Generic Constructions of Quantum-Resistant Cryptosystems

RUHR
UNIVERSITÄT
BOCHUM

RUB

Doctoral Thesis

Kathrin Hövelmanns

Fakultät für Mathematik

Ruhr-Universität Bochum

Bochum, December 9, 2020

Acknowledgments.

There is a vast number of people I am greatly indebted to, who contributed to this thesis, and to whom I now want to say 'thank you'.

Above all, I want to express my gratitude to my advisor Eike Kiltz for introducing me to the intriguing research area of quantum-resistant cryptography which I deeply enjoyed working on, as I feel it is very timely and, more importantly, really fun. I am also very grateful for the numerous opportunities to travel, letting me get to know many members of the crypto community which I by now so appreciate. Furthermore, I really treasure his honest feedback, his patience when answering even the stupidest question, his trust in me, and the occasional shared anecdote about failed attempts to dye one's hair blue.

Second, I want to thank all of my coauthors for the fruitful and enjoyable collaboration, the exchange of ideas beyond our publications and plenty of laughs. I also want to thank Damien Stehlé, Vadim Lyubashevsky and Christian Schaffner for inviting me to present my research and for fruitful discussions subsequent to my talks.

Many thanks to all other current and former members of the crypto group, that is, Benedikt, Bertram, Dominik, Doreen, Eduard, Felix, Federico, Giorgia, Julian, Julien, Manuel, and Ralph for being so much fun to be around, and for shared deadlines, travels, and movie evenings. I will surely miss all of you. Special thanks go out to Bene for suffering through my puns, to Federico for providing me with an indefinite amount of sweets if needed (and on any other occasion as well), and to Manuel for being a friend, for crossing some kinda shady streets in Baltimore with me just to visit Green Mount, and for sending the cutest picture of his adorable daughter Tilda right before I defended this thesis. I also had plenty of fun getting educated by Bertie and Gio on which craft beer is the best, even if your efforts were kinda wasted on me.

Special thanks also go out to Alex May for making the time to review my thesis even though he already had a lot on his plate, and to Manuel for looking at the presentation of my results with a keen eye.

When it comes to moral and 'meta-academical' support, I most thankfully relied on the thoughtful and constructive feedback and the kind words of Chris Brzuska, Bertram, and Manuel. Thanks to all of you for pointing out my strengths when I experienced doubts.

I also want to thank the organisers of Women in Theory (WIT). I am thankful for the opportunity to meet so many inspiring female colleagues, and I deeply appreciate the whole concept of WIT.

Last, but certainly not least I want to express my love and gratitude towards my 'framily', above all others towards Lennart, Andres and Judith, for supporting and believing in me, and for always being there when I need you.

Contents

Introduction

This thesis investigates the question if and how practical public-key cryptosystems can be proven post-quantum secure. In the post-quantum scenario, we consider attackers with quantum capabilities that interact with classical networks. As many practical cryptosystems were originally proven secure in idealised models that do not reflect quantum capabilities, we will revisit their design rationale and show that a proof of security is feasible, even in a model that does capture these additional capabilities. We begin with a short introduction to the concept of provable security in general, focusing in particular on the setting of post-quantum security.

Public-key cryptography and provable security

Cryptography is the science of designing schemes "to work with a communication system in the presence of adversaries, for the purpose of defeating the adversaries' intention" [Riv]. This rather broad definition encompasses many goals, amongst which some of the most important ones are the design of systems that provide communication secrecy, as well as authentication and integrity of communicated messages. The importance to achieve these goals is immediate, as the processing and communication of data has become more and more ubiquitous. Until the late 70's, however, all designs were given in an "ad-hoc" fashion, meaning that they were conjectured to meet their goal because it was not known how to execute a practical attack, and it was believed that nobody would come up with such in due time. This intuition sometimes was falsified later, with one notable example being Bleichenbacher's attack [Ble98], leading to devastating consequences. Furthermore, it wasn't always made explicit what exactly the design even was conjectured to achieve, as the security goals were often stated only informally. For more advanced security goals like, e.g., authenticated key exchange or multiparty computation, there exists a plethora of different scenarios that could be considered, depending on context. Simply stating that a design is secure might hence be misleading.

Provable security. First introduced in [GM82] in the context of public-key encryption, provable security is a methodological approach that consists of three components: First, a formal definition of the goal that is to be achieved, second, a construction, and third, a rigorous proof that under some underlying hardness assumption, the construction indeed meets the defined goal. In the context of public-key encryption, common examples for problems that are conjectured to be hard are prime factorisation and computing discrete logarithms in certain groups.

In more detail, the proof shows that for any adversary, successfully attacking the scheme according to the given definition, there also exists a solver (which is usually called a reduction) that can exploit this adversary to efficiently solve the underlying problem. A security proof hence reduces the (potentially rather complex) task of cryptanalising a particular scheme to the task of studying the underlying hardness assumption. Today, the paradigm of provable security is widely accepted as standard to justify a design rationale, see, e.g., [KL14].

The importance of *tight* security proofs. While on a high level, a security reduction might be understood as a proof by contradiction, it is crucial to quantify how tightly a hypothetical attacker can be related to the reduction, both in terms of success probability and running time. Say we consider a hypothetical attacker A, we constructed a reduction B and we showed that A's success probability in attacking the scheme can be upper bounded in terms of B's success probability in solving a (presumably hard) problem P. The reduction is called *tight* (see, e.g., [BR96, BBM00, BR09]) if B does not run significantly longer than A itself, and it is shown that B succeeds in solving P whenever A succeeds in attacking the scheme. In this case, the reduction allows us to argue that the scheme is as secure as the problem is hard, and we can simply choose parameters for which the underlying problem is conjectured to be so. On the other hand, we call a reduction *non-tight* if the reduction B does run significantly longer than the attacker A itself, or if the proven upper bound for A's success probability is significantly larger than B's success probability. To make up for this gap, the parameters of the underlying problem then have to be scaled up to render it harder. Since this scaling up in turn would lead to less efficient schemes, it is by far more desirable to give a tight reduction whenever possible. There also exist definitions that take into consideration the amount of memory that has to be used, see, e.g., [ACFK17, BJL17, WMHT18], this thesis, however, will only be considered with the definition given above.

Actively secure public-key encryption. The security definition given in [GM82] was superseded by the definition of Indistinguishability under Chosen Plaintext Attacks (IND-CPA security, [GM84]). While the two definitions are equivalent, the notion of

IND-CPA security is easier to work with. Intuitively, IND-CPA security formalises that an adversary, given the public key, cannot tell apart encryptions of plaintexts of their choosing. The definition is given in terms of a security game, in which the adversary receives a public key and must choose two messages. The game then provides the adversary with an encryption of one of these messages, and the adversary wins the game if it can tell which message was encrypted. A scheme is said to possess IND-CPA security if no adversary can win both with high probability and in reasonable time. This definition was expanded further to the notion of Indistinguishability under Chosen Ciphertext Attacks (IND-CCA security, [NY90, RS92]). The respective security game differs from the IND-CPA game by additionally providing the adversary with a decryption oracle that decrypts ciphertexts of the adversary's choosing. While it might not be clear at a first glance how a real-world adversary interacting with the scheme could enforce the system to provide such an oracle, IND-CCA-like attacks indeed have occured in practice (e.g., Bleichenbacher's attack [Ble98, BFK+12]): Intuitively, the additional attack surface stems from the fact that the scheme's functionality is "plugged in" into a more complex primitive (e.g., a protocol to exchange secret key material over an insecure channel). As Bleichenbacher's attack has shown, the notion of IND-CPA security is not sufficient to show that this plugging-in can be done, securely.

Today, the notion of IND-CCA security is hence widely accepted as the standard security notion for public-key encryption schemes, however, it is usually much more difficult to prove than IND-CPA security. In order to simplify the construction of IND-CCA secure schemes, several transformations have been suggested that turn a public-key encryption scheme with weaker security properties into an IND-CCA one, generically, with the most efficient constructions being in the random oracle model.

The Random Oracle Model (ROM)

The random oracle model is a proof heuristic in which unkeyed public hash functions are replaced with an idealisation: The hash function is replaced with a perfectly random function to which the attacker has oracle access. Since its introduction [BR93], the ROM has allowed cryptographers to prove practical cryptosystems secure for which proofs in the standard model have been elusive, with notable examples in the realm of encryption being OAEP [BR95], REACT [OP01], GEM [CHJ+02], DHIES [ABR01] and the Fujisako-Okamoto (FO) transform [FO99, FO13]. In the realm of signatures, some notable examples are the Fiat-Shamir transform [PS96b], PSS [BR96] and Okamoto-

Schnorr blind signatures [PS96a]. Indeed, REACT, GEM, FO and the Fiat-Shamir transform will be important subject of this thesis, as in the random oracle, they achieve the desired level of security from weaker properties, albeit not necessarily with a tight reduction. In the next section, however, we will see that their original proofs were unfit to argue that these constructions are also post-quantum secure. In order to already hint at how these proofs might fail when quantum attackers have to be considered, we will now quickly recap some of the most important properties of the ROM.

In general, the ROM allows for proofs that are conceptually simpler and often tighter than proofs in the standard model, with an important reason being that due to its oracle nature, the ROM enjoys the following two convenient properties:

Preimage awareness. Since a random oracle O has to be accessed by an oracle query in order to learn the value $\mathsf{O}(x)$ of a particular preimage x, a security reduction can keep track of the preimages that the attacker was interested in. A common proof strategy is to argue that the hypothetical attacker could not possibly succeed unless it queries the oracle on a particular preimage, and that this preimage then can be used by the reduction to solve the underlying problem. Since preimage awareness allows a reduction to extract a solution from the attacker, this property is also sometimes called *extractability*.

Programmability. Since random oracle values are undetermined until the attacker poses a query, the underlying function can be changed on all unqueried values during the security proof. As a consequence, a reduction can determine oracle answers according to its own needs, e.g., by embedding its own problem instance in one of its responses. As long as the answers are uniform and consistent with the rest of the attacker's view, this change will not be noticed.

While there exist pathological examples of ROM uninstantiability [CGH98, BFM15], meaning that there exist constructions that are provably secure in the ROM, but insecure when instantiated with any concrete hash function, the schemes we encounter in practice do not show the behavior of those schemes. Indeed, we do not know of any attacks against schemes that were proven secure in the random oracle model. Hence, the security of a cryptographic scheme in the ROM is still believed to be a good indicator for its security in the real world.

Post-quantum security

Opposed to the computers that are in use as of today, the internal workings of quantum computers are based on the laws of quantum mechanics, and work with quantum states.

As a simplified intuition, we can envision a quantum bitstring as a collection of several classical bitstrings, all waiting to happen with a certain probability. The bitstring is then said to be *a superposition* of the constituting classical bitstrings. Together with the fact that classical information can be obtained from a quantum state by performing measurements, we observe that quantum states enjoy fascinating features: Unlike their classical counterparts, two quantum states can become entangled, meaning that classical information resulting from a measurement of either one is perfectly correlated to the classical information that would be obtained from the other one.

Entanglement is one of the reasons why quantum computers are considered more powerful than classical computing devices. As the first (theoretical) example for quantum supremacy, Shor [Sho94] proved in 1994 that a sufficiently scaled quantum computer can factor large integers efficiently. In subsequent works, it was further shown that quantum computers have a significant advantage when it comes to searching on an unstructured database [Gro96] or solving the discrete logarithm problem [Wat01]. For a classical computer, these problems are considered infeasible (for large enough instances), which is why they were used to define the core of many real-world public-key cryptosystems.

The realisation of a suffiently scalable quantum computer hence could pose a threat to most of the IT infrastructure that is in use today. This threat is often viewed as quite immediate for two reasons: First, even information that was transmitted today will not remain private once a suffiently scalable quantum computer is built, as an attacker can record and store publicly exchanged data with the purpose to exploit it later. Second, migration to quantum-secure infrastructures will take a lot of time due to, e.g., standardisation and legacy issues. As a reaction, the National Institute of Standards and Technology (NIST) posed a call for proposals in 2017 with the goal to standardise new public-key primitives [NIS17] with security against quantum adversaries. Given that these new standards, once they are established, are supposed to be put into action as fast as possible, the primitives are not required to consist of quantum algorithms, but rather to consist of classical algorithms that are resistant against quantum attackers interacting with a classical network. This scenario is called *post-quantum cryptography*. One can also envision a world in which even the network itself is quantum, which is the scenario of *quantum cryptography*. The scenario of quantum cryptography, however, will not be considered in this thesis.

Clearly, the first important step in the transition towards a quantum-secure future is the search for problems that remain infeasible even for a computer with quantum capabilities, in order to replace the problems which are known to be "broken". Promising candidates are certain problems over lattices and codes: As we do not know of any quantum attacks that succeed significantly faster than known classical attacks, it is conjectured that these problems are sufficiently secure replacements. Similarly important, we can base desired primitives upon these problems in a way such that the penalty in terms of efficiency is not too large.

Unfortunately, there still exist further obstacles on the way to find new, post-quantum secure primitives: One might believe that by replacing the "broken" computational hardness assumptions with a hardness assumption that is conjectured to be hard, even for a quantum computer, it might be straightforward to maintain the rest of the design rationale. We identify two main issues with this belief:

- Many design rationales themselves are proven secure in a way that does not necessarily translate to the setting of post-quantum security. As we will discuss in more detail below, an important example in which we have to revisit the overall proof heuristic is the random oracle model.

- The proofs for the rationale sometimes make implicit assumptions which were met by the "broken" assumptions, and that do not hold true for the new, presumably "quantum-hard" assumptions. An important example will be discussed after our discussion of the random oracle model. While such a gap might be easy to understand once it is identified, these kinds of "translation issues" can become increasingly hard to spot when dealing with increasingly advanced primitives, as their security proofs do not always make explicit the additional assumptions they rely upon.

Ignoring these difficulties can turn a security "proof" meaningless, as its prerequisites then might no longer be met, or the proven statement does not even capture the security scenario in consideration.

The quantum random oracle model

In the scenario of post-quantum security, a quantum adversary interacts with a non-quantum network. As a consequence, all "online" primitives (like encryption executed by honest users or signing) will remain to be considered as classically accessible. All "offline" primitives, however, can be computed by the adversary on its own, and hence, in superposition. With the introduction of quantum adversaries, and the advent of post-quantum cryptography, the ROM hence had to be generalised: Recall that the random oracle model replaces hash functions with a truly random idealisation that the attacker has oracle access to. Given that hash functions are public, the ROM does not capture a quantum attacker's capabilities to compute their code in superposition. To account for these stronger capabilities, the quantum-accessible ROM (QROM) was introduced in [BDF+11].

While successfully fixing the definitional gap, the QROM does come with its own challenges: In general, the ROM allowed for proofs that are conceptually simpler and often tighter than standard model security proofs, as it enjoyed several convenient properties we already mentioned. In the quantum random oracle model, however, things become slightly more involved, as neither preimage awareness nor reprogrammability carry over in an obvious manner for oracles that are quantum-accessible. Intuitively, one can envision the preimage in question to be hidden within the superposition. It hence is not clear how to extract a preimage without performing a measurement, and thereby derailing the attacker's behaviour. Similarly, it is not immediately clear whether reprogramming does not cause a change in the attacker's view, as the reprogramming position might already have been contained in a former query to the random oracle.

Given that these properties usually are the reason why random oracle proofs are conceptually simpler than proofs in the standard model, it comes as no surprise that proofs in the quantum random oracle model fail to meet their classical counterparts with respect to simplicity. More importantly, while there do exist quantum generalisations that can be used to execute a similar reasoning in many important cases, they usually come with a large penalty in terms of the reduction's tightness. As a consequence, deploying these generalisations to prove security leads to less efficient schemes, as the parameters of the underlying problem have to be scaled up, accordingly. This is the reason why there exists a lot of interest in the question whether a) the generalisations can be improved with regards to tightness, and/or b) whether proofs can be found that manage to circumvent these difficulties altogether, without introducing additional requirements.

The NIST standardisation process and generic constructions

At the time that NIST posed its call for proposals, it was well understood how to base passively secure encryption schemes on code- or lattice-based problems.[1] All 17 proposals for public-key encryption that made it to the second round of the standardisation process

[1] This is not meant to say that the underlying lattice problems themselves do not have to be subjected to further cryptanalysis. However, conditioned on the assumption that they do withstand cryptanalysis, the resulting schemes can be considered secure when appropriate parameters are chosen.

follow the strategy to construct such a passively secure scheme, and then to apply some variant of the FO transformation. Only four [BCL+19, BCLv16, ABD+18, JAC+19] out of those 17 proposals, however, were designed in a way such that decryption of a faithfully encrypted plaintext m never fails to yield m. All other code- and lattice-based proposals fail to decrypt for particularly unfavourable combinations of key, message, and encryption randomness.

Apart from being in the (non-quantum) ROM, the original proof for FO implicitly relies on perfect correctness (which was satisfied by Diffie-Hellman- and RSA-based decryption algorithms), this property hence indeed is an example for implicit assumptions that limit the applicability of a proof to new constructions. While one might assume that non-perfect correctness should be a non-issue due to the small probability of decryption failures in practice, it turns out that even the possibility of such failure opens up an attack avenue that was nonexistent in formerly deployed standards.

Furthermore, the proof for FO was non-tight, leading to bigger parameters. While REACT and GEM came with a tight security reduction, they require the underlying encryption scheme to fulfill a stronger security notion that is not naturally met by lattice-based encryption schemes.

Another important construction in the context of the NIST standardisation process is the Fiat-Shamir transform, which is used to design signature schemes. Of the nine proposals for signature schemes that made it to the second round, four [DKL+18, ZCD+19, CHR+16, ABB+20] deploy variants of Fiat-Shamir. The original proof strategy for these variants, however, exploits the reprogrammability of the involved random oracle.

Main Results of this Thesis

We conclude this chapter by summarising the main results presented in this thesis.

Generic transformations to achieve actively secure public-key encryption. In Chapter 2, we are interested in three central questions: Are there constructions that achieve $\mathsf{IND\text{-}CCA}$ from schemes with weaker security properties,

- with a security proof as tight as possible,
- even if there exists a possibility of decryption failure,
- and even if the adversary possesses quantum capabilities?

To answer those questions, we provide in Section 2.1 a modular toolkit of transformations that can be combined, according to the properties of the underlying scheme.

We remark that some of our transformations essentially recover more versatile variants of the original FO and GEM/REACT transformations. Whereas previously known security proofs in the random oracle model for the FO transformation were non-tight, our combinations tightly turn IND-CPA into IND-CCA security, with a very small efficiency overhead. A selection of these transformations is then also proven secure in the quantum random oracle model in Sections 2.2 and Section 2.3. Since all of our security proofs furthermore account for the possibility of decryption failures, the results are applicable to all proposals currently under consideration for standardisation by the NIST, and indeed, our analysis was used to give guidance on how to choose the parameters, in particular with respect to correctness errors. In Section 2.4, we further broaden the class of schemes to which our security proofs apply by weakening our correctness requirements.

The results presented in Section 2.1 and Section 2.2 are based on joint work with Dennis Hofheinz and Eike Kiltz, published in [HHK17], and the result presented in Section 2.3 was established as a helper result in [HKSU20] to achieve authenticated key exchange (see “further publications”). The result presented in Section 2.4 is an unpublished manuscript, based on discussions with Eike Kiltz, Vadim Lyubashevsky and Dominique Unruh.

Adaptive reprogrammability in the quantum random oracle model. As we have pointed out, The ROM still enjoys widespread popularity, mostly because it tends to allow for tight and conceptually simple proofs where provable security in the standard model is elusive or costly. While being the adequate replacement of the ROM for post-quantum security, the QROM has thus far failed to provide these advantages in many settings. In Chapter 3, we focus on *adaptive reprogrammability*: We show that adaptive reprogramming is feasible also in the QROM. More precisely, we prove a bound on the adversarial advantage in distinguishing whether a random oracle has been reprogrammed or not. The achieved bound is essentially optimal, as there exists an attack that matches our bound, up to a constant factor. The resulting statement is a straightforward QROM generalisation of adaptive reprogrammability and can serve as a drop-in replacement in many important cases. We demonstrate that our technique recovers the mentioned advantages of the ROM in several QROM applications:

- We show that the standard ROM proof of chosen-message security for Fiat-Shamir signatures can be lifted to the QROM, straightforwardly, achieving a reduction that is tighter than previously known, and requires less from the underlying scheme.

- We give the first QROM proof of security against fault injection and nonce attacks for the hedged Fiat-Shamir transform.

The results presented in Chapter 3 are based on joint work with Alex Grilo, Andreas Hülsing, and Christian Majenz (currently in submission).

Further publications.

Post-quantum secure authenticated key exchange (AKE) with weak perfect forward secrecy. AKE protocols allow applications to switch from the use of asymmetric cryptography (which usually features computationally more expensive mathematical structures) to symmetric key cryptography with simpler structure and faster algorithms. The overall quantum resistance of many security systems crucially relies on that of the underlying AKE protocol. Most AKE protocols, however, rely on constructions based on an ad-hoc Diffie-Hellman key exchange that is authenticated either via digital signatures, Diffie-Hellman-based mechanisms, or public-key encryption. Since classical Diffie-Hellman based mechanisms do not provide security in the presence of quantum attackers, it is an important task to build efficient post-quantum secure AKE protocols entirely from alternative mathematical assumptions that are not known to be vulnerable to quantum attacks. Lattice-based cryptography currently is seen as one of the most popular and versatile approaches to mitigate the quantum threat. However, digital signatures based on lattice assumptions are usually considered less efficient. This issue can be circumvented by considering constructions that, instead of authenticating via signatures, use implicit authentication. While there already existed constructions [FSXY12, FSXY13] that only rely on PKE to authenticate implicitly, they required the underlying PKE scheme to be perfectly correct. Similar to the case of actively secure encryption, dealing with the possibility of correctness errors is one of the major difficulties in a setting involving active attacks against an AKE protocol. In a joint work with Eike Kiltz, Sven Schäge, and Dominique Unruh [HKSU20], it was therefore shown how to achieve a post-quantum secure two-message authenticated key exchange (AKE) protocol from any passively secure public-key encryption scheme, without having to rely on signatures or perfect correctness. As a consequence, our construction can be instantiated with any of the submissions to the recent NIST competition, e.g., those based on codes and lattices. In fact, when applied to schemes such as these, our construction is more natural than ones that were previously known.

Tighter proofs of IND-CCA security. A lot of research has been invested in finding tighter proofs for different variants of FO-like conversions, in the quantum random oracle model. Amongst these efforts is a joint work with Nina Bindel, Mike Hamburg, Andreas Hülsing, and Edoardo Persichetti [BHH+19], in which we revisited the constructions given in [HHK17]. In particular, this work provides a proof of IND-CCA security for constructions from deterministic public-key encryption schemes that is less non-tight than previous ones. This result is enabled by a new quantum query extraction technique which gives a tighter bound than previous ones. Since there has been real-world interest in the question if and how the choice of transformation impacts security, the relation between these different constructions is clarified by proving several equivalences and implications.

Chapter 1

Preliminaries

For $n \in \mathbb{N}$, we let $[n] := \{1, \ldots, n\}$. For a finite set S, we denote by $|S|$ the cardinality of S, and by $x \leftarrow_{\$} S$ we denote the sampling of a uniform random element x. We denote the sampling according to some distribution $\mathfrak{D}$ by $x \leftarrow \mathfrak{D}$. By $[\![B]\!]$ we denote the bit that is 1 if the boolean statement B is true, and otherwise 0. For a subset $U \subset S$, we will denote by U^c the complement of U in S. By $\mathbb{1}$ we denote the identity map.

Algorithms. We denote deterministic output y of an algorithm A on input x by $y := \mathsf{A}(x)$. 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 $y \leftarrow \mathsf{A}(x)$.

Random variables. Given a discrete distribution $\mathfrak{D}$ over a set X, we define the support of $\mathfrak{D}$ as the set $\mathrm{supp}(\mathfrak{D}) := \{x \in X : \mathfrak{D}(x) > 0\}$. For distributions on $\mathbb{R}$ with a finite number of finite outcomes, we define the *expectation of* $\mathfrak{D}$ as

$$\mathbb{E}[\mathfrak{D}] := \sum_{x \in \mathrm{supp}(\mathfrak{D})} x \cdot \mathfrak{D}(x) \ .$$

We furthermore define the statistical distance between two discrete distributions $\mathfrak{D}_1$ and $\mathfrak{D}_2$ over a set X as

$$\mathrm{Dist}(\mathfrak{D}_1, \mathfrak{D}_2) := \frac{1}{2} \cdot \sum_{x \in X} |\mathfrak{D}_1(x) - \mathfrak{D}_2(x)| \ .$$

Games. Following [Sho04b, BR06], we use code-based games. Games will run adversaries A as a subroutine. We will say that A *wins in game G* if the game terminates with output 1, and will denote this event by $G^{\mathsf{A}} \Rightarrow 1$. We implicitly assume boolean variables to be initialised to false, numerical types to 0, sets to $\emptyset$, and strings to the empty string. A boolean game variable is called *a flag* if it starts off as **false** and changes its value at most once: Once a flag is set to **true**, it can never revert to **false**. We will call two games G and H *identical-until-bad* if they share the same pseudocode, except for statements that are subsequent to the event that a flag BAD was set to **true**. Let BAD_G (BAD_H) be the event that flag BAD is set to **true** at the end of execution of A in game G (game H). If G and H are identical-until-bad, we have that $\Pr[\mathrm{BAD}_G] = \Pr[\mathrm{BAD}_H]$, and we can simply write $\Pr[\mathrm{BAD}]$ instead.

Lemma 1.0.1. (Difference Lemma [Sho04b, Lem. 1]) Let A be an adversary, and let G, H be identical-until-bad games with respect to flag BAD. Then

$$|\Pr[G^{\mathsf{A}} \Rightarrow 1] - \Pr[H^{\mathsf{A}} \Rightarrow 1]| \leq \Pr[\mathrm{BAD}] \ .$$

In the next two sections, we will define syntax and security notions for several public-key primitives. For a more detailed discussion of the standard definitions included in Sections 1.1 and 1.2, we refer to [KL14]. The algorithms of such primitives are usually parametrised by a public system parameter *par*. For convenience, we will treat the algorithms as implicitly parametrised and omit the system parameter *par* from out notation. While we provide a concrete treatment of those definitions, we sometimes misstep by informally writing something like “Transformation X turns A security into B security”, by which we mean that the advantage of a B-attacker against the X-transformed can be upper bounded in terms of a reduction that attacks the A security of the underlying primitive.

1.1 Public-Key Encryption and Key Encapsulation Mechanisms

Definition 1.1.1 (Public-Key Encryption Schemes)**.** A $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ consists of three algorithms, and a finite message space $\mathcal{M}$.

- The key generation algorithm KG outputs a key pair (pk, sk), where pk also defines a finite randomness space $\mathcal{R} = \mathcal{R}(pk)$ as well as a ciphertext space $\mathcal{C}$.
- The encryption algorithm Enc, on input pk and a message $m \in \mathcal{M}$, outputs an encryption $c \leftarrow \mathsf{Enc}(pk, m)$ of m under the public key pk. If necessary, we make the used randomness of encryption explicit by writing $c := \mathsf{Enc}(pk, m; r)$, where $r \leftarrow_{\$} \mathcal{R}$.
- The deterministic decryption algorithm Dec, on input sk and a ciphertext c,

outputs either a message $m = \mathsf{Dec}(sk, c) \in \mathcal{M}$ or a special symbol $\perp \notin \mathcal{M}$ to indicate that c is not a valid ciphertext.

Definition 1.1.2 (γ-spreadness [FO99])**.** We say that PKE is γ-spread iff for all key pairs $(pk, sk) \in \mathrm{supp}(\mathsf{KG})$ and all messages $m \in \mathcal{M}$ it holds that

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

where the probability is taken over the internal randomness Enc.

In one section of this thesis, we will consider encryption schemes that come with an additional property: We will require that given a ciphertext and its engendering plaintext, it is possible to recover the used randomness. In particular, we will require that recovering the wrong randomness occurs only if decryption failure occurs. We formalise this property in the next definition.

Definition 1.1.3 (Randomness recovery [LS19])**.** We say that PKE *comes with randomness recovery* if there exists an algorithm Rec that takes as input sk, a message m, and a ciphertext c, and outputs some randomness r such that for all key pairs $(pk, sk) \in \mathrm{supp}(\mathsf{KG})$, all messages $m \in \mathcal{M}$ and all $r \in \mathcal{R}$ it holds that either

$$m' := \mathsf{Dec}(sk, c) \neq m$$

or

$$\mathsf{Rec}(sk, m', c) = r \ ,$$

where $c := \mathsf{Enc}(pk, m; r)$.

We will furthermore require that, vice versa, it is possible to recover the plaintext from the ciphertext and the used randomness. We formalise this property in the next definition.

Definition 1.1.4 (Invertible Encryption)**.** We say that PKE *comes with invertible encryption* if there exists an algorithm Inv that takes as input pk, r and c and outputs a message m such that for all key pairs $(pk, sk) \in \mathrm{supp}(\mathsf{KG})$, all messages $m \in \mathcal{M}$ and all $r \in \mathcal{R}$ it holds that

$$\mathsf{Inv}(pk, \mathsf{Enc}(pk, m; r), r) = m \ .$$

1.1.1 Security Notions for Public-Key Encryption

Standard notions under passive attacks

First, we define two standard security notions for public-key encryption: One-Wayness (OW) and Indistinguishability under Chosen-Plaintext Attacks ($\mathsf{IND\text{-}CPA}$).

Definition 1.1.5 (OW, $\mathsf{IND\text{-}CPA}$)**.** Let $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme with message space $\mathcal{M}$. We define game OW as in Figure 1.1, and the OW *advantage function of an adversary* A *against* PKE as

$$\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{A}) := \Pr[\mathsf{OW}^{\mathsf{A}}_{\mathsf{PKE}} \Rightarrow 1] \ .$$

Furthermore, we define game $\mathsf{IND\text{-}CPA}$ (also in Figure 1.1), and the $\mathsf{IND\text{-}CPA}$ *advantage function of an adversary* $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ *against* PKE (where A_2 has binary output) as

$$\mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{A}) := |\Pr[\mathsf{IND\text{-}CPA}^{\mathsf{A}} \Rightarrow 1] - \frac{1}{2}| \ .$$

```
GAME OW
$(pk, sk) \leftarrow \mathsf{KG}$
$m^* \leftarrow_\$ \mathcal{M}$
$c^* \leftarrow \mathsf{Enc}(pk, m^*)$
$m' \leftarrow \mathsf{A}(pk, c^*)$
return $[\![m' = m^*]\!]$

GAME IND-CPA
$(pk, sk) \leftarrow \mathsf{KG}$
$b \leftarrow_\$ \{0,1\}$
$(m_0^*, m_1^*, \mathrm{st}) \leftarrow \mathsf{A}_1(pk)$
$c^* \leftarrow \mathsf{Enc}(pk, m_b^*)$
$b' \leftarrow \mathsf{A}_2(pk, c^*, \mathrm{st})$
return $[\![b' = b]\!]$
```

Fig. 1.1: Games OW and $\mathsf{IND\text{-}CPA}$ for PKE.

As proven in, e.g., [KL14], $\mathsf{IND\text{-}CPA}$ security of a scheme PKE with sufficiently large message space $\mathcal{M}$ implies its OW security.

Lemma 1.1.6. For any adversary A, there exists an adversary B such that $\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B}) + {}^{1}/_{|\mathcal{M}|}$, and the running time of A is about that of A.

We also consider $\mathsf{IND\text{-}CPA}$ security in the (quantum) random oracle model, where PKE and adversary A are given access to a (quantum) random oracle. (How to model quantum access will be made explicit in Section 1.3.) We will usually denote the number of random oracle queries to O by q_{O}.

```
GAME OW-ATK:
    $(pk, sk) \leftarrow \mathsf{KG}$
    $m^* \leftarrow_\$ \mathcal{M}$
    $c^* \leftarrow \mathsf{Enc}(pk, m^*)$
    $m' \leftarrow \mathsf{A}^{\mathsf{O}_{\mathsf{ATK}}}(pk, c)$
    return $[\![ m' = m^* ]\!]$

PCO($m \in \mathcal{M}, c$)
    return $[\![ \mathsf{Dec}(sk, c) = m ]\!]$

VALID($c \neq c^*$)
    $m := \mathsf{Dec}(sk, c)$
    return $[\![ m \in \mathcal{M} ]\!]$
```

Fig. 1.2: Games $\mathsf{OW\text{-}ATK}$ for PKE, where $\mathsf{ATK} \in \{\mathsf{PCA}, \mathsf{VCA}, \mathsf{PVCA}\}$ and $\mathsf{O}_{\mathsf{ATK}}$ is defined in Definition 1.1.7. PCO is the Plaintext Checking Oracle, and VALID is the Validity checking Oracle.

One-Wayness with access to additional oracles (OW-PVCA)

Next, we introduce some intermediate helper notions which we will use during our security proofs. These intermediate notions are variations of one-wayness, in which the adversary has additional access to one or both of the following two oracles:

- **Plaintext-ciphertext checking oracle PCO.** Oracle PCO takes as input a message $m \neq \perp$ and a ciphertext c, and returns to the adversary whether c decrypts to m.

- **Validity checking oracle VALID.** Oracle VALID takes as input a ciphertext c and returns whether c decrypts to a message that lies within the message space.

We will call the respective notions One-Wayness under Plaintext Checking Attacks (OW-PCA), under Validity Checking Attacks (OW-VCA), and under Plaintext and Validity Checking Attacks (OW-PVCA). In previous literature [Den03], OW-VCA security was called OW-CPA$^+$ security, and OW-PVCA security was called "OW-CPA$^+$ security with access to a PCO oracle".

Definition 1.1.7 (OW-ATK). Let $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a public-key encryption scheme with message space $\mathcal{M}$. For $\mathsf{ATK} \in \{\mathsf{PCA}, \mathsf{VCA}, \mathsf{PVCA}\}$, we define $\mathsf{OW\text{-}ATK}$ games as in Figure 1.2, where

$$\mathsf{O}_{\mathsf{ATK}} := \begin{cases} \mathrm{PCO} & \mathsf{ATK} = \mathsf{PCA} \\ \mathrm{VALID} & \mathsf{ATK} = \mathsf{VCA} \\ \mathrm{PCO}, \mathrm{VALID} & \mathsf{ATK} = \mathsf{PVCA} \end{cases} .$$

We define the $\mathsf{OW\text{-}ATK}$ *advantage function of an adversary* A *against* PKE as

$$\mathrm{Adv}^{\mathsf{OW\text{-}ATK}}_{\mathsf{PKE}}(\mathsf{A}) := \Pr[\mathsf{OW\text{-}ATK}^{\mathsf{A}}_{\mathsf{PKE}} \Rightarrow 1] .$$

```
GAME $\mathsf{DS}_b$:
$(pk, sk) \leftarrow \mathsf{KG}$
$m^* \leftarrow_\$ \mathcal{M}$
$c_0^* \leftarrow \mathsf{Enc}(pk, m^*)$
$c_1^* \leftarrow \mathsf{fakeEnc}(pk)$
$b' \leftarrow \mathsf{A}(pk, c_b^*)$
return $b'$
```

Fig. 1.3: Games DS_b (where $b \in \{0,1\}$) for PKE.

Note that our definition of the plaintext checking oracle PCO (see Figure 1.2) implicitly disallows queries on messages $m \notin \mathcal{M}$. (We make the convention that for messages $m \notin \mathcal{M}$, $\mathrm{PCO}(m, c)$ always returns $\perp$.) This restriction is important since otherwise, the ciphertext validity oracle VALID could be simulated via access to PCO, as $\mathrm{VALID}(m) = \mathrm{PCO}(\perp, c)$. Similarly, the ciphertext validity oracle $\mathrm{VALID}(c)$ implicitly disallows queries on the challenge ciphertext c^*.

Disjoint Simulatability (DS)

Following [SXY18], we will also consider PKE schemes for which it is possible to sample fake ciphertexts such that they are indistinguishable from proper encryptions of a random message (simulatability), while the intersection of fake ciphertexts and proper encryptions is unlikely (disjointness).

Definition 1.1.8. (DS) Let $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme with message space $\mathcal{M}$, coming with an additional algorithm $\mathsf{fakeEnc}$ that takes as input a public key pk and outputs bitstrings. We define games DS_b (where $b \in \{0,1\}$) in Figure 1.3, and the DS *advantage function of an adversary* A *against* $(\mathsf{PKE}, \mathsf{fakeEnc})$ as

$$\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE},\mathsf{fakeEnc}}(\mathsf{A}) :=\mid \Pr[\mathsf{DS}_0^{\mathsf{A}} \Rightarrow 1] - \Pr[\mathsf{DS}_1^{\mathsf{A}} \Rightarrow 1] \mid \ .$$

When there is no chance of ambiguity, we will drop $\mathsf{fakeEnc}$ from the advantage's subscript for convenience.

Let $\mathsf{Enc}(pk, \mathcal{M}) := \cup_{m \in \mathcal{M}} \operatorname{supp}(\mathsf{Enc}(pk, m))$. We call PKE ϵ_{dis}*-disjoint* if for all $(pk, sk) \in \operatorname{supp}(\mathsf{KG})$ it holds that $\Pr[c \in \mathsf{Enc}(pk, \mathcal{M})] \leq \epsilon_{\mathrm{dis}}$, where the probability is taken over $c \leftarrow \mathsf{fakeEnc}(pk)$.

1.1.2 Key Encapsulation Mechanisms (KEMs)

Key encapsulation mechanisms are schemes that allow a sender to securely transmit a cryptographic key, using the receiver's public key. The motivation of defining this primitive is that public-key encryption schemes are usually less efficient than symmetric encryption schemes, it hence is desirable to use a public-key primitive (the KEM) only to securely transmit a (usually relatively short) symmetric key, and use the symmetric key to encrypt bulk data. Combining an IND-CCA secure KEM with any (one-time) chosen-ciphertext secure symmetric encryption scheme indeed gives rise to an IND-CCA secure public-key encryption scheme [CS03], however, KEMs are also important building blocks in the design of more advanced primitives like authenticated key exchange.

Definition 1.1.9 (Key Encapsulation Mechanisms)**.** A key encapsulation mechanism $\mathsf{KEM} = (\mathsf{KG}, \mathsf{Encaps}, \mathsf{Decaps})$ consists of three algorithms.

- The key generation algorithm KG outputs a key pair (pk, sk), where pk also defines a finite key space $\mathcal{K}$.
- The encapsulation algorithm Encaps, on input pk, outputs a tuple (K, c), where c is said to be an encapsulation of the key K which is contained in key space $\mathcal{K}$.
- The deterministic decapsulation algorithm Decaps, on input sk and an encapsulation c, outputs either a key $K := \mathsf{Decaps}(sk, c) \in \mathcal{K}$ or a special symbol $\perp \notin \mathcal{K}$ to indicate that c is not a valid encapsulation.

We now define Indistinguishability under Chosen-Ciphertext Attacks (IND-CCA).

Definition 1.1.10 (IND-CCA)**.** [RS92] *We define the* IND-CCA *game as in Figure 1.4 and the* IND-CCA *advantage function of an adversary* A *(with binary output) against* KEM *as*

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}(\mathsf{A}) := |\Pr[\mathsf{IND\text{-}CCA}^{\mathsf{A}} \Rightarrow 1] - {}^{1}\!/_{2}| \; .$$

```
GAME IND-CCA
  $(pk, sk) \leftarrow \mathsf{KG}$
  $b \leftarrow_\$ \{0,1\}$
  $(K_0^*, c^*) \leftarrow \mathsf{Encaps}(pk)$
  $K_1^* \leftarrow_\$ \mathcal{K}$
  $b' \leftarrow \mathsf{A}^{\textsc{Dec}}(pk, c^*, K_b^*)$
  return $[\![b' = b]\!]$

Dec$(c \neq c^*)$
  $K := \mathsf{Decaps}(sk, c)$
  return $K$
```

Fig. 1.4: Game IND-CCA for KEM.

1.1.3 Correctness Errors and Rigidity

Average-case correctness of PKE. In previous literature (e.g., [DNR04, BV17]), non-perfect correctness was defined by considering the probability that an encryption of a random message fails to decrypt. As we will soon point out, our results require an alternative definition that is more conservative. To be able to differentiate between those different definitions, however, we will first reintroduce the definition that was given in previous literature, which we will call *average-case correctness* throughout this thesis.

Definition 1.1.11 (Average-case correctness)**.** For a PKE scheme PKE, we define

$$\delta_{\mathrm{ac}} := \Pr\left[\mathsf{Dec}(sk, c) \neq m\right] \quad ,$$

where the probability is taken over $(pk, sk) \leftarrow \mathsf{KG}$, $m \leftarrow_{\$} \mathcal{M}$, and $c \leftarrow \mathsf{Enc}(pk, m)$. We say that a scheme is *δ-average-case correct* if $\delta_{\mathrm{ac}} \leq \delta$.

Note that the phrasing is somewhat counterintuitive, as with this definition, a perfectly correct scheme would be called 0-correct. Since we do not want to deviate from recent literature that deals with correctness errors [BHH+19, JZM19b, KSS+20, BS20], however, we decided to stick with the phrasing above.

Worst-case correctness of PKE. During the proofs of our main results, we will have to take into account that adversaries may deliberately search for messages whose encryptions induce decryption failure[1]. To deal with this setting, using average-case correctness proves insufficient, since messages are chosen adversarially instead of being drawn uniformly at random. Therefore, we will now strengthen our correctness definition such that it takes into account the worst message possible, and call this definition *worst-case correctness*. We want to stress that this differentiation is significant, as in practice, worst-case correctness can be strictly stronger than average-case correctness.[2] This new definition, however, has been carefully crafted such that it is sufficient to prove some of our main theorems, while at the same time being achieved by most recent proposed lattice-based encryption schemes with non-perfect correctness.

[1] For an example in which this possibility indeed has to be tended to while proving IND-CCA security, we refer to Sections 2.1.2 and 2.1.3, pages 68 to 80.

[2] As a simple, albeit artificial example, consider a scheme that is perfectly correct, except for one publicly known message that always induces decryption failure. While this scheme has an average-case correctness error of $\delta_{\mathrm{ac}} = {}^{1}/_{|\mathcal{M}|}$, an adversary could easily trigger decryption failure. As a more practical example of real-world importance, consider the NTTRU encryption scheme defined in [LS19].

Definition 1.1.12 (Worst-case correctness)**.** For a PKE scheme PKE, we define

$$\delta_{\mathrm{wc}} := \mathbb{E}\left[\max_{m\in\mathcal{M}} \Pr[\mathsf{Dec}(sk, c) \neq m]\right] \ ,$$

where the expectation is taken over $(pk, sk) \leftarrow \mathsf{KG}$, and the probability is taken over $c \leftarrow \mathsf{Enc}(pk, m)$. We say that a scheme is *δ-worst-case correct* if $\delta_{\mathrm{wc}} \leq \delta$, and we call a scheme perfectly correct if $\delta_{\mathrm{wc}} = 0$.

In particular, δ_{wc}-worst-case correctness means that even (possibly unbounded) adversaries with access to the secret key will succeed in triggering decryption failure with probability at most δ_{wc}.

We formalise this property with the following game-based definition, in which the adversary outputs a list of messages and wins if at least one message exhibits decryption failure. We also need to include a game-based version in which the adversary has access to a random oracle, as we will cover schemes in this thesis that are defined relative to such.[3]

Definition 1.1.13. We define correctness game COR in Figure 1.5 (left), and the advantage of an adversary A returning a list of N many messages as

$$\mathrm{Adv}^{\mathsf{COR}_N}_{\mathsf{PKE}}(\mathsf{A}) := \Pr[\mathsf{COR}^{\mathsf{A}}_{\mathsf{PKE}} \Rightarrow 1] \ .$$

We furthermore define the random-oracle correctness game $\mathsf{COR\text{-}RO}$ in Figure 1.5 (right), and the advantage of an adversary A as

$$\mathrm{Adv}^{\mathsf{COR\text{-}RO}_N}_{\mathsf{PKE}}(\mathsf{A}) := \Pr[\mathsf{COR\text{-}RO}^{\mathsf{A}}_{\mathsf{PKE}} \Rightarrow 1] \ .$$

We will also consider game $\mathsf{COR\text{-}RO}$ in the quantum random oracle model, adversary A is given quantum access to random oracle G. (Again, how to model quantum access will be made explicit in Section 1.3.)

Note that our game-based definition in the standard model is a special case of the one in the random oracle model, where the number of random oracle queries is zero.

Clearly, δ-worst-case correctness of a scheme implies its δ-average-case correctness.[4] In Lemma 1.1.14, we establish that the other direction holds if and only if all messages induce decryption failure with the same probability. [5]

[3] For an example in which the random oracle queries actually matter in the context of correctness, we refer to Theorem 2.1.2 in Section 2.1.1, see page 62.

[4] This can be easily verified by applying the law of total probability in order to upper bound δ_{ac} in terms of δ_{wc}.

[5] Again, Lemma 1.1.14 can be easily verified by applying the law of total probability to both terms and comparing the difference in the respective summands.

```
GAME COR:
$(pk, sk) \leftarrow \mathsf{KG}$
$\mathfrak{L}_{\mathcal{M}} \leftarrow \mathsf{A}(sk, pk)$
for $m \in \mathfrak{L}_{\mathcal{M}}$
    $c \leftarrow \mathsf{Enc}(pk, m)$
    if $\mathsf{Dec}(sk, c) \neq m$
        return 1
return 0
```

```
GAME COR-RO:
$(pk, sk) \leftarrow \mathsf{KG}$
$\mathfrak{L}_{\mathcal{M}} \leftarrow \mathsf{A}^{\mathsf{G}}(sk, pk)$
for $m \in \mathfrak{L}_{\mathcal{M}}$
    $c \leftarrow \mathsf{Enc}(pk, m)$
    if $\mathsf{Dec}(sk, c) \neq m$
        return 1
return 0
```

Fig. 1.5: Correctness games COR for PKE in the standard model (left), and COR-RO for PKE