*-queries with the same input x give identical responses.*
3. *Any two subsequent independent queries to* `eCO.E` *commute. In particular, two subsequent* `eCO.E` *queries with the same input t give identical responses.*
4. *Any two subsequent independent queries to* `eCO.E` *and* `eCO.RO` *$8\sqrt{2\Gamma(f)/2^n}$-almost-commute.*

In our application we will not give the adversary the extraction interface. We still restrict the adversary only to the RO queries. On the other hand the reduction is able to perform extraction queries to check if the oracle has been queried on the inputs that are interesting for us.

2.1 BPR model of PAKE security

We restate the model and preliminaries given in [AHH+23]. The security analysis is based on the BPR model for authenticated key exchange [BPR00]: security of a protocol Π is modeled through a security experiment in which the attacker interacts with oracles that represent honest parties (`Execute` and `Send`) as well as oracles that represent leakage of secret material (`Reveal` and `Corrupt`), and wins if it can distinguish an established session key from random. The involved oracles are described in more detail in Fig. 2. To exclude trivial attacks from consideration, [BPR00] define a freshness condition (Definition 10 below) that permits revealing a key on one side, and then testing the other (partnered) side, where partnered is defined as follows:

Definition 9 (Partnering)**.** Two instances $(P,i),(P',j)$ are partnered iff both instances have accepted (i.e. reached an `accept` instruction) with the same transcript and session key.

Intuitively, 'unfreshness' expresses that the adversary may have learned the to-be-tested session's key SK in a trivial way, i.e., by having interacted with the oracles revealing secret information in a way such that SK becomes trivially derivable regardless of the protocol's nature. Concretely, the cases we cover in our freshness definition below are a) simply requesting the key from the `Reveal` oracle, and b) learning a password pw via `Corrupt` and then actively interfering with the test session, e.g., using pw to manipulate the peer into using a session key of the adversary's choosing.

Definition 10 (Freshness with Forward Secrecy)**.** Suppose that the adversary made exactly one `Test` query, and it was to party P and instance i. We say session i of party P is **unfresh** if at any time, there was a `Reveal` query to instance (P,i) above or the instance (P',j) that it is partnered with. We also say the session is **unfresh** if both the following conditions hold:

Query	Return Value	Description
$\texttt{Execute}(P, i, P', j)$	(apk, c, tag)	Passive attack: Return transcript of an honest protocol execution between parties P and P', using the ith/jth session of P/P'.
$\texttt{Send}(P, i, msg, flow)$	msg'	Active attack: Send message msg to the oracle representing honest party P, causing it to proceed depending on its state. Flow indicator enumerates the messages in a run of the protocol and improves readability of the oracle.
$\texttt{Reveal}(P, i)$	$\text{SK}[P, i]/\perp$	Session key leakage: Return session key SK of (P, i) iff (P, i) terminated, else $\perp$; marks this instance and its matching instance "unfresh".
$\texttt{Corrupt}(P, \texttt{PWD}')$	$\text{PWD}[P, :]$	Password leakage or overwrite: Either return dictionary of passwords $\text{PWD}[P, :]$ held by party P, or allow adversary to overwrite password dictionary with PWD'$[P, :]$.
$\texttt{Test}^b(P, i)$	$\text{SK}[P, i]/SK^{\$}$	Session key challenge: Attack ith session of party P. Only for fresh, accepting instances. Returns either real or random session key depending on challenge bit b.
$\texttt{KDF}(pw)$	k_{pw}	Random oracle, input password $pw \in \text{PWD}$, output Ideal Cipher key k_{pw}.
$\texttt{H}(msg)$	tag, SK	Random oracle, input message msg, output $tag \in \mathcal{T}$ and $SK \in \mathcal{SK}$.
$\mathsf{E}(k, m)$	c	Ideal cipher encryption on input (key, message).
$\mathsf{D}(k, c)$	m	Ideal cipher decryption on input (key, ciphertext).

Figure 2: Overview of the PAKE adversary's oracles provided by the security game. Top part (above double midrule): oracles present in the BPR model. Bottom part: Random oracles and ideal cipher oracles to which the attacker additionally has access to when attacking the OCAKE protocol.

- Before the `Test` query, there was a `Corrupt` query on the test session's holder P or its partnered peer P'.
- One of the messages sent to P concerning the test session was *manipulated* by the adversary, i.e., there was a `Send`(P, i) query.

The session of (P, i) is only considered **fresh** if neither of these conditions are met.

Experiment $\mathrm{Exp}^{\mathsf{BPR}}_{\Pi}(\mathcal{A})$

18 $b \xleftarrow{unif} \$\ \{0,1\}$
19 $b' \leftarrow \mathcal{A}^{\mathcal{O}^b}(\mathcal{P})$
20 **return** $[\![b = b']\!]$

Figure 3: The BPR security game for active adversaries. $\mathcal{O}^b =$ indicates the collection of oracles {`Execute`, `KDF`, `H`, E, D, `Send`, `Reveal`, `Corrupt`,`Test`^{b}}. Here, $\mathcal{P}$ is the party set.

Definition 11 (Key indistinguishability of PAKE)**.** Let Π be a PAKE protocol. We say that an adversary $\mathcal{A}$, run in experiment $\mathrm{Exp}^{\mathsf{BPR}}_{\Pi}$, *wins* if it correctly guesses the bit according to which the test query was defined and if the `Test` query was issued for a party (P, i) that has terminated and is fresh (see Definition 10). We define the *advantage of* $\mathcal{A}$ *against a PAKE protocol* Π as

$$\mathbf{Adv}^{\mathsf{BPR}}_{\Pi}(\mathcal{A}) := |\Pr[\mathrm{Exp}^{\mathsf{BPR}}_{\Pi}(\mathcal{A}) \Rightarrow 1] - 1/2| \ .$$

3 Protocols CAKE and OCAKE

In our analysis of the required proof techniques for post-quantum PAKE, we focus on two related protocols: OCAKE (Fig. 4) and CAKE (Fig. 4).

These two protocols are variants of the EKE paradigm for PAKE, instantiated by KEM. In both protocols, the goal is the same: two parties in possession of a shared password pw want to establish a shared session key. First, the initiator **I** generates a KEM key pair (pk, sk). It then encrypt the public key pk under the password using a symmetric cipher, and sends the resulting encrypted public key apk to the responder **R**. Upon receiving apk, **R** uses the password to recover pk, which it then used to compute an encapsulation c and a pre-key K. At this point, the two protocols diverge: In the OCAKE protocol, **R** sends the encapsulation c to **I**, together with a responder tag tag computed using the password that the **I** can then use to confirm the key. In the CAKE protocol, instead of sending c and the tag, **R** sends an encryption of c, again using a symmetric cipher and the password. Upon receiving (the encryption of) c, **I** (decrypts and) decapsulates the ciphertext to obtain a pre-key K', which can then again be used to derive the session key. In both cases, **R** then derives its session key SK' from pre-key K and the session transcript. In OCAKE, **I** only derives the session key if key confirmation succeeds, i.e., if the received tag equals the one **I** derives from its own state. On the other hand, in this variant(without explicit rejection/key confirmation), **R** always derives a key and does not abort.

Informally, the security guarantee is that **R** and **I** end up with the same session key if and only if both parties used the same password and that KEM worked correctly. Both protocols require an additional round of key confirmation in order to achieve explicit authentication. However, as CAKE is implicitly authenticating for both parties, it requires confirmation for both parties, while OCAKE is explicitly rejecting on the initiator side (the Responder or server is explicitly authenticated.)

We made a minor conceptual modification to the previous descriptions of the OCAKE protocol: we derive tag and SK in the same call to `H`. This variant constitutes a minor optimization and simplifies the analysis in the quantum random oracle model.

Initiator I (Client)	**Protocol** OCAKE	**Responder R (Server)**
Password pw		Password pw
	Public Key	
$k_{pw} \leftarrow \mathtt{KDF}(pw)$		$k_{pw} \leftarrow \mathtt{KDF}(pw)$
$(pk, sk) \leftarrow\!\$\ \mathsf{KGen}$		
$apk \leftarrow \mathsf{BC}(k_{pw}, pk)$	$\xrightarrow{apk}$	$pk' \leftarrow \mathsf{BC}^{-1}(k_{pw}, apk)$
	Session Pre-Key	
$K' \leftarrow \mathsf{Decap}(sk, c)$	$\xleftarrow{c}$	$(c, K) \leftarrow\!\$\ \mathsf{Encap}(pk')$
	Key Derivation	
	$\xleftarrow{tag}$	$tag, SK \leftarrow \mathtt{H}(K, pw, apk, pk', c)$
$tag', SK' \leftarrow \mathtt{H}(K', pw, apk, pk, c)$		$\mathtt{output}\ SK$
$\mathtt{if}\ tag' = tag$		
$\quad \mathtt{output}\ SK'$		

Initiator I (Client)	**Protocol** CAKE	**Responder R (Server)**
Password pw		Password pw
	Public Key	
$(pk, sk) \leftarrow\!\$\ \mathsf{KGen}$		
$e_1 \leftarrow \mathsf{BC}_1(pw, pk)$	$\xrightarrow{e_1}$	$pk' \leftarrow \mathsf{BC}_1^{-1}(pw, e_1)$
	Session Pre-Key	
		$(c, K) \leftarrow\!\$\ \mathsf{Encap}(pk')$
$c' \leftarrow \mathsf{BC}_2^{-1}(pw, e_2)$	$\xleftarrow{e_2}$	$e_2 \leftarrow \mathsf{BC}_2(pw, c)$
$K' \leftarrow \mathsf{Decap}(sk, c')$		
	Key Derivation	
$SK' \leftarrow \mathtt{H}(pk, c', K', pw)$		$SK \leftarrow \mathtt{H}(pk', c, K, pw)$
$\mathtt{output}\ SK'$		$\mathtt{output}\ SK$

Figure 4: TOP: The OCAKE protocol, using a key encapsulation mechanism $\mathsf{KEM} = (\mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$, key derivation function KDF, hash function H and a block cipher $\mathsf{BC} = (\mathsf{BC}, \mathsf{BC}^{-1})$. We require that the message space $\mathcal{M}$ of BC and the public space $\mathcal{PK}$ of KEM coincide, i.e., $\mathcal{PK} = \mathcal{M}$. Especially, every element of $\mathcal{M}$ must constitute a valid public key for KEM. This variant enables an optimization where tag and SK are computed using the same call to H. BOTTOM: The CAKE protocol, using a key encapsulation mechanism $\mathsf{KEM} = (\mathsf{KGen}, \mathsf{Encap}, \mathsf{Decap})$, hash function H and a block cipher $\mathsf{BC} = (\mathsf{BC}, \mathsf{BC}^{-1})$.

Security of CAKE and OCAKE The first security proofs for protocols CAKE and OCAKE were given in [BCP+23] in the universal composability (UC) framework and was limited to attackers without quantum capabilities. As a first step towards capturing security even against attackers with quantum capabilities, [PZ23] and [AHH+23] gave proofs in the Bellare-Pointcheval-Rogaway (BPR) model for CAKE and OCAKE, respectively. The analysis in this work is based on these two security proofs. For completeness, we include the relevant security definitions of the BPR model in Section 2.1.

4 Definition of Programmability (SIM)

We now define programmability by specifying a 'simulator' algorithm we denote SIM. SIM interacts with an adversary, providing it with an encryption and decryption oracle. Intuitively, to that adversary, SIM is computationally indistinguishable from a family of random permutations, however it 'reprograms' responses to the adversary's queries to be inside a challenge set. Given the current uncertainty in achieving programmability for ideal ciphers with quantum queries, this definition aims to maximize flexibility in its instantiation while clearly outlining the requirement.
More formally: Consider an adversary $\mathcal{A}$, a message space $\mathcal{M}$, and a key space $\mathcal{K}_{\mathsf{SIM}}$. From these we define a family of bijections $\mathsf{E}_k : \mathcal{M} \to \mathcal{M}$ that is indexed on a key $k \in \mathcal{K}_{\mathsf{SIM}}$ where each bijection is sampled independently from the others according to distribution $\mathcal{B}$. We denote the inverse of E_k as D_k and we have $\mathsf{D}_k(\mathsf{E}_k(m)) = m \forall m \in \mathcal{M}$. Occasionally we also use $\mathsf{E}(k, m)$ to denote $\mathsf{E}_k(m)$.
In the special case where $\mathcal{B}$ is the uniform distribution over all permutations, this definition is equivalent to that of an ideal cipher.We now define the algorithm that captures the programmability requirement, parameterized on its distance $\varepsilon_{\mathsf{SIM}}$ from a true family of permutations and the probability of a programming failure δ_{SIM}:

Definition 12 (($\delta_{\mathsf{SIM}}, \varepsilon_{\mathsf{SIM}}$)−Programming Simulator SIM)**.** Let $\mathcal{A}$ be an adversary, $\mathcal{M}$ a message space, $\mathcal{K}_{\mathsf{SIM}}$ a (potentially small) key space, and E as defined above.We now define $\mathsf{SIM} = (\mathsf{E}_{\mathsf{SIM}}, \mathsf{D}_{\mathsf{SIM}})$ with respect to a challenge set $\mathcal{X} \subset \mathcal{M}$ consisting of independently and uniformly sampled elements from $\mathcal{M}$, and a key k drawn uniformly at random from the key space at the beginning of the game. We call SIM a $(\delta_{\mathsf{SIM}}, \varepsilon_{\mathsf{SIM}})$-programming simulator if, for every adversary $\mathcal{A}$ that interacts with the oracles and eventually outputs some value $apk \in \mathcal{M}$, the following holds:

1. SIM is indistinguishable from family(E, D) up to some failure probability $\varepsilon_{\mathsf{SIM}}$. I.e., the success probability of any distinguishing adversary $\mathcal{A}_{dist}$ that wins if it outputs 0 when interacting with E, D and 1 when interacting with SIM $(\mathcal{X})$ is bounded by $\varepsilon_{\mathsf{SIM}}$:

$$|\Pr[\mathcal{A}_{dist}^{\mathsf{E},\mathsf{D}} \Rightarrow 0] - \Pr[\mathcal{A}_{dist}^{\mathsf{SIM}(\mathcal{X})} \Rightarrow 0]| \leq \varepsilon_{\mathsf{SIM}}.$$

2. The decryption of the apk output by the adversary is in the challenge set $\mathcal{X}$ with probability at least $1 - \delta_{\mathsf{SIM}}$:

$$\Pr[apk \leftarrow \mathcal{A}^{\mathsf{SIM}(\mathcal{X})} : \mathsf{D}_{\mathsf{SIM}}(k, apk) \in \mathcal{X}] \geq 1 - \delta_{\mathsf{SIM}}.$$

Generalization to multiple sessions We define the multi-session variant of SIM analogously to the single-session version, except the adversary now outputs a target session identifier with each of the n_s ouptuts: (apk_i, sid_i). In the later reduction, we see that we can achieve a tight reduction to the multi-user security of KEM whenever all n decryptions are in the challenge set: $\forall(apk_i, sid_i)$ output by $\mathcal{A}$, we require that $\mathsf{D}_{k(sid_i)}(apk_i) \in \mathcal{X}$ with probability $1 - \delta_{\mathsf{SIM}}$.

Definition 13 (($n, \delta_{\mathsf{SIM}}, \varepsilon_{\mathsf{SIM}}$)−Programming Simulator SIM (multi-session))**.** Let message space $\mathcal{M}$, key space $\mathcal{K}_{\mathsf{SIM}}$ and E be as above.Let $\mathcal{A}$ be a multi-stage adversary $\mathcal{A}_0, ...\mathcal{A}_{n-1}$. We now define simulator SIM: $(\mathsf{E}_{\mathsf{SIM}}, \mathsf{D}_{\mathsf{SIM}})$ relative to a challenge set $\mathcal{X} \subset \mathcal{M}$ of independently and uniformly random elements of the message space $\mathcal{M}$ and at most n keys K_i that are sampled uniformly at random from the key space at the beginning of the game s.t. each session is associated with one such key. Each adversary $\mathcal{A}_i$ interacts with the oracles and outputs some value $apk_i \in \mathcal{M}$, a session ID sid_i and a state st. We say that SIM is a $(\delta_{\mathsf{SIM}}, \varepsilon_{\mathsf{SIM}})$−programming simulator whenever we have for any such adversary that:

1. SIM is indistinguishable from family (E,D) up to some failure probability $\varepsilon_{\mathsf{SIM}}$, i.e., the success probability of any distinguishing adversary $\mathcal{A}_{dist}$ that wins if it outputs 0 when interacting with E, D and 1 when interacting with SIM ($\mathcal{X}$) is bounded by $\varepsilon_{\mathsf{SIM}}$: $|\Pr[\mathcal{A}_{dist}^{\mathsf{E},\mathsf{D}} \Rightarrow 0] - \Pr[\mathcal{A}_{dist}^{\mathsf{SIM}(\mathcal{X})} \Rightarrow 0]| \leq \varepsilon_{\mathsf{SIM}}$.
2. the decryption of every $apk_i, i \in \{0, 1, ..., n-1\}$ is in the challenge set $\mathcal{X}$ with probability at least $1 - \delta_{\mathsf{SIM}}$:

$$\Pr[(apk_i, sid_i, st_i) \leftarrow \mathcal{A}_i(st_{i-1})^{\mathsf{SIM}(\mathcal{X})} : \mathsf{D}_{\mathsf{SIM}}(k[sid_i], apk_i) \in \mathcal{X}] \geq (1 - \delta_{\mathsf{SIM}}).$$

where $k[sid_i]$ is the key associated with session sid_i.

5 Necessity of SIM for OCAKE

We now prove that any generic security proof for (O)CAKE in the BPR model necessarily relies on programming, exactly in the sense captured by our notion of a programming simulator SIM (Definition 12).
We first show that (O)CAKE becomes insecure if the underlying KEM does not provide anonymity by providing an attack on the protocol based on anonymity. Hence, any proof that aims to avoid assuming statistical anonymity must incorporate a reduction from the anonymity of the KEM.Using a meta-reduction, we then demonstrate that any generic reduction from anonymity must either program the cipher decryptions in the sense above, or else it is already powerful enough to break anonymity on its own. To this end, we first notice that identifying a single KEM s.t. the meta-reduction holds implies that no generic reduction can exist, as it would have to cover this KEM. We identify such a class in KEMs obtained from a PKE scheme via the full Fujisaki-Okamoto transform, provided they achieve $\mathsf{ANO\text{-}CPA}$ and that their explicitly rejecting variant satisfies a weak robustness condition called hROB. This class contains Kyber and likely most other practically relevant post-quantum KEMs.

The meta-reduction features an adversary algorithm that follows the same blueprint as the above-mentioned attack, however since this adverary is tailored to this specific KEM class, it is more efficient than the generic attack. More concretely, we require weak robustness to enable the meta-reduction to supply the reduction with this successful adversary. Other meta-reductions often build adversaries either by rewinding the reduction or by working within strong idealized frameworks such as the algebraic group model. In contrast, we rely on (Q)ROM techniques to equip our adversary with the ability to solve the anonymity problem.

More concretely, we prove that for this class of KEMs, anonymity fails statistically, since a ciphertext generated under one of two public keys will, with high probability, not be valid under the other public key. Consequently, access to an oracle that decides whether a ciphertext is 'valid' under a given public key allows to break anonymity. In contrast, we must argue that access to such a validity oracle does not trivialize solving the anonymity problem for a fixed pair of public keys, as long as this particular pair is excluded from the allowed inputs. The crucial point is this: as long as the reduction does not implement SIM, all public keys for which the meta-reduction must simulate `Valid` are distinct from the challenge keys. Therefore, it suffices to show that the meta-reduction can consistently simulate the `Valid` oracle for FO-transformed KEMs towards the adversary, for all inputs except on the real anonymity challenge.

Anonymity is Necessary. In security proofs for PAKE, it is crucial to exclude dictionary attacks in which an adversary could perform offline computations to recover the password used in specific transcripts. For this reason, proofs typically include steps that unlink the protocol messages from each other and from the password. Consider, for instance, the motivation behind the anonymity game hop for OCAKE: the first protocol message

is an encryption of an ephemeral public key under the password. If an attacker could infer too much information about this public key from the second message (which is an encapsulation under that public key), then they could mount a dictionary attack by attempting to decrypt the first message with candidate passwords and checking whether the resulting public key matches the information obtained.

To formalize this intuition, we show how to translate an attack on anonymity of KEM into an attack against the security of OCAKE.

Theorem 3 (OCAKE is insecure if KEM is not anonymous)**.** *Let $\mathcal{B}$ be an algorithm in the anonymity game against* KEM *with advantage* $\mathbf{Adv}^{\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{B})$*, i.e. by definition we have:*

$$\mathbf{Adv}^{\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{B}) = |\Pr[\mathsf{ANO\text{-}PCA}^0(\mathcal{B}) \Rightarrow 0] - \Pr[\mathsf{ANO\text{-}PCA}^1(\mathcal{B}) \Rightarrow 0]|$$

i.e., $\mathcal{B}$ outputs 0 whenever it guesses pk_0 was used, and 1 if pk_1. Then, $\mathcal{B}$ can be used to launch an attack on the BPR security of PAKE if OCAKE *is instantiated with this* KEM*.*

We describe the adversary $\mathcal{A}$ against the security of OCAKE built from $\mathcal{B}$ in pseudocode in Fig. 5. We give an analysis of the success probability of this attack, however we note that the attack given here is included to show the concept. The actual attacker used in the meta-reduction in the next section has, informally speaking, the additional guarantee that w.h.p., the ciphertext will be valid under only one public key out of a set. This allows it to employ a majority-vote pairwise comparison strategy that is not guaranteed to work in the general case. We thank an anonymous reviewer for first noticing this bug.

Attack Analysis Let $\mathcal{A}$ be the attack algorithm with access to an anonymity adversary $\mathcal{B}$. First, the adversary $\mathcal{A}$ obtains a transcript apk, c, tag by running a passive attack on the protocol. Then, $\mathcal{A}$ assigns each password in the dictionary a unique index in $[\mathcal{D}]$. We denote the index of the 'correct' password as i^* and the ciphertext c becomes c_{i^*} to indicate that it was computed on i^*. Next, $\mathcal{A}$ selects two passwords pw, pw' from the dictionary, which is equivalent to sampling two indices i and j from $[\mathcal{D}]$, with the condition $i \neq j$. We denote the two selected candidate keys as $pk_0 = \mathsf{D}(pw_i, apk), pk_1 = \mathsf{D}(pw_j, apk)$. $\mathcal{A}$ then runs $\mathcal{B}$ on input c, pk_0, pk_1 and outputs i if $\mathcal{B}$ guesses 0 and j if $\mathcal{B}$ guesses 1. Right away we see that $\mathcal{B}$ outputs either 0 or 1, so in all cases where $i^* \notin (i, j)$, $\mathcal{A}$ loses. However, in the case where one of the passwords was correct, $\mathcal{B}$ is presented with an ANO-CPA challenge. Finally, $\mathcal{A}$ finishes the attack by running the sender side of the protocol using pw^* and sending the output to the receiver. The probability of $\mathcal{A}$ using the correct password in its attack is therefore increased by the anonymity advantage of $\mathcal{B}$, provided the correct password was among the two that were sampled. From this we conclude that the attack's success probability is increased to $\frac{1+\mathbf{Adv}^{\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}}{|\mathcal{D}|}$. Given that $|\mathcal{D}|$ is cryptographically small by assumption, it follows that whenever KEM is not anonymous, OCAKE is vulnerable to this attack.

Finally, we note that this attack can be greatly improved whenever $\mathcal{B}$ is additionally able to distinguish the case where it is run on two public keys s.t. *neither* of them was used to generate c. E.g., if we extend $\mathcal{B}$ to output an error symbol with high probability in this case, $\mathcal{A}$ can simply run $\mathcal{B}$ pairwise on different passwords until it finds one that matches. We show that this is attainable for many KEM, a fact which we formalize further in the following meta-reductions. For the unoptimized case, we can make an argument that lower bounds the success of $\mathcal{A}$ over the randomness of i and j (and any internal randomness of $\mathcal{B}$):

$$\begin{aligned}\Pr_{i,j}[\mathcal{A}_{(i,j)} \Rightarrow i^*] &= \Pr_{i,j}[\mathcal{A}_{(i,j)} \Rightarrow i^* \wedge i^* \in \{i,j\}] + \underbrace{\Pr[\mathcal{A}_{(i,j)} \Rightarrow i^* \wedge i^* \notin \{i,j\}]}_{=0\ (\mathcal{A}\text{ outputs } i \text{ or } j)} \\ &= \Pr_{i,j}[\mathcal{B}(c_{i^*}, pk_0, pk_1) \Rightarrow 0 | i = i^*] \cdot \underbrace{\Pr[i = i^*]}_{=\frac{1}{|\mathcal{D}|}}\end{aligned}$$

Algorithm $\mathtt{Attack}$

$P, P' \leftarrow \mathcal{P}^2$ // Choose parties to attack
$(apk, c, tag^*) \leftarrow \mathtt{Execute}(P, P')$
$pw_0, pw_1 \leftarrow\!\$\, \mathcal{D}^2$ // Sample two passwords to distinguish
for pw_0, pw_1: $pk[pw] \leftarrow \mathsf{BC}^{-1}(pw, apk)$
$b \leftarrow \mathcal{B}(c, pk[pw_0], pk[pw_1])$ //Chance to win if one pw correct
$pw^* \leftarrow pw_b$ //select appropriate password
$(pk^*, sk^*) \leftarrow \mathsf{KGen}$ // Attack using candidate password
$apk^* \leftarrow \mathsf{BC}(pw^*, pk^*)$
$(c, tag^*) \leftarrow \mathtt{Send}^1(P', apk^*)$
$K^* \leftarrow \mathsf{Decap}(sk^*, c)$
$SK \leftarrow \mathtt{KDF}(\ldots, K^*)$
$SK^* \leftarrow \mathtt{Test}(P')$
if $SK == SK^*$ return 0
else return 1

Figure 5: Attack Algorithm for adversary $\mathcal{A}$ using anonymity adversary $\mathcal{B}$ to attack security of OCAKE. The attack first obtains a ciphertext, computes the the most likely password based on the anonymity adversary, and then actively attacks a session with that password.

$$
\begin{aligned}
&+ \Pr_{i,j}[\mathcal{B}(c_{i^*}, pk_0, pk_1) \Rightarrow 1 | j = i^*] \cdot \underbrace{\Pr[j = i^* \wedge i \neq i^*]}_{= \frac{1}{|\mathcal{D}|-1} \cdot (1 - (\frac{1}{|\mathcal{D}|})) = \frac{1}{|\mathcal{D}|}} \\
&= \frac{1}{|\mathcal{D}|} \cdot (\Pr_{i,j}[\mathcal{B}(c_{i^*}, pk_0, pk_1) \Rightarrow 0 | i = i^*] + \underbrace{\Pr_{i,j}[\mathcal{B}(c_{i^*}, pk_0, pk_1) \Rightarrow 1 | j = i^*]}_{= 1 - \Pr_{i,j}[\mathcal{B} \Rightarrow 0 | j = i^*]}) \\
&= \frac{1}{|\mathcal{D}|} \cdot (1 + \Pr[\mathcal{B}(c_0, pk_0, pk_1) \Rightarrow 0] - \Pr[\mathcal{B}(c_1, pk_0, pk_1) \Rightarrow 0]) \\
&= \frac{1 + \mathbf{Adv}_{\mathsf{ANO\text{-}CPA}}(\mathcal{B})}{|\mathcal{D}|} \geq \frac{1}{|\mathcal{D}|}.
\end{aligned}
$$

5.1 Any successful anonymity reduction must program.

We now use the meta-reduction technique to investigate how any successful reduction between the security of OCAKE and the anonymity of KEM must program the the cipher. More precisely, we prove Theorem 1 which states that there can be no reduction from the anonymity property of KEM to some game hop in the proof *unless* the reduction implements a programming simulator SIM that programs the block cipher exactly as defined in Definition 12. To this end, we distinguish two complementary cases, based on a flag that records whether the "programming" condition is satisfied: any reduction that *does not* trigger this flag directly yields a successful anonymity adversary (Lemma 2), while any reduction that *does* trigger it has programmed exactly as required by the definition (Lemma 3).

Our strategy is as follows. We restrict our attention to a specific class of KEM, which we refer to as $\mathtt{Valid}$-resistant Anonymous KEM and formally define in Definition 14. Informally, such KEMs preserve anonymity even when the adversary is granted access to a "validity" oracle which, on input a public key pk and a ciphertext c, outputs whether or not c is a valid ciphertext under pk (while rejecting whenever the challenge values are queried). We prove that, given access to a $\mathtt{Valid}$ oracle, one can decide anonymity for δ-correct KEMs that are hROB in their explicitly rejecting variant (Lemma 4). Building

Oracle $\texttt{Valid}^{c^*}(c, pk)$
01 if $c = c^*$ **return** $\perp$
02 $sk \leftarrow pk.\mathsf{compute_secret_key}$ //unbounded algorithm
03 $k \leftarrow \mathsf{Decap}^*(sk, c)$
04 if $k \neq \perp$ **return** `True`
05 else **return** `False`

Figure 6: Validity oracle `Valid` outputs True if and only if c decrypts successfully under some secret key associated with public key pk. Decap^* is an algorithm that runs decapsulation, but indicates decryption failures even for implicitly rejecting schemes. This can be instantiated in various ways, e.g. using a re-encryption check.

on this, we then demonstrate that the meta-reduction can construct an adversary against OCAKE by combining the `Valid`-based anonymity attack with the anonymity attack on OCAKE described above.

We then prove that KEMs derived via the full FO transform (including the de-randomization step commonly called the T-transform [HHK17]) are `Valid`-resistant Anonymous KEMs (Lemma 5), meaning that the `Valid` oracle does not provide any additional power in solving the anonymity challenge (as long as it rejects on the challenge elements). We then combine the attack algorithm sketched above with these two lemmas to establish Lemmas 2 and 3.

We proceed by introducing the definition of `Valid`-resistant Anonymous KEM. In this definition, we compare the advantage of two adversaries in the ANO-PCA experiment (cf. Definition 6), where one of them is additionally granted access to the validation oracle described earlier. This oracle is specified in Fig. 6. In defining this oracle, we postulate an unbounded challenger that can brute-force secret keys in order to recover a valid secret key corresponding to a given public key. We refer to this brute-force procedure as $\mathsf{compute_secret_key}$. Later, we will argue that for KEMs obtained via the FO transform, one can efficiently emulate this oracle using the random oracle employed in the T-transform to de-randomize the underlying scheme.

Definition 14 (`Valid`-resistant Anonymous KEM)**.** Let KEM be a key encapsulation mechanism with public-key space $\mathcal{PK}$ and key space $\mathcal{K}$. KEM is $\varepsilon_{\texttt{Valid}}$*−`Valid`-resistant anonymous* if the distance between the ANO-PCA-advantage of an adversary $\mathcal{B}$ with access to an additional oracle `Valid` as in Fig. 6 to that of a regular anonymity adversary $\mathcal{A}$ in the ANO-PCA game is bounded by $\varepsilon_{\texttt{Valid}}$. I.e., we have that $\mathbf{Adv}^{\mathsf{KEM}}_{\mathsf{ANO\text{-}PCA}}(\mathcal{B}^{\texttt{Valid}}) \leq \mathbf{Adv}^{\mathsf{KEM}}_{\mathsf{ANO\text{-}PCA}}(\mathcal{A}) + \varepsilon_{\texttt{Valid}}$.

Proof of Theorem 1. Informal Theorem 1 states that there cannot exist a non-programming reduction. To prove this claim, we give a meta-reduction $\mathcal{M}$ and show that: whenever $\mathcal{R}$ does not program, it solves the anonymity challenge without the adversary and when it does, it does so according to the definition of SIM. This is formalized in Proposition 1.

Proposition 1 (Any successful reduction either breaks anonymity or programs.)**.** *Let $\mathcal{R}$ be any generic reduction from* ANO-PCA *security to the distinguishing problem in the proof of* OCAKE *with advantage $\mathbf{Adv}^{\mathcal{R}}_{\mathsf{ANO\text{-}PCA}}(\mathcal{A})$ given a distinguishing adversary $\mathcal{A}$. Then, the meta-reduction $\mathcal{M}$ providing $\mathcal{R}$ with such an adversary $\mathcal{A}_{\mathcal{M}}$ succeeds in either the anonymity or simulation game. More formally, the success probability of $\mathcal{R}$ is upper-bounded by it's ability to program (defined by the programming error δ_{SIM}) or solve anonymity:*

$$\Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A}_{\mathcal{M}}}) \Rightarrow 1] \leq \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{M}^{\mathcal{R}}) \Rightarrow 1] + \Pr[\mathrm{G}_{\mathsf{SIM}}(\mathcal{M}^{\mathcal{R}}) \Rightarrow 1]$$

We prove this proposition in two parts, giving a meta-reduction for each case.

Lemma 2 ($\mathcal{M}$ is a successful anonymity adversary whenever $\mathcal{R}$ *does not* program)**.** *For* `Valid`*-resistant anonymous, honest robust and* $\delta_{\mathsf{KEM}}-$*correct KEM, the meta-reduction in figure 7 wins the anonymity game whenever* $\mathcal{R}$ *does, provided that* program *does not occur:*

$$\Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{M}^{\mathcal{R}}) \Rightarrow 1] \geq \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A},\mathcal{M}}) \Rightarrow 1 | \neg \mathsf{program}]$$

Lemma 3 ($\mathcal{M}$ implements a programming simulator according to SIM *whenever* $\mathcal{R}$ programs)**.** *For* `Valid`*-resistant anonymous, honest robust and* $\delta_{\mathsf{KEM}}-$*correct KEM, the meta-reduction in figure 7 succeeds in programming according to* SIM *for any simulation adversary* $\mathcal{A}_{\mathsf{SIM}}$ *whenever* program *occurs:*

$$\Pr[(apk, sid) \leftarrow {\mathcal{A}_{\mathsf{SIM}}}^{\mathcal{M}(\mathcal{R}.\mathsf{E}_{\mathsf{SIM}}, \mathcal{R}.\mathsf{D}_{\mathsf{SIM}})} : \mathsf{D}_{\mathsf{SIM}}(K[sid], apk) \in \mathcal{X}] = \Pr[\mathsf{program}]$$

Therefore the meta-reduction using the reduction to implement a programming simulator SIM *has programming failure probability:*

$$\delta_{\mathsf{SIM}} = 1 - \Pr[\mathsf{program}].$$

With these supplementary lemmata in hand, we can give a combined bound:

Proposition 1. We split the success probability of any reduction $\mathcal{R}$:

$$\begin{aligned}
\Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A},\mathcal{M}}) \Rightarrow 1] &= \underbrace{\Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A},\mathcal{M}}) \Rightarrow 1 | \neg \mathsf{program}]}_{\substack{\leq \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{M}^{\mathcal{R}}) \Rightarrow 1] \\ (\textit{Lemma 2})}} \cdot \underbrace{\Pr[\neg \mathsf{program}]}_{\leq 1} \\
&\quad + \underbrace{\Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A},\mathcal{M}}) \Rightarrow 1 | \mathsf{program}]}_{\leq 1} \cdot \underbrace{\Pr[\mathsf{program}]}_{\substack{\leq (1-\delta_{\mathsf{SIM}}) \\ (\textit{Lemma 3})}} \\
&\leq \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{M}^{\mathcal{R}}) \Rightarrow 1] + (1 - \delta_{\mathsf{SIM}}).
\end{aligned}$$

□

Proof of Lemma 2. The success of the meta-reduction 7 in the anonymity game is shown in two parts: First we show that to the reduction, the meta-reduction provides a successful anonymity adversary. For this we show how to implement the anonymity attack from above using the `Valid` oracle. Second, we show that the meta-reduction does not trivialize the challenge for the reduction.To do so, we show that the internal oracles can be simulated when the KEM is obtained via the FO-transform, including the T-transform.

Proof of Lemma 2 (part 1): The meta-reduction provides a successful adversary. By definition, a PAKE adversary wins its game whenever it distinguishes a session key from random. We show the meta-reduction $\mathcal{M}$'s attack algorithm instantiated using the `Valid` oracle behaves as a successful PAKE adversary leveraging non-anonymity of the scheme. We summarize the stages of the attack. The attacker:

1. *queries* the reduction's on an authenticated public key apk to receive (c, tag),
2. *determines* the most likely public key of that session,
3. *derives* the password associated with that public key, and
4. *attacks* a fresh session using that password.

To determine the most likely public key, the meta-reduction makes use of the `Valid` oracle, querying on the ciphertext c received from the reduction and each of the candidate public keys pk_{pw_i} associated with the different passwords. We show that the attack succeeds whenever KEM is correct and its explicitly rejecting version is honest robust (hROB).

Meta-reduction $\mathcal{M}^{\mathcal{R}}(\mathcal{A}_{\mathsf{SIM}}, \mathcal{C}^{\mathsf{ANO\text{-}PCA}})$

Simulated **Adversary** $\mathcal{A}_{\mathcal{M}}$

forward $\mathcal{A}_{\mathsf{SIM}}$'s queries to $\mathcal{R}.\mathsf{BC}$,$\mathcal{R}.\mathsf{BC}^{-1}$
$(apk^*, sid) \leftarrow \mathcal{A}_{\mathsf{SIM}}$ //Select point to attack
$(c^*, tag^*) \leftarrow \mathcal{R}.\mathtt{Send}^1(P'[sid], apk^*)$
for $pw \in \mathcal{D}$: //Improvement over generic attack since `Valid` works on any pk
  $pk \leftarrow \mathcal{R}.\mathsf{BC}^{-1}(pw, apk)$ // program if $pk \in \mathcal{X}$ and $pw = PW[P']$
  if $\mathtt{Valid}(c, pk) : pw^* \leftarrow pw$ // unique and simulatable for certain schemes
$(pk^*, sk^*) \leftarrow \mathsf{KGen}$ // Attack using candidate password
$apk \leftarrow \mathcal{R}.\mathsf{BC}(pw^*, pk^*)$
$(c^*, tag^*) \leftarrow \mathcal{R}.\mathtt{Send}^1(P'[sid], apk)$
$K^* \leftarrow \mathsf{Decap}(sk^*, c^*)$
$SK \leftarrow \mathcal{R}.\mathtt{KDF}(\ldots, K^*)$
$SK^* \leftarrow \mathcal{R}.\mathtt{Test}(P'[sid])$
if $SK == SK^*$ return 0 to $\mathcal{R}$, else return 1

Meta-reduction $\mathcal{M}^{\mathcal{R}}$

$b' \leftarrow \mathcal{R}^{\mathcal{C}^{\mathsf{ANO\text{-}PCA}}, \mathcal{A}_{\mathcal{M}}}$ //Run $\mathcal{R}$ with access to challenger and adversary
output b'

Figure 7: Pseudocode for the meta-reduction algorithm using reduction $\mathcal{R}$ for challenges in the anonymity or simulation game. The meta-reduction is defined relative to the anonymity challenger $\mathcal{C}^{\mathsf{ANO\text{-}PCA}}$, the simulation adversary $\mathcal{A}_{\mathsf{SIM}}$, and the reduction $\mathcal{R}$. We have for the respective challenges of the anonymity and simulation that $\mathcal{X} = \{pk_0, pk_1\}$. The validity oracle can be efficiently simulated for `Valid`-resistant KEM (Lemma 6). Note that this attack is an improvement over the one presented in the previous section since for the schemes considered here, we can build an adversary that is stronger than a standard anonymity adversary.

Lemma 4 (Attack using `Valid` oracle is successful if KEM is hROB and correct)**.** *For* KEM *that is* δ_{KEM}*−correct and honest robust in its explicitly rejecting version, an attacker with access to a* `Valid` *oracle (Fig. 6) is a successful* BPR *PAKE adversary. More formally, we show that for an* hROB *adversary* $\mathcal{B}$ *and dictionary size* $\mathcal{D}$*:*

$$\Pr[\mathrm{G}_{\mathsf{BPR}}(\mathcal{A}_{\mathcal{M}}^{\mathtt{Valid}}) \Rightarrow 1] \geq 1 - |\mathcal{D}| \cdot \mathbf{Adv}^{\mathsf{hROB}}_{\mathsf{KEM}^{\perp}}(\mathcal{B}) - 2\delta_{\mathsf{KEM}}.$$

Lemma 4. We prove Lemma 4 by analyzing the success probability of the meta-reduction's adversary. The game hops for this proof are shown in Fig. 8. The first change is semantic and $\Pr[\mathrm{G}_1 \Rightarrow 1] = \Pr[\mathrm{G}_0 \Rightarrow 1]$. We obtain G_2 from G_1 by aborting on correctness errors. For KEM that is δ_{KEM}−correct, we bound the distance to the previous game as $|\Pr[\mathrm{G}_2 \Rightarrow 1] - \Pr[\mathrm{G}_1 \Rightarrow 1]| \leq \delta_{\mathsf{KEM}}$. G_3 adds another abort when decapsulation under the wrong pk does not result in a failure. The distance can be bounded via a hybrid argument (adding an abort for each password at a time) and the honest robustness property of KEM: $|\Pr[\mathrm{G}_3 \Rightarrow 1] - \Pr[\mathrm{G}_2 \Rightarrow 1]| \leq |\mathcal{D}| \cdot \mathbf{Adv}^{\mathsf{KEM}^{\perp}}_{\mathsf{hROB}}$. Next, we inline the definition of the `Valid` oracle. This is a syntactic rewrite that does not change the success probability: $\Pr[\mathrm{G}_4 \Rightarrow 1] = \Pr[\mathrm{G}_3 \Rightarrow 1]$. Finally, we again abort on a correctness error in the second session and have $|\Pr[\mathrm{G}_5 \Rightarrow 1] - \Pr[\mathrm{G}_4 \Rightarrow 1]| \leq \delta_{\mathsf{KEM}}$. We see that $\mathcal{A}_{\mathcal{M}}^{\mathtt{Valid}}$ always wins in game G_5 and get $\Pr[\mathrm{G}_5 \Rightarrow 1] = 1$. In summary we get

$$\Pr[\mathrm{G}_0] \geq \underbrace{\Pr[\mathrm{G}_5]}_{1} - \underbrace{|\Pr[\mathrm{G}_5] - \Pr[\mathrm{G}_4]|}_{\leq \delta_{\mathsf{KEM}}} - \underbrace{|\Pr[\mathrm{G}_4] - \Pr[\mathrm{G}_3]|}_{0} - \underbrace{|\Pr[\mathrm{G}_3] - \Pr[\mathrm{G}_2]|}_{\leq |\mathcal{D}| \cdot \mathbf{Adv}^{\mathsf{KEM}^{\perp}}_{\mathsf{hROB}}}$$
$$- \underbrace{|\Pr[\mathrm{G}_2] - \Pr[\mathrm{G}_1]|}_{\leq \delta_{\mathsf{KEM}}} - \underbrace{|\Pr[\mathrm{G}_1] - \Pr[\mathrm{G}_0]|}_{0} \geq 1 - |\mathcal{D}| \cdot \mathbf{Adv}^{\mathsf{hROB}}_{\mathsf{KEM}^{\perp}}(\mathcal{B}) - 2\delta_{\mathsf{KEM}}$$

Oracle $\texttt{Valid}^{c^*,pk_0,pk_1}(c,pk)$

01 if $c = c^*$ return `False` //reject mapped to `False`
02 $k \leftarrow \mathsf{Decap}^*(sk[pk], c)$
03 if $k \neq \perp$ return `True` else return `False`

Game G_0, Attack Algorithm G_1 //Inline send, test

01 $pw^* \leftarrow \perp$ //Initialize guess
02 // forward $\mathcal{A}_{\mathsf{SIM}}$'s queries to $\mathcal{R}.\mathsf{BC}$,$\mathcal{R}.\mathsf{BC}^{-1}$
03 $(apk^*, sid) \leftarrow \mathcal{A}_{\mathsf{SIM}}$
04 ~~$(c^*, tag^*) \leftarrow \mathcal{R}.\texttt{Send}^1(P'[sid], apk^*)$~~
05 $\texttt{Send}^1$: $pk' \leftarrow \mathsf{BC}^{-1}(pw[P'[sid]], apk^*))$; $(c^*, K) \leftarrow \mathsf{Encap}(pk')$
06 for $(j, pw_j) \in (|\mathcal{D}|, \mathcal{D})$:
07 $pk_j \leftarrow \mathcal{R}.\mathsf{BC}^{-1}(pw, apk^*)$ //program if $\mathcal{R}.\mathsf{BC}^{-1}(pw^*, apk^*) \in \mathcal{X}$
08 if $\texttt{Valid}(c, pk_j) : pw^* \leftarrow pw_j$
09 fail if $pw^* = \perp$ //Attack fails if there is no matching pk
10 $(pk^*, sk^*) \leftarrow \mathsf{KGen}$
11 $apk \leftarrow \mathcal{R}.\mathsf{BC}(pw^*, pk^*)$
12 ~~$(c^*, tag^*) \leftarrow \mathcal{R}.\texttt{Send}^1(P'[sid], apk)$~~
13 $\texttt{Send}^1$: $pk' \leftarrow \mathsf{BC}^{-1}(pw[P'[sid]], apk))$; $(c^*, K) \leftarrow \mathsf{Encap}(pk')$
14 $K' \leftarrow \mathsf{Decap}(sk^*, c^*)$
15 $SK \leftarrow \mathcal{R}.\texttt{KDF}(\ldots, K^*)$
16 ~~$SK^* \leftarrow \mathcal{R}.\texttt{Test}(P'[sid])$~~
17 $\texttt{Test}$: $SK_0 \leftarrow \mathcal{R}.\texttt{KDF}(\ldots, K')$; $SK_1 \leftarrow\!\!\$\, \mathcal{SK}$; $SK^* \leftarrow SK_b$
18 if $SK == SK^*$ return 0, else return 1

Hybrids G_1 G_2, G_3, G_4, G_5 //Correctness, honest robustness, validity, correctness

01 $pw^* \leftarrow \perp$
02 forward $\mathcal{A}_{\mathsf{SIM}}$'s queries to $\mathcal{R}.\mathsf{BC}$,$\mathcal{R}.\mathsf{BC}^{-1}$
03 $(apk^*, sid) \leftarrow \mathcal{A}_{\mathsf{SIM}}$
04 $pk' \leftarrow \mathsf{BC}^{-1}(pw[P'[sid]], apk^*))$; $(c^*, K) \leftarrow \mathsf{Encap}(pk')$
05 win if $\mathsf{Decap}(sk[pk']) \neq K$ //Correctness (G_2)
06 for $(j, pw_j) \in (|\mathcal{D}|, \mathcal{D})$:
07 $pk_j \leftarrow \mathcal{R}.\mathsf{BC}^{-1}(pw_j, apk^*)$
08 win if $\mathsf{Decap}^*(sk_j, c) \neq \perp$ and $sk \neq sk[pk']$ //Honest robustness (G_3)
09 ~~if $\texttt{Valid}(c, pk_j) : pw^* \leftarrow pw_j$~~ //Validity ($\mathrm{G}_4$)
10 if $\mathsf{Decap}^*(sk[pk_j], c) \not\to \perp : pw^* \leftarrow pw_j$ //Validity (G_4)
11 fail if $pw^* = \perp$ //fail occurs iff $(c^*, K) \leftarrow \mathsf{Encap}(pk^{\$})$
12 $(pk^*, sk^*) \leftarrow \mathsf{KGen}$
13 $apk \leftarrow \mathcal{R}.\mathsf{BC}(pw^*, pk^*)$
14 $pk'' \leftarrow \mathsf{BC}^{-1}(pw[P'[sid]], apk))$ //$pk'' \leftarrow pk^*$
15 $(c^*, K^*) \leftarrow \mathsf{Encap}(pk'')$; $K' \leftarrow \mathsf{Decap}(sk^*, c^*)$ $K' \leftarrow K^*$ //Correctness (G_5)
16 $SK \leftarrow \mathcal{R}.\texttt{KDF}(\ldots, K')$
17 $SK_0 \leftarrow \mathcal{R}.\texttt{KDF}(\ldots, K')$; $SK_1 \leftarrow\!\!\$\, \mathcal{SK}$; $SK^* \leftarrow SK_b$
18 if $SK == SK^*$ return 0, else return 1

Figure 8: Hybrid games to show that the attack algorithm given by $\mathcal{M}$ is (1) successful and the public key used to compute c matches the correct password's public key and (2) fails whenever the public key was independent of the password provide that the scheme is correct and honest robust.

Reduction $\mathcal{B}$ (pk_0, pk_1, c)

19 $b' \leftarrow \mathcal{A}^{\mathtt{Valid}^{c^*}_{\mathcal{B}}}(pk_0, pk_1, c)$
20 **return** b'

Simulated oracle $\mathtt{G}_{\mathcal{B}}(m)$

01 $r \leftarrow \mathtt{G}(m)$
02 if $\mathsf{PKE.Enc}(pk, m, r) = c^*$
03 **return** r
04 else **return** $\mathtt{eCO.RO}(m)$

Simulated oracle $\mathtt{Valid}^{c^*, pk_0, pk_1}_{\mathcal{B}}(pk, c)$

05 if $c = c^*$ return $\perp$
06 $m \leftarrow \mathsf{eCO.E}_{c=\mathsf{PKE.Enc}(pk,m)}(c, pk)$
07 if $m \neq \perp$ **return** `True`
08 else **return** `False`

Figure 9: The simulated oracles of Adversary $\mathcal{B}$ reducing security under the additional validity oracle to ANO-PCA security.

□

We then have a bound for the success of any reduction $\mathcal{R}$ using this adversary:

$$\begin{aligned}
&\Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}}) \Rightarrow 1] = \\
&\quad \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}}) \Rightarrow 1 | \mathrm{G}_{\mathsf{BPR}}(\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}) \Rightarrow 1] \cdot \Pr[\mathrm{G}_{\mathsf{BPR}}(\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}) \Rightarrow 1] \\
&\quad + \underbrace{\Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}}) \Rightarrow 1 | \mathrm{G}_{\mathsf{BPR}}(\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}) \Rightarrow 0] \cdot \Pr[\mathrm{G}_{\mathsf{BPR}}(\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}) \Rightarrow 0]}_{\geq 0} \\
&\geq \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}^{\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}}) \Rightarrow 1 | \mathrm{G}_{\mathsf{BPR}}(\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}) \Rightarrow 1] \cdot \underbrace{\Pr[\mathrm{G}_{\mathsf{BPR}}(\mathcal{A}^{\mathtt{Valid}}_{\mathcal{M}}) \Rightarrow 1]}_{\geq 1 - |\mathcal{D}| \cdot \mathbf{Adv}^{\mathsf{hROB}}_{\mathsf{KEM}^{\perp}}(\mathcal{B}) - 2\delta_{\mathsf{KEM}}}
\end{aligned}$$

We note here that no step in the proof relies on the specific value of *apk* or on the prior queries to $\mathcal{R}.\mathsf{IC}$. Hence, the attack remains valid for any choice of $\mathcal{A}_{\mathsf{SIM}}$.Consequently, $\mathcal{A}_{\mathsf{SIM}}$ can likewise be simulated within this meta-reduction.

Proof of Lemma 2 (part 2): Anonymity is preserved under Validity oracle. To show that the validity oracle does not solve the challenge for the reduction, we investigate when ANO-PCA security is preserved under addition of the `Valid` oracle. More precisely, we show that for FO-transformed schemes any successful adversary that makes use of the `Valid` oracle can be converted into a general anonymity adversary in the QROM.

Lemma 5 (FO-transformed schemes are `Valid`-resistant Anonymous.)**.** *Let* KEM *be constructed using the T-transform [HHK17] from a PKE that is* $\gamma-$*spread and* $\delta^{\mathsf{PKE}}_{corr}-$*correct. Then access to an oracle* `Valid` *(Fig. 6) does not significantly improve the advantage of an adversary in the* ANO-PCA *game.*
Concretely, for an ANO-PCA *adversary* $\mathcal{A}^{\mathtt{Valid}}$ *that additionally has access to the validity oracle* `Valid`*, making* q_V *many queries to* `Valid` *,* q_P *many queries to the plaintext checking oracle* 1-PCO *and* q_G *queries to random oracle* G *(the one used to derive randomness), we can construct an* ANO-PCA *adversary* $\mathcal{B}$ *without access to* `Valid` *s.t.*

$$\begin{aligned}
\mathbf{Adv}^{\mathsf{KEM}}_{\mathsf{ANO\text{-}PCA}}(\mathcal{A}^{\mathtt{Valid}}) \leq \mathbf{Adv}^{\mathsf{KEM}}_{\mathsf{ANO\text{-}PCA}}(\mathcal{B}) + q_V \cdot \delta^{\mathsf{PKE}}_{corr} + q_V \cdot 2^{-\gamma} \\
+ 12 q_V (q_V + q_P + q_G) \cdot 2^{-\frac{\gamma}{2}}.
\end{aligned}$$

Lemma 5. We show that ANO-PCA security implies security under the addition of the `Valid` oracle, using a reduction in the QROM. Adversary $\mathcal{B}$ runs $\mathcal{A}^{\mathtt{Valid}}$ and simulates its oracles by forwarding the challenge public keys and the queries to the challenge and plaintext-checking oracles to its ANO-PCA challenger. When $\mathcal{A}^{\mathtt{Valid}}$ outputs a bit, $\mathcal{B}$ forwards it to the ANO-PCA challenger and wins whenever $\mathcal{A}^{\mathtt{Valid}}$ does.

Adversary $\mathcal{B}$ simulates the validity oracle $\texttt{Valid}_{\mathcal{B}}$ and random oracle $\texttt{G}_{\mathcal{B}}$ as shown in Fig. 9. First we note that there are two cases where the simulated oracle behaves differently from the way is defined for an unbounded adversary:

- Case 1: $\texttt{Valid}_{\mathcal{B}}$ returns `False` while `Valid` does not. This case occurs whenever the adversary has found m s.t. $c = \mathsf{PKE.Enc}(pk, m, \texttt{G}(m))$ without querying `G` on m. Using Lemma 1 of [HHM22] and noticing that the simulated `Valid` oracle implements a decapsulation oracle with reject internally, we can bound the probability of this event as $q_V \cdot 2^{-\gamma}$ when PKE is $\gamma-$spread.
- Case 2: $\texttt{Valid}_{\mathcal{B}}$ returns `True` while `Valid` does not. This case corresponds to a correctness error of the KEM: $\mathcal{A}^{\texttt{Valid}}$ has found a ciphertext s.t. $\mathsf{Decap}(sk, c) \neq K$ for $(c, K) \leftarrow\!\$\, \mathsf{Encap}(pk)$. We can bound the probability of this event $q_V \cdot \delta_{corr}^{\mathsf{PKE}}$.

Secondly, the simulated oracle extracts messages from the random oracle that match the ciphertext. To argue about this we again make use of the extractable QROM formalism [DFMS22] as well Lemma 1 of [HHM22].
The simulated $\texttt{Valid}_{\mathcal{B}}$ oracle uses the extraction interface $\texttt{eCO.E}_f$ defined relative to a function $f : X \times Y = T$, where X and Y are the random oracle's domain and co-domain, respectively, and T is some other set. Here, we use the encryption function $\mathsf{PKE.Enc}$ (pk,m,r) as our function f, i.e., we extract messages m s.t. for queried public key pk and ciphertext c, we have that $c = \mathsf{PKE.Enc}(pk, m, \texttt{G}(m))$.
If the adversary has previously made a query that fulfills this predicate, the extraction returns this message m and the $\texttt{Valid}_{\mathcal{B}}$ oracle returns true. In the case that no such query was made, the extraction returns $\perp$ and the $\texttt{Valid}_{\mathcal{B}}$ oracle returns false. Finally, we see using the technique from [HHM22] that the disturbance the online extraction incurs can be bounded as $8\sqrt{2\Gamma(f)/|\mathcal{R}|}$ with

$$\Gamma(f) = \max_t \max_x |\{y \mid R(x, y) = t\}| = 2^{-\gamma} \cdot |\mathcal{R}| \tag{1}$$

since $\texttt{eCO.RO}$ and $\texttt{eCO.E}_f$ almost-commute, and PKE is $\gamma-$spread.
We swap the $\texttt{eCO.RO}$ call that produces m with all calls to $\texttt{eCO.E}$ that happen after the adversary submits c, including the calls inside the validity oracle and the plaintext-checking oracle. Therefore, we can bound the distance between the games:

$$\mathbf{Adv}^{\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{A}^{\texttt{Valid}}) \leq \mathbf{Adv}^{\mathsf{KEM}}_{\mathsf{ANO\text{-}PCA}}(\mathcal{B}) + \underbrace{q_V \cdot \delta_{corr}^{\mathsf{PKE}} + q_V \cdot 2^{-\gamma} + 12 q_V (q_V + q_P + q_G) \cdot 2^{-\frac{\gamma}{2}}}_{=:\varepsilon_{FO}} .$$

Therefore, we conclude that the KEM under consideration here are `Valid`-resistant Anonymous with ε_{FO}. □

Honest robustness of FO-based KEMs. In this section, we used that the explicitly rejecting counterpart of FO-based KEMs has hROB. We now prove this property *if the KEM uses prefix hashing*, i.e., if it includes the public key into the hash when deriving the encryption randomness. In Appendix A, we show how to easily generalize this for FO-based KEMs *without* prefix hashing if the KEM satisfies an additional mild statistical property (weak collision freeness).

Lemma 6 (hROB of FO-based KEMs with prefix hashing)**.** *Let* PKE *be a public-key encryption scheme that is* γ*-spread and for which two honestly generated public keys collide with probability at most* η*, and let* KEM *be the KEM obtained from* PKE *by applying the variant of the explicitly rejecting* $\mathsf{FO}^{\perp}$ *transformation that computes the to-be-used encryption randomness as* $r := \mathsf{G}(pk, m)$ *instead of* $r := \mathsf{G}(m)$*. Then*

$$\mathbf{Adv}^{\mathsf{hROB}}_{\mathsf{KEM}}(\mathcal{A}) \leq 2^{-\gamma} + \eta \ .$$

Lemma 6. Plugging in the concrete algorithms for the FO-based KEM, $(c, .) \leftarrow\$ \mathsf{Encap}(pk_1)$ translates to $c \leftarrow\$ \mathsf{Encrypt}(pk_1, m; \mathtt{G}(pk_1, m))$ for a uniformly chosen message m, and $k := \mathsf{Decap}(sk_2, c)$ differs from $\perp$ if and only if c passes FO's re-encryption check under the public key that is included in sk_2, that is, if $c = c'$, where $c' := \mathsf{Encrypt}(pk_2, m'; \mathtt{G}(pk_2, m'))$ for $m' := \mathsf{Decrypt}(sk_2, c)$.

If $m' \neq m$, k will differ from $\perp$ if and only if encrypting m', a message different from m, accidentally hits c. Since we assume PKE to be γ-spread, for any key pair and any message m', this happens with probability at most $2^{-\gamma}$. (The probability is taken over the randomness chosen in $\mathsf{Encrypt}$, which is the output of random oracle $\mathtt{G}$ on a fresh message and consequently uniform.)

But even if $m' = m$, the two ciphertexts c and c' will still have used different encryption randomness if $pk_1 \neq pk_2$, in which case the bound $2^{-\gamma}$ still applies, and the case $pk_1 = pk_2$ can only happen with probability at most η. □

Lemma 2. With these tools in hand, we can now conclude the proof of Lemma 2: Let KEM be $\mathtt{Valid}$-resistant anonymous, honest robust, $\delta_{\mathsf{KEM}}-$correct; let $\mathcal{R}$ be any reduction that reduces $\mathsf{ANO\text{-}PCA}$ of KEM to BPR security of OCAKE. Then the meta-reduction $\mathcal{M}^{\mathcal{R}}$ can use $\mathcal{R}$ to break $\mathsf{ANO\text{-}PCA}$ of KEM, unless $\mathcal{R}$ causes the program flag to be raised. Firstly, the adversary provided by the meta-reduction is successful according to lemma 4. Secondly, from lemma 5 we conclude that for FO-transformed schemes, the adversary provided by the meta-reduction can be efficiently simulated, unless program occurs. From these two points, we conclude that the meta-reduction $\mathcal{M}$ efficiently transforms the non-programming reduction $\mathcal{R}$ into a successful adversary in the anonymity game:

$$\mathbf{Adv}^{\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{M}^{\mathcal{R}}) = \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{M}^{\mathcal{R}})] = \Pr[\mathrm{G}_{\mathsf{ANO\text{-}PCA}}(\mathcal{R}) \Rightarrow 1] \cdot \Pr[\neg \mathsf{program}]$$

□

Proof of Lemma 3. We build a meta-reduction $\mathcal{M}$ to show that whenever the program flag is raised, the reduction $\mathcal{R}$ instantiates a programming simulator according to SIM. This meta-reduction is defined w.r.t. a simulation challenge $\mathcal{X}$, i.e., a set of public keys; and an adversary $\mathcal{A}_{\mathsf{SIM}}$ that queries the oracles E and D to finally output a value sid, apk. The meta-reduction is said to program whenever at the end we have that for this apk and the session password $pw(sid)$: $\mathsf{D}(pw, apk) \in \mathcal{X}$ according to Definition 12.2. The meta-reduction also makes use of a challenger in the anonymity game since the reduction is only defined with respect to such a challenger, however this challenger can be simulated by the meta-reduction.

Lemma 3. Let KEM be $\mathtt{Valid}$-resistant anonymous, honest robust, $\delta_{\mathsf{KEM}}-$correct; let $\mathcal{R}$ be any reduction that reduces $\mathsf{ANO\text{-}PCA}$ of KEM to BPR security of OCAKE. Then the meta-reduction $\mathcal{M}^{\mathcal{R}}$ using $\mathcal{R}$ instantiates SIM whenever $\mathcal{R}$ causes the program flag to be raised. I.e.,

$$\Pr[(apk, i) \leftarrow \mathcal{A}_{\mathsf{SIM}}{}^{\mathcal{M}(\mathcal{R}.\mathsf{E}_{\mathsf{SIM}}, \mathcal{R}.\mathsf{D}_{\mathsf{SIM}})} : \mathsf{D}_{\mathsf{SIM}}(K[i], apk) \in \mathcal{X}] = \Pr[\mathsf{program}].$$

We analyse the adversary: In the proof of Lemma 2 we showed that the adversary provided by the meta-reduction is successful according to Lemmas 4 and 5, unless $\mathcal{A}_{\mathcal{M}}$ receives a ciphertext c^* (in line 15) that was derived from one of the public keys in $\mathcal{X}$. This ciphertext c^* is either associated with the matching decrypted public key of the session, or an independent key. Therefore, we must have $\mathsf{D}(pw, apk) = pk_0$ or $\mathsf{D}(pw, apk) = pk_1$ for the password pw of the attacked party. Since we chose $apk = apk^*$ this implies $\mathsf{D}(pw, apk^*) \in \mathcal{X}$, exactly as desired. Therefore we conclude that $\mathcal{M}$ implements SIM with failure probabiltiy $\delta_{\mathsf{SIM}} = 1 - \Pr[\mathsf{program}]$. □

Game	Game Description	Quantum?
G_0	Original BPR game.	
G_1	Abort on KEM Key Generation collision.	-
G_2	Abort on collision in key derivation function.	-
G_3	Lazy sampling the ideal cipher	SIM
G_4	Abort on collisions in ideal cipher	SIM
G_5	Abort on colliding tags.	QROM (CO)
G_6	Sample ideal cipher using KEM KGen	SIM
G_7	Do not decrypt honest c (correctness).	-
G_8	Abort on correct password guess (tag).	QROM (eCO)
G_9	NEW: Make eCO extractions online.	QROM (eCO)
G_{10}	Randomize public key (anonymity).	SIM
G_{11}	Randomize pre-key (indistinguishability).	-
G_{12}	NEW: Move eCO extractions to end.	QROM (eCO)
G_{13}	Randomize Tag, Session Key.	QROM (CO)
G_{14}	Randomize Passwords.	-

Figure 10: An overview of the game hops in the proof of OCAKE in the BPR model in the QROM assuming SIM. Game-hops that required changes when adapted to the quantum setting are highlighted in grey. All arguments involving the simulator are labeled SIM, those involving the 'compressed oracle technique' for quantum random oracles are labeled (CO) and those involving the 'extractable' extension to compressed oracles are labeled (eCO).

6 Security of OCAKE in the QROM assuming SIM

To show that the definition of SIM is meaningful, we show that SIM suffices to prove security of OCAKE in the QROM, in the BPR model. Since OCAKE already has a security proof against classical attackers in the ROM and ICM, it suffices to

1. lift all proof steps involving random oracles to the quantum setting, i.e., the QROM,
2. argue *all* proof steps involving the ideal cipher using SIM.

Due to space limitations and the fact that the new game hops are most clearly understood in the context of the full *existing* proof, we postpone the complete technical treatment of this proof (including the relevant security definitions and the pseudocode for every game hop) to Section 6.1. This includes the relevant definitions for the QROM techniques used. We do, however, provide a detailed intuitive explanation of all game hops and the methods used to lift the arguments to the QROM. For readers who are already acquainted with the proof(s) of OCAKE, Fig. 10 gives an overview of the game hops in the original proof and indicates which steps we give lifted arguments for, and the technique used to achieve it.

6.1 Proof of OCAKE under SIM in the QROM

We show that the simulation described in section 4 is sufficient to prove security of PAKE in the BPR model, using a generic KEM, by providing an updated security statement for OCAKE.Security of the OCAKE protocol has been shown in the universal composability framework [BCP+23] as well as in the BPR model [AHH+23]. Both results use the ROM and the ICM. Building on the proof given in [AHH+23], we give a security proof for the OCAKE protocol in the BPR model, using the QROM and assuming the existence of a simulator SIM that effectively replaces the IC.

To lift the classical proof to the quantum setting, all *public* functions – that is, those that can be computed by the attacker – need to be modeled to allow quantum queries [BDF+11].

Therefore, any proof argument that considers hash functions, key derivation functions or block ciphers needs adapting to quantum queries. In this setting, we now give our new security theorem.

Theorem 4 (Security of OCAKE in the QROM from multi-user security of KEM and SIM)**.** *Let* KEM *be a key encapsulation mechanism that is* $(1-\delta_{\mathsf{KEM}})$*-correct, let* KDF*, and* H *be modeled as quantum random oracles with codomains* $\mathcal{K}_{pw}$ *and* $\mathcal{T}\times\mathcal{SK}$*,* BC *be a quantumly-accessible block cipher that we model according to* SIM *with parameters* δ_{SIM} *and* $\varepsilon_{\mathsf{SIM}}$*, and let* $\mathcal{A}$ *be a* BPR *adversary against* OCAKE[KEM, KDF, H, BC]*, issuing at most* n_a *many* ***Send*** *queries (active attacks),* n_p *many* ***Execute*** *queries (passive attacks), and* q_H *many queries to its respective random oracles. Let* $n_s := n_a + n_p$ *be the total number of sessions. Then there exist a multi-user-*IND-CPA *adversary* $\mathcal{B}^{\mathsf{IND}}$ *and a multi-user-*ANO-PCA *adversary* $\mathcal{B}^{\mathsf{ANO}}$ *against* KEM *such that*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{BPR}}_{\mathsf{OCAKE}}(\mathcal{A}) \leq \sum_{i=0}^{14}|\Pr[\mathrm{G}_i(\mathcal{A})\Rightarrow 1]| \leq \frac{n_a}{|\mathcal{D}|} + n_a^2\cdot\eta_{\mathsf{KGen}} + \frac{|\mathcal{D}|^2}{|\mathcal{K}_{pw}|} + \frac{80e^2q_H^3+2}{|\mathcal{T}|}\\ + n_a\cdot\delta_{\mathsf{KEM}} + 4\cdot n_a\cdot(n_s+q_H)\sqrt{2/|\mathcal{T}|} +\\ 2\cdot\left(\varepsilon_{\mathsf{SIM}} + \frac{1}{(1-\delta_{\mathsf{SIM}})}\cdot\mathbf{Adv}^{(n_s,\,n_a+1)\text{-}\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{B}^{\mathsf{ANO}})\right)\\ + 2\cdot\mathbf{Adv}^{(n_s,\,n_a+1)\text{-}\mathsf{IND\text{-}CPA}}_{\mathsf{KEM}}(\mathcal{B}^{\mathsf{IND}}) + 2\cdot\frac{q_H}{|\mathcal{K}|}\end{aligned}$$

and the running time of $\mathcal{B}^{\mathsf{IND}}$*, and* $\mathcal{B}^{\mathsf{ANO}}$ *is about that of* $\mathcal{A}$*.*

To obtain this bound, we update the classical BPR proof for OCAKE [AHH+23] to capturing quantum attackers. We give an overview over the classical proof's game-hops in Table Fig. 10, indicate which game-hops we need to update, the new proof techniques we use to do so, and the resulting loss.

To summarize our changes: firstly, we lift all game-hops that use random oracle properties (games G_5, G_{13}) to the QROM. Secondly, for a concise presentation of how to apply our simulator properties, we fold all game-hops that use IC properties (games G_3,G_4,G_6,G_{10}) into a single game (G_{10}). The resulting upgraded proof is fully adapted to capturing quantum attackers.

Game G_0**: Original** BPR **game.** The BPR oracles are exactly as in the existing BPR proof, i.e., as shown in Fig. 11. H, KDF and IC are quantum-accessible.

Game G_1**: Collisions of KEM key pairs. (Unchanged)** The game aborts whenever there are at least two sessions that sampled the same ephemeral KEM key pair (pk, sk). Let η_{KGen} be the collision probability of KGen. Since games G_0 and G_1 are identical unless a collision occurs, we have $|\mathbf{Adv}_1 - \mathbf{Adv}_0| = \Pr[\mathtt{KDFColl}] \leq n_a^2\cdot\eta_{\mathsf{KGen}}$.

Game G_2**: Abort on KDF Collision.** The game aborts whenever two passwords lead to a collision in the key derivation function that maps passwords to block cipher keys. This bound only depends on the choice of KDF and dictionary $\mathcal{D}$, not on any adversarial input. The resulting bound thus is statistical. Concretely, if KDF is a uniformly random function $f:\mathcal{D}\to\mathcal{K}_{pw}$, the probability that two inputs pw, pw' are mapped to the same output k_{pw} can be upper-bounded by the birthday collision bound: $|\mathbf{Adv}_2 - \mathbf{Adv}_1| \leq \Pr[\mathsf{KDFColl}] \leq \frac{|\mathcal{D}|^2}{|\mathcal{K}_{pw}|}$.

Games G_3 **and** G_4: we fold them into game G_{10}.

Game G_5**: Abort on Tag Collision.**

One of the informal security goals of PAKE proofs is to rule out attacks where an adversary can *test* more than one password in a single online attack. One example of such an attack is the exploitation of a tag collision: assume there exist two passwords pw, pw' s.t. the

Initialisation

for $(P, P') \in \mathcal{P} \times \mathcal{P}$
  $pw \leftarrow\$ \mathrm{PW}()$
  $\mathrm{PWD}[\{P, P'\}] \leftarrow pw$

$\underline{\texttt{Execute}(P, i, P', j)}$

$apk \leftarrow \texttt{Send}^{\mathbf{0}}(P, i, \bot)$
$c, tag \leftarrow \texttt{Send}^{\mathbf{1}}(P', j, apk)$
$\texttt{Send}^{\mathbf{2}}(P, i, (c, tag))$
$\mathrm{MANIP}[\{\mathrm{P},:\}] \leftarrow \texttt{false}$
return (apk, c, tag)

$\underline{\texttt{Corrupt}(P, \mathrm{PWD'})}$

$\mathrm{PWD}_P \leftarrow \mathrm{PWD}[\{P, :\}]$
$\mathrm{CRPT}[\{\mathrm{P},:\}] \leftarrow \texttt{true}$
$\mathrm{PWD}[\{P, :\}] \leftarrow \mathrm{PWD'}[\{P, :\}]$
return PWD_P

$\underline{\texttt{Reveal}(P, i)}$

$\mathrm{RVL}[(P, i)] \leftarrow \texttt{true}$
return $\mathrm{SK}[(P, i)]$

$\underline{\texttt{Test}^b(P, i)}$

$SK_0 \leftarrow \mathrm{K}[(P, i)]$
$SK_1 \xleftarrow{unif} \$\, \mathcal{SK}$
if (CRPT[{P,P'}] and MANIP[(P, i)])
or if (RVL[(P, i)] or RVL[(P', j)])
or if ($\mathrm{K}[(P, i)] = \bot$): **return** $\bot$
else: **return** SK_b

$\underline{\texttt{Send}^{\mathbf{0}}(P, i, msg)}$

$\mathrm{MANIP}[(P, i)] \leftarrow \texttt{true}$
$k_{pw} \leftarrow \mathsf{KDF}(pw)$
$(pk, sk) \leftarrow\$ \mathsf{KGen}$
$apk \leftarrow \mathsf{E}(k_{pw}, pk)$
return apk

$\underline{\texttt{Send}^{\mathbf{1}}(P, i, msg)}$

$\mathrm{MANIP}[(P, i)] \leftarrow \texttt{true}$
$apk \xleftarrow{parse} msg$
$k_{pw} \leftarrow \mathsf{KDF}(pw)$
$pk' \leftarrow \mathsf{D}(k_{pw}, apk)$
$(c, K) \leftarrow\$ \mathsf{Encap}(pk')$
$tag, SK \leftarrow \mathtt{H}(K, pw, apk, pk', c)$
$\mathrm{SK}[(P, i)] \leftarrow SK$
return (c, tag)

$\underline{\texttt{Send}^{\mathbf{2}}(P, i, msg)}$

$\mathrm{MANIP}[(P, i)] \leftarrow \texttt{true}$
$c, tag \xleftarrow{parse} msg$
$K' \leftarrow \mathsf{Decap}(sk, c)$
$tag', SK' \leftarrow \mathtt{H}(K'; pw, apk, pk, c)$
if $tag' = tag$
  $\mathrm{SK}[(P, i)] \leftarrow SK$

Figure 11: The oracles in the security game for OCAKE. PWD is the dictionary of the parties' passwords and CRPT, MANIP and RVL indicate the corruption status of a session. Password generation in the initialization phase (**Initialization**) is modeled using the long-lived key generator PW.

responder tag tag computed from them for a given transcript is the same. Then the attacker could win the game by submitting tag without having uniquely identified the right password.

Therefore, G_5 aborts whenever the attacker has found such a collision in $\mathtt{H}$. To argue about the distance between G_5 and G_4, we use the compressed oracle technique [CFHL21].

Let $\mathcal{A}$ be the adversary in the PAKE game. We define an adversary $\mathcal{B}$ in the collision-finding game that runs $\mathcal{A}$ and is successful if it outputs x, x' s.t. $\mathtt{H}(x) = \mathtt{H}(x')$ at the end of the game.

$\mathcal{B}$ simulates $\mathcal{A}$'s PAKE and random oracle queries. First, $\mathcal{B}$ replaces the random oracle with a compressed random oracle. The compressed random oracle allows $\mathcal{B}$ to record $\mathcal{A}$'s queries in a database.

Then, at the end of the game, $\mathcal{B}$ measures the database for collisions. We denote the partitioning of the output of the hash function into tag and session key as $H(x) = tag||SK =: H(x)[0]||H(x)[1]$. Then we can define as R the relation that describes a collision in the tag: for distinct x, x' and arbitrary sk, sk', we have $H(x)[0]||sk = H(x')[0]||sk'$.

Initialisation

01 for $(P, P') \in \mathcal{P} \times \mathcal{P}$
02 $pw \leftarrow\$ \mathrm{PW}()$
03 $\mathrm{PWD}[\{P, P'\}] \leftarrow pw$
04 $PK \leftarrow \emptyset$

$\underline{\mathtt{Send}^{\mathbf{0}}(P, i, msg)}$

05 $\mathrm{MANIP}[(P, i)] \leftarrow \mathtt{true}$
06 $k_{pw} \leftarrow \mathsf{KDF}(pw)$
07 $(pk, sk) \leftarrow\$ \mathsf{KGen}$
08 if $pk \in PK$ abort
09 $PK = PK \cup \{pk\}$
10 $apk \leftarrow \mathsf{E}(k_{pw}, pk)$
11 **return** apk

Figure 12: The game change for game G_1 Abort on KGen Collision.

Informally, this is equivalent to discarding the part that represents the session key, or projecting onto the space of the tag. Then we have $R \subseteq \mathcal{X}^l \times (\mathcal{T} \times \mathcal{SK})^l$ for $l = 2$. Then, by Corollary 4.2 [CFHL21, Corollary 4.2], we have that for success probability in the original game p and success probability in the game using the compressed random oracle p', we have that $\sqrt{p} \leq \sqrt{p'} + \sqrt{\frac{2}{|\mathcal{T}|}}$.

We now argue about the success probability p', closely following the reasoning in section 2.3 of [CFHL21], (*Finding a Collision (with Parallel Queries)*). Let $\mathsf{SZ}_{\leq(s-1)}$ be a database of size at most $s-1$. We set $k = 1$ as we do not model parallel queries, so we have using the transition probability from a database that does not contain a collision to one that *does*

$$\sqrt{p'} \leq \sum_{s=1}^{q_{\mathtt{H}}} [\![\mathsf{SZ}_{\leq(s-1)} \backslash \mathsf{CL} \to \mathsf{CL}]\!] \leq \sum_{s=1}^{q_{\mathtt{H}}} 2e\sqrt{10\frac{q_{\mathtt{H}}}{|\mathcal{T}|}} \leq 2q_{\mathtt{H}}e\sqrt{10\frac{q_{\mathtt{H}}}{|\mathcal{T}|}}$$

where e is Euler's number. Therefore we conclude

$$p' \leq \frac{40q_{\mathtt{H}}^3 e^2}{|\mathcal{T}|}.$$

From $\sqrt{p} \leq \sqrt{p'} + \sqrt{\frac{2}{|\mathcal{T}|}}$ we then see by squaring that

$$p \leq p' + \frac{2}{|\mathcal{T}|} + 2\sqrt{\frac{2p'}{|\mathcal{T}|}} \leq \frac{e^2(40q_{\mathtt{H}}^3 + \sqrt{5}q_{\mathtt{H}}^{3/2}) + 2}{|\mathcal{T}|} \leq \frac{80e^2q_{\mathtt{H}}^3 + 2}{|\mathcal{T}|}.$$

Finally, this allows us to bound the distance: $|\mathbf{Adv}_5 - \mathbf{Adv}_4| \leq p$.

Game G_6 is also folded into game G_{10}.

Game G_7: Correctness. (Unchanged)

In game G_7, whenever there is a flow 2 query where the message was honestly generated by a matching session, we do not decapsulate to obtain the pre-key. Instead, if the message was generated by a matching session, we use the pre-key that was encapsulated by that instance. Adversarially generated messages as well as ones that are forwarded from a non-matching session are decapsulated as before. As in the classical proof, the distance to the previous game can be bounded by $|\mathbf{Adv}_7 - \mathbf{Adv}_6| \leq n_a \cdot \delta_{\mathsf{KEM}}$.

Game G_8: Abort on dishonest tags using correct pw.

We now abort whenever the adversary actively interacts with a session, meaning the session is not executed between two honest parties, and submits a tag that was generated using the correct password. This models the case that the attacker successfully guessed the password and used it to attack an honest part, which can happen both for

1. initiators, in which case the attacker sent a responder tag *tag* that uses the correct password, and for
2. responders, in which case the attacker sent an *apk* that uses the correct password.

Initialisation

01 for $(P, P') \in \mathcal{P} \times \mathcal{P}$
02 $pw \leftarrow\$ \mathrm{PW}()$
03 $\mathrm{PWD}[\{P, P'\}] \leftarrow pw$
04 $PK \leftarrow \emptyset$
05 $\mathcal{L}_{pw} \leftarrow \emptyset$
06 $C \leftarrow \emptyset$
07 $TAG \leftarrow \emptyset$

$\mathtt{Send}^{\mathtt{2}}(P, i, msg)$

08 $\mathrm{MANIP}[(P, i)] \leftarrow \mathtt{true}$
09 $c, tag \xleftarrow{parse} msg$
10 $K' \leftarrow K$
11 $tag', SK' \leftarrow \mathtt{H}(K', pw, apk, pk, c)$
12 if $tag' = tag$
13 if not $tag \in TAG$ and not $\mathrm{CRPT}[\{\mathrm{P,P'}\}]$ abort
14 $\mathrm{SK}[(P, i)] \leftarrow SK$

Figure 13: The game change for game G_8 Abort on correct password. Abort if we receive a tag that is correct unless it is honest or the password was corrupted.

Here we focus on case 1, since the other case is handled by simulation SIM (see 4). We need to adapt previous classical reasoning about this change to capture that oracle H now is quantum-accessible – previous reasoning exploited that game and reductions can observe the issued queries to H, thus being able to immediately notice that tag was generated with the correct password. As a quantum counterpart to this approach, we use the extractable QROM formalism [DFMS22]. This models quantum-accessible random oracles like H as an extractable compressed random oracle eCO that has two interfaces, random oracle interface eCO.RO and an extraction interface $\mathtt{eCO.E}_f$ that is defined relative to a function $f : X \times Y = T$, where X and Y are the random oracle's domain and co-domain, respectively, and T is some other set. For our purposes, we identify X with the domain of H, so $X := \mathcal{K} \times \mathcal{D} \times \mathcal{PK}^2 \times \mathcal{C}$, and $Y := T := \mathcal{T} \times \mathcal{SK}$, the tag and session key space. Since we want to isolate the first part of the output, we choose the function f defined by the projection onto the tag space $f(input, y = (tag, K)) = tag$. Extraction interface $\mathtt{eCO.E}_f$ takes as input a classical value $t \in T$. It does a quantum analogue to going through the random oracle queries and returning an x such that $f(x, \mathtt{H}(x)) = t$: it performs suitable measurements that collapse the oracle database, just enough so that the classical procedure would yield one particular outcome x for all parts of the superposition. For our choice of f, $\mathtt{eCO.E}_f(tag)$ simply returns a random oracle pre-image of $tag||sk$ for any sk. Equipped with this extraction interface, the game waits until the attacker has finished and then calls $\mathtt{eCO.E}_f$ on all 'dishonest' tries, i.e., all tags tag that were received by an honest initiator whose password was not corrupted, without tag having been computed during a preceding honest execution of the respective responder. The game aborts if there exists a dishonest tag such that $\mathtt{eCO.E}_f(tag)$ returns a pre-image (K, pw, apk, pk', c) such that (K, apk, pk', c) matches the values which the initiator computed during the session to which the tag belongs. Since we already ruled out random oracle collisions, this in particular means that pw is the correct password. We denote this event by corrPW. We note that the extraction queries themselves do not change $\mathcal{A}$'s view since they are performed only after $\mathcal{A}$ finished [DFMS22, Th. 4.3, item 1].
We can now conclude that $|\mathbf{Adv}_8 - \mathbf{Adv}_7| \leq \Pr[\mathsf{corrPW}]_{\mathrm{G}_8}$. Like in previous (classical) proofs, the change of probability for corrPW can be traced through the subsequent games G_8-G_{14}, by building anonymity and indistinguishability reductions that define their output based on corrPW.

Game G_9: Notice correct pw guess already during game.

In this game, we move the identification of corrPW into the game: instead of performing the extraction queries determining corrPW after running the attacker, we perform them already during the session runs. We perform them whenever the attacker sends a tag tag to an honest initiator whose password was not corrupted, without tag having been computed during a preceding honest execution of the respective responder. This change

is in preparation for the following two game hops, where the reductions use corrPW to detect certain edge-cases of adversarial input that need to be addressed during the game.

To argue that this introduces little change, we use [DFMS22, Th. 4.3]: by properties 2.b and 2.c of Th. 4.3, any two subsequent queries to $\mathtt{eCO.RO}$ and $\mathtt{eCO.E}_f$ $8\sqrt{2\Gamma(f)/|\mathcal{K}| \cdot |\mathcal{T}|}$-almost-commute, where $\Gamma(f) = \max_t \max_x |\{y \mid f(x,y) = t\}|$, which for our choice $f(x,(tag,K)) = tag$ equals $|\mathcal{K}|$. This means that we can commute the extraction queries into the game to right after when the respective Send query was received. We need to commute at most $n_a \cdot (n_s + q_{\mathtt{H}}H)$ many times until the extractions are at the right place, thus

$$|\mathbf{Adv}_{9} - \mathbf{Adv}_{8}| = |\Pr[\mathsf{corrPW}_{\mathrm{G}_9}] - \Pr[\mathsf{corrPW}_{\mathrm{G}_8}]| \leq n_a \cdot (n_s + q_{\mathtt{H}})\sqrt{2/|\mathcal{T}|} \ .$$

Game G_{10}: Randomize public-key pk.

In this game, we randomize the public key that is used to generate the responder's ciphertext. We begin by recalling the setting: to break the link between the first protocol message (apk) and the second message (c, tag), we need to argue that the ciphertext c does not reveal which public key was used to create it. More precisely, we swap the original public key used to derive c for a freshly sampled, uniformly random one, and then bound the distinguishing advantage between the resulting games by the ANO-PCA security of KEM.

We build a reduction that simulates the PAKE game for an adversary $\mathcal{A}$ which distinguishes the BPR game G_{10} from G_9, and then leverages $\mathcal{A}$ to solve its own ANO-PCA challenge. Observe first that sessions subjected only to passive attacks (i.e. `Execute` queries) or sessions in which the initiator is honest can always be simulated without programming the ideal cipher: honest parties simply substitute the challenge public keys for their own honest keys.

The more delicate situation arises when the responder is honest but the initiator is not: here the initiator provides an *adversarially selected* encrypted public key apk, which the honest responder decrypts to obtain a public key.

To transition to game G_{10}, in which ciphertexts c are independent of the actual public keys (or, more precisely, the actual password), the reduction reprograms decrypted public keys to challenge public keys in the ANO-PCA game. We now demonstrate this proof step relying solely on reprogramming within the capabilities of SIM as described above.

As a first step, the reduction responds to all E and D queries using the corresponding oracles provided by SIM. This change to SIM is not perfectly indistinguishable; hence we obtain $\mathbf{Adv}_{10,1} \leq \varepsilon_{\mathsf{SIM}}$, where $\varepsilon_{\mathsf{SIM}}$ is defined via SIM.

The challenge set $\mathcal{X}$ used to initialize SIM is set to the first collection of public keys $\{pk_{0,0}, pk_{0,1}, \ldots pk_{0,n_a}\}$ that the anonymity challenger provides. This set corresponds to the "real" public keys, while the set $\{pk_{1,0}, pk_{1,1}, \ldots pk_{1,n_a}\}$ consists of the independent keys used *after* the reduction. For the $n_a$actively attacked sessions, the definition of SIM (Definition 13, condition 2) guarantees that the decryption of each apk_i that the adversary queries the `Send` oracle on lies in the challenge set $\mathcal{X}$ with probability at least $1 - \delta_{\mathsf{SIM}}$:

$$\Pr[(apk_i, sid_i) \leftarrow \mathcal{A}_i(st)^{\mathsf{SIM}(\mathcal{X})} : \mathsf{D}_{\mathsf{SIM}}(k[sid_i], apk_i) \in \mathcal{X}] \geq (1 - \delta_{\mathsf{SIM}}).$$

Upon receiving input apk_i, the reduction decrypts apk_i to obtain a challenge public key $pk_{0,j} \in \mathcal{X}$. It then requests from its challenger a fresh anonymity challenge c associated with that public key and incorporates this value into its reply. In this manner, in every session the adversary is given a ciphertext that corresponds to either pk_0 or pk_1, precisely as in the two games.

We note that since the reduction simulates all of $\mathcal{A}$'s oracles there is an edge case to consider: the case where $\mathcal{A}$ *corrupts* a party after it has queried the $\mathtt{Send}^0$ and $\mathtt{Send}^1$,

$\underline{\texttt{Send}^1(P, i, msg)}$

01 $\text{MANIP}[(P, i)] \leftarrow \texttt{true}$
02 $apk \xleftarrow{parse} msg$
03 $k_{pw} \leftarrow \texttt{KDF}(pw)$
04 $pk' \leftarrow \mathsf{D}(k_{pw}, apk)$ $pk_{\$} \leftarrow \mathcal{PK}$
05 $(c, K) \leftarrow\!\$\ \mathsf{Encap}(pk')$ $(c, K) \leftarrow\!\$\ \mathsf{Encap}(pk'_{\$})$
06 $C \leftarrow C \cup c$
07 $tag, SK \leftarrow \texttt{H}(K, pw, apk, pk', c)$
08 $TAG \leftarrow TAG \cup tag$
09 $\text{SK}[(P, i)] \leftarrow SK$
10 **return** (c, tag)

Figure 14: The game change for game G_{10} Anonymity.

but before it queries $\texttt{Send}^2$ for matching sessions while forwarding messages. In this case, the oracle $\texttt{Send}^2$ uses a challenge public key pk_0 that the attacker now *knows* due to the corruption. However, the reduction does not have the secret key associated with the public key, and it cannot decapsulate the ciphertext and recompute the tag. Therefore, when the oracle decides whether or not to reject an adversarially generated (c, tag), the reduction has use the extraction interface of the extractable random oracle $\texttt{eCO}$ to determine the pre-key that was used by the adversary. Then, the reduction can query its plaintext-checking oracle to decide if the message should be rejected or not. Since we switched to the extractable oracle in a previous game, this does not incur any additional loss here, however this edge case is the reason the two commuting bounds are needed. Finally, the reduction outputs 0 whenever $\mathcal{A}$ wins, and 1 else. Therefore, the reduction wins whenever $\mathcal{A}$ wins and SIM successfully programmed for all its queries and we have

$$adv10, 2 \leq \frac{1}{(1 - \delta_{\mathsf{SIM}})} \cdot \mathbf{Adv}^{\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}.$$

We then arrive at the bound

$$\begin{aligned} |\Pr[\mathsf{corrPW}_{\text{G}_{10}}] - \Pr[\mathsf{corrPW}_{\text{G}_9}]| &= \mathbf{Adv}_{10} \\ &\leq \mathbf{Adv}_{10,1} + \mathbf{Adv}_{10,2} \\ &\leq \varepsilon_{\mathsf{SIM}} + \frac{1}{(1 - \delta_{\mathsf{SIM}})} \cdot \mathbf{Adv}^{(n_s, n_a + 1) - \mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{B}^{\mathsf{ANO}}). \end{aligned}$$

Game G_{11}: Randomize pre-key K. (Unchanged)

For all queries to the $\texttt{Send}$ or $\texttt{Execute}$ oracles where flag $\texttt{trivGuess}$ is not raised before the query, we now randomize the pre-key K that is used to derive the final session key and the responder tag. For more details, see the pseudo-code in Figure 15. This change makes the pre-key independent of the ciphertext and the password for all fresh sessions. As previously, the change in success probability can be bounded using the indistinguishability advantage of KEM:

$$|\mathbf{Adv}_{11} - \mathbf{Adv}_{10}| \leq \mathbf{Adv}^{(n_s, n_a + 1) - \mathsf{IND\text{-}CPA}}_{\mathsf{KEM}}(\mathcal{B}^{\mathsf{IND}}).$$

Game G_{12}: Move dishonest tag identification back to end of game.

In this game, we switch back to offline extraction, i.e., we only perform the extraction queries determining if a tag was valid after running the attacker. This is done in preparation

$\underline{\texttt{Send}^{\mathbf{1}}(P, i, msg)}$

01 $\text{MANIP}[(P,i)] \leftarrow \texttt{true}$
02 $apk \xleftarrow{parse} msg$
03 $k_{pw} \leftarrow \texttt{KDF}(pw)$
04 $pk_{\$} \leftarrow \mathcal{PK}$
05 $(c, K) \leftarrow\!\$\ \mathsf{Encap}(pk'_{\$})$
06 $K_{\$} \leftarrow\!\$\ \mathcal{K}$
07 $C \leftarrow C \cup c$
08 $tag, SK \leftarrow \texttt{H}(K, pw, apk, pk', c)$ $tag, SK \leftarrow \texttt{H}(K_{\$}, pw, apk, pk', c)$
09 $TAG \leftarrow TAG \cup tag$
10 $\text{SK}[(P,i)] \leftarrow SK$
11 **return** (c, tag)

Figure 15: The game change for game G_{11} Indistinguishability.

for the next step that relies on measuring the compressed database at the end of the game. With the same reasoning as for game G_9, we find

$$|\Pr[\mathsf{corrPW}_{\text{G}_{12}}] - \Pr[\mathsf{corrPW}_{\text{G}_{11}}]| = |\mathbf{Adv}_{12} - \mathbf{Adv}_{11}| \leq n_a \cdot (n_s + q_{\texttt{H}})\sqrt{2/|\mathcal{T}|}.$$

Game G_{13}: Randomize Tag & Session Key.

In this step of the proof, we randomize the responder tag tag and the session key SK, i.e., we simply sample random values from $\mathcal{T}$ and $\mathcal{SK}$, and return them as tag and session key. Recall, that in Game G_{12} the tag and session key are computed as $(tag, SK) \leftarrow \texttt{H}(K_{\$}; pw, apk, pk', c)$ where $K_{\$}$ is a fresh uniformly random sample from $\mathcal{K}$ that is not used anywhere else in the protocol. Hence, the values in both games follow the same marginal distribution. However, in Game G_{12} they are consistent with the random oracle, while in Game G_{13} they are not. Intuitively, an adversary can only notice this inconsistency in Game G_{13} if it ever learned the value of $\texttt{H}$ at position $x^* = (K_{\$}; pw, apk, pk', c)$. Given that we simulate $\texttt{H}$ using the compressed oracle technique, this means that the difference between this and the last game can be bounded by the probability that the final database D of the compressed oracle in Game G_{13} contains a value for x^* when measured in the end, i.e., $|\mathbf{Adv}_{13} - \mathbf{Adv}_{12}| \leq \Pr[D(x^*) \neq \bot]$. It remains to quantify $\Pr[D(x^*) \neq \bot]$. For this, note that in Game G_{13}, $K_{\$}$ is not used anywhere in the protocol anymore and therefore the whole view of the adversary is independent of $K_{\$}$. For that reason, we can delay the sampling of $K_{\$}$ until after we measured the database D. The database D has $q_{\texttt{H}}$ entries. The probability that one of them is of the form $(K_{\$}; pw, apk, pk', c)$ for any value of $K_{\$}$ is therefore upper bounded by $1/|\mathcal{K}|$ and therefore

$$|\mathbf{Adv}_{13} - \mathbf{Adv}_{12}| \leq \Pr[D(x^*) \neq \bot] \leq \frac{q_{\texttt{H}}}{|\mathcal{K}|}.$$

Game G_{14}: Randomize Passwords.

It remains to upper-bound corrPW. In this game, the attacker's view is completely independent of the chosen passwords, up to corrupted ones. This means that we can replace all non-corrupted passwords with fresh ones at the end, after running the attacker, and defining event corrPW with respect to the resampled ones. We note that there exists exactly one pre-image (K, pw, apk, pk', c) per tag that could trigger corrPW, and that the simulation SIM does not raise this flag as it did in the previous proof. We can now bound the probability of corrPW in game G_8, using the number of send queries and the password distribution. Assuming a uniform distribution on a password dictionary of size $|\mathcal{D}|$, and

$\underline{\texttt{Send}^{\mathbf{1}}(P, i, msg)}$

01 $\text{MANIP}[(P,i)] \leftarrow \texttt{true}$
02 $apk \xleftarrow{parse} msg$
03 $k_{pw} \leftarrow \texttt{KDF}(pw)$
04 $pk_{\$} \leftarrow \mathcal{PK}$
05 $(c, K) \leftarrow\!\!\$\ \mathsf{Encap}(pk'_{\$})$
06 $C \leftarrow C \cup c$
07 $K_{\$} \leftarrow\!\!\$\ \mathcal{K}$
08 $tag, SK \leftarrow \texttt{H}(K_{\$}, pw, apk, pk', c)$
09 $TAG \leftarrow TAG \cup tag$
10 $tag_{\$} \leftarrow\!\!\$\ \mathcal{T}, SK_{\$} \leftarrow\!\!\$\ \mathcal{SK}$
11 $\text{SK}[(P,i)] \leftarrow SK$ $\text{SK}[(P,i)] \leftarrow SK_{\$}$
12 **return** (c, tag) **return** $(c, tag_{\$})$

Figure 16: The game change for game G_{13} Randomize Tag and Session key.

upper-bounding $\mathcal{A}$'s number of send queries by n_a, $\Pr[\mathsf{corrPW}_{\text{G}_{14}}] \leq \frac{n_a}{|\mathcal{D}|}$ And therefore $\Pr[\mathsf{corrPW}_{\text{G}_8}] \leq \sum_{i=8}^{14} \Pr[\mathsf{corrPW}_{\text{G}_i}]$. Since the passwords and session keys are completely random from the adversary's view, we have that $|\Pr[\text{G}_{14}(\mathcal{A}) \Rightarrow 1]| = \frac{1}{2}$. We can finally sum up the terms and state the bound:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{BPR}}_{\mathsf{OCAKE}}(\mathcal{A}) \leq \sum_{i=0}^{14} |\Pr[\text{G}_i(\mathcal{A}) \Rightarrow 1]| \leq \frac{n_a}{|\mathcal{D}|} + n_a^2 \cdot \eta_{\mathsf{KGen}} + \frac{|\mathcal{D}|^2}{|\mathcal{K}_{pw}|} + \frac{80e^2 q_{\texttt{H}}^3 + 2}{|\mathcal{T}|} \\ + n_a \cdot \delta_{\mathsf{KEM}} + 4 \cdot n_a \cdot (n_s + q_{\texttt{H}})\sqrt{2/|\mathcal{T}|} + \\ 2 \cdot \left(\varepsilon_{\mathsf{SIM}} + \frac{1}{(1-\delta_{\mathsf{SIM}})} \cdot \mathbf{Adv}^{(n_s, n_a+1)-\mathsf{ANO\text{-}PCA}}_{\mathsf{KEM}}(\mathcal{B}^{\mathsf{ANO}}) \right) \\ + 2 \cdot \mathbf{Adv}^{(n_s, n_a+1)-\mathsf{IND\text{-}CPA}}_{\mathsf{KEM}}(\mathcal{B}^{\mathsf{IND}}) + 2 \cdot \frac{q_{\texttt{H}}}{|\mathcal{K}|}\end{aligned}$$

References

[ABJ25] Afonso Arriaga, Manuel Barbosa, and Stanislaw Jarecki. NoIC: PAKE from KEM without ideal ciphers. Cryptology ePrint Archive, Paper 2025/231, 2025. URL: https://eprint.iacr.org/2025/231.

[ABMS26] Gorjan Alagic, Chen Bai, Christian Majenz, and Kaiyan Shi. Post-quantum security of block cipher constructions, 2026. URL: https://arxiv.org/abs/2510.08725, arXiv:2510.08725.

[ACMT25] Gorjan Alagic, Joseph Carolan, Christian Majenz, and Saliha Tokat. The sponge is quantum indifferentiable. Cryptology ePrint Archive, Paper 2025/731, 2025. URL: https://eprint.iacr.org/2025/731.

[AHH+23] Nouri Alnahawi, Kathrin Hövelmanns, Andreas Hülsing, Silvia Ritsch, and Alexander Wiesmaier. Towards post-quantum secure pake-a tight security proof for ocake in the bpr model. *Cryptology ePrint Archive*, 2023.

[AHHR24] Nouri Alnahawi, Kathrin Hövelmanns, Andreas Hülsing, and Silvia Ritsch. Towards post-quantum secure PAKE - A tight security proof for OCAKE in the BPR model. In Markulf Kohlweiss, Roberto Di Pietro, and Alastair R.

Beresford, editors, *CANS 2024: 23rd International Conference on Cryptology and Network Security, Part II*, volume 14906 of *Lecture Notes in Computer Science*, pages 191–212, Cambridge, UK, September 24–27, 2024. Springer, Singapore, Singapore. doi:10.1007/978-981-97-8016-7_9.

[AHMW25] Nouri Alnahawi, David Haas, Erik Mauß, and Alexander Wiesmaier. SoK: PQC PAKEs - design, security and performance. Cryptology ePrint Archive, Report 2025/119, 2025. URL: https://eprint.iacr.org/2025/119.

[BCP+23] Hugo Beguinet, Céline Chevalier, David Pointcheval, Thomas Ricosset, and Mélissa Rossi. GeT a CAKE: Generic transformations from key encaspulation mechanisms to password authenticated key exchanges. In Mehdi Tibouchi and Xiaofeng Wang, editors, *ACNS 2023: 21st International Conference on Applied Cryptography and Network Security, Part II*, volume 13906 of *Lecture Notes in Computer Science*, pages 516–538, Kyoto, Japan, June 19–22, 2023. Springer, Cham, Switzerland. doi:10.1007/978-3-031-33491-7_19.

[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, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 41–69, Seoul, South Korea, December 4–8, 2011. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-25385-0_3.

[Bla06] John Black. The ideal-cipher model, revisited: An uninstantiable blockcipher-based hash function. In Matthew J. B. Robshaw, editor, *Fast Software Encryption – FSE 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 328–340, Graz, Austria, March 15–17, 2006. Springer Berlin Heidelberg, Germany. doi:10.1007/11799313_21.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155, Bruges, Belgium, May 14–18, 2000. Springer Berlin Heidelberg, Germany. doi:10.1007/3-540-45539-6_11.

[Car25] Joseph Carolan. Compressed permutation oracles. Cryptology ePrint Archive, Report 2025/1734, 2025. URL: https://eprint.iacr.org/2025/1734.

[CFHL21] Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 598–629, Zagreb, Croatia, October 17–21, 2021. Springer, Cham, Switzerland. doi:10.1007/978-3-030-77886-6_21.

[CHK+05] Ran Canetti, Shai Halevi, Jonathan Katz, Yehuda Lindell, and Philip D. MacKenzie. Universally composable password-based key exchange. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 404–421, Aarhus, Denmark, May 22–26, 2005. Springer Berlin Heidelberg, Germany. doi:10.1007/11426639_24.

[CHL+25] Alexandru Cojocaru, Minki Hhan, Qipeng Liu, Takashi Yamakawa, and Aaram Yun. Quantum lifting for invertible permutations and ideal ciphers. Cryptology ePrint Archive, Paper 2025/738, 2025. URL: https://eprint.iacr.org/2025/738.

[CMSZ19] Jan Czajkowski, Christian Majenz, Christian Schaffner, and Sebastian Zur. Quantum lazy sampling and game-playing proofs for quantum indifferentiability. Cryptology ePrint Archive, Paper 2019/428, 2019. URL: https://eprint.iacr.org/2019/428.

[CP25] Joseph Carolan and Alexander Poremba. Quantum one-wayness of the single-round sponge with invertible permutations, 2025. URL: https://arxiv.org/abs/2403.04740, arXiv:2403.04740.

[DAL+17] Jintai Ding, Saed Alsayigh, Jean Lancrenon, Saraswathy Rv, and Michael Snook. Provably secure password authenticated key exchange based on rlwe for the post-quantum world. In *Cryptographers' Track at the RSA conference*, pages 183–204. Springer, 2017.

[DFMS22] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Online-extractability in the quantum random-oracle model. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 677–706, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland. doi:10.1007/978-3-031-07082-2_24.

[DKBY24] Vivek Dabra, Saru Kumari, Anju Bala, and Sonam Yadav. Sl3pake: Simple lattice-based three-party password authenticated key exchange for post-quantum world. *Journal of Information Security and Applications*, 84:103826, 2024.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Annual international cryptology conference*, pages 537–554. Springer, 1999.

[GMP22] Paul Grubbs, Varun Maram, and Kenneth G. Paterson. Anonymous, robust post-quantum public key encryption. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 402–432, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland. doi:10.1007/978-3-031-07082-2_15.

[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: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 341–371, Baltimore, MD, USA, November 12–15, 2017. Springer, Cham, Switzerland. doi:10.1007/978-3-319-70500-2_12.

[HHM22] Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Failing gracefully: Decryption failures and the Fujisaki-Okamoto transform. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part IV*, volume 13794 of *Lecture Notes in Computer Science*, pages 414–443, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland. doi:10.1007/978-3-031-22972-5_15.

[HHM+26a] Yu-Hsuan Huang, Andreas Hülsing, Varun Maram, Silvia Ritsch, and Abishanka Saha. Feistel tools: Reprogramming and query-recording for qrps. *Cryptology ePrint Archive*, 2026.

[HHM+26b] Yu-Hsuan Huang, Andreas Hülsing, Varun Maram, Silvia Ritsch, and Abishanka Saha. Feistel tools: Reprogramming and query-recording for QRPs. Cryptology ePrint Archive, Paper 2026/146, 2026. URL: https://eprint.iacr.org/2026/146.

[HM24] Kathrin Hövelmanns and Christian Majenz. A note on failing gracefully: Completing the picture for explicitly rejecting Fujisaki-Okamoto transforms using worst-case correctness. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part II*, pages 245–265, Oxford, UK, June 12–14, 2024. Springer, Cham, Switzerland. doi:10.1007/978-3-031-62746-0_11.

[Hos25] Akinori Hosoyamada. Post-quantum security of keyed sponge-based constructions through a modular approach. Cryptology ePrint Archive, Paper 2025/1059, 2025. URL: https://eprint.iacr.org/2025/1059.

[LLH24] You Lyu, Shengli Liu, and Shuai Han. Universal composable password authenticated key exchange for the post-quantum world. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 120–150. Springer, 2024.

[MMW24] Christian Majenz, Giulio Malavolta, and Michael Walter. Permutation superposition oracles for quantum query lower bounds. Cryptology ePrint Archive, Paper 2024/1140, 2024. URL: https://eprint.iacr.org/2024/1140.

[MMW25] Christian Majenz, Giulio Malavolta, and Michael Walter. Permutation superposition oracles for quantum query lower bounds. In Michal Koucký and Nikhil Bansal, editors, *57th Annual ACM Symposium on Theory of Computing*, pages 1508–1519, Prague, Czechia, June 23–27, 2025. ACM Press. doi:10.1145/3717823.3718266.

[MX23] Varun Maram and Keita Xagawa. Post-quantum anonymity of Kyber. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023: 26th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 13940 of *Lecture Notes in Computer Science*, pages 3–35, Atlanta, GA, USA, May 7–10, 2023. Springer, Cham, Switzerland. doi:10.1007/978-3-031-31368-4_1.

[PZ23] Jiaxin Pan and Runzhi Zeng. A generic construction of tightly secure password-based authenticated key exchange. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023, Part VIII*, volume 14445 of *Lecture Notes in Computer Science*, pages 143–175, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore. doi:10.1007/978-981-99-8742-9_5.

[SA23] Kübra Seyhan and Sedat Akleylek. A new password-authenticated module learning with rounding-based key exchange protocol: Saber. pake. *The Journal of Supercomputing*, 79(16):17859–17896, 2023.

[SH19] Vladimir Soukharev and Basil Hess. Pqdh: a quantum-safe replacement for diffie-hellman based on sidh. *Cryptology ePrint Archive*, 2019.

[TY19] Shintaro Terada and Kazuki Yoneyama. Password-based authenticated key exchange from standard isogeny assumptions. In *Provable Security: 13th International Conference, ProvSec 2019, Cairns, QLD, Australia, October 1–4, 2019, Proceedings 13*, pages 41–56. Springer, 2019.

[Unr21] Dominique Unruh. Compressed permutation oracles (and the collision-resistance of sponge/SHA3). Cryptology ePrint Archive, Paper 2021/062, 2021. URL: https://eprint.iacr.org/2021/062.

[Unr23a] Dominique Unruh. Towards compressed permutation oracles. Cryptology ePrint Archive, Paper 2023/770, 2023. URL: https://eprint.iacr.org/2023/770.

[Unr23b] Dominique Unruh. Towards compressed permutation oracles. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023, Part IV*, volume 14441 of *Lecture Notes in Computer Science*, pages 369–400, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore. doi:10.1007/978-981-99-8730-6_12.

[Xag22] Keita Xagawa. Anonymity of NIST PQC round 3 KEMs. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 551–581, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland. doi:10.1007/978-3-031-07082-2_20.

[Zha19] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 239–268, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland. doi:10.1007/978-3-030-26951-7_9.

A Honest robustness of FO-KEMs without prefix hashing

In Section 5, we showed our simulator necessity result for an important class of KEMs, KEMs obtained via the full FO-transform. For this result, we needed that the explicitly rejecting counterpart of these KEMs has hROB. We showed this property (see Lemma 6) for such KEMs *if they use prefix hashing*, i.e., if they include the public key into their hashing (at least when deriving the encryption randomness). To broaden the result, we now argue hROB of FO-based KEMs *without* prefix hashing – with Lemma 7 below, we relate hROB of any such FO-based KEM to properties of the KEM's underlying PKE scheme, γ-spreadness and a property we call weak collision freeness. Weak collision freeness captures that ciphertexts generated under one key pair will not likely decrypt to their originating plaintext under another key pair, where all values were sampled honestly. We first formalize weak collision freeness with Definition 15 and then give Lemma 7.

Definition 15 (Weak collision freeness)**.** Let PKE be a public-key encryption scheme. We say that PKE satisfies ρ-weak collision freeness iff

$$\Pr\left[m' = m \;\middle|\; \begin{array}{l}(pk_1,.),(.,sk_2) \leftarrow\!\$\ \mathsf{KGen}() \\ m \leftarrow\!\$\ \mathcal{M} \\ c \leftarrow\!\$\ \mathsf{Encrypt}(pk_1, m) \\ m' := \mathsf{Decrypt}(sk_2, c)\end{array}\right] \leq \rho \ .$$

After proving Lemma 7, it only remains to analyze the required PKE properties.

Lemma 7 (hROB of FO-based KEMs)**.** *Explicitly rejecting FO-based KEMs satisfy* hROB *if their underlying encryption scheme is sufficiently spread and weak collision free.*

Concretely, let PKE *be a public-key encryption scheme that is γ-spread and ρ-weak collision free and for which two honestly generated public keys collide with probability at most η, and let* $\mathsf{KEM} := \mathsf{FO}^{\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}]$*. Then*

$$\mathbf{Adv}^{\mathsf{hROB}}_{\mathsf{KEM}}(\mathcal{A}) \leq 2^{-\gamma} + \rho \ .$$

Lemma 7. We want to show that $2^{-\gamma} + \rho$ is an upper bound for

$$\mathbf{Adv}^{\mathsf{hROB}}_{\mathsf{KEM}}(\mathcal{A}) = \Pr\left[k \neq \perp \;\middle|\; \begin{array}{l}(pk_1,.),(.,sk_2) \leftarrow\!\$\ \mathsf{KGen}() \\ (c,.) \leftarrow\!\$\ \mathsf{Encap}(pk_1) \\ k := \mathsf{Decap}(sk_2, c)\end{array}\right] \ .$$

The first part of the proof proceeds like the one of Lemma 6. Plugging in the concrete algorithms for the FO-based KEM, $(c,.) \leftarrow\!\$\ \mathsf{Encap}(pk_1)$ translates to $c \leftarrow\!\$\ \mathsf{Encrypt}(pk_1, m; \mathtt{G}(m))$ for a uniformly chosen message m, and $k := \mathsf{Decap}(sk_2, c)$ will differ from $\perp$ if and only if c passes FO's re-encryption check, that is, if $c = \mathsf{Encrypt}(pk_2, m'; \mathtt{G}(m'))$ for $m' := \mathsf{Decrypt}(sk_2, c)$.

We make a case distinction: either $m' = m$ or $m' \neq m$. If $m' \neq m$, k will differ from $\perp$ if and only if encrypting m', a message different from m, accidentally hits c. Since we assume PKE to be γ-spread, for any key pair and any message m', this happens with probability at most $2^{-\gamma}$. This probability is taken over the randomness chosen in Encrypt which here is the output of random oracle G on a fresh message and consequently uniform.

The other case, $m' = m$, is exactly the case that encrypting a random message m under one key pair and then decrypting the result under an independently sampled one still yields the original message. By assumption, we can bound the probability of this case occurring by ρ. (Note that in our analysis, we can replace the used encryption randomness $\mathtt{G}(m)$ by uniform randomness since Gis a random oracle and we are interested in a statistical property.) □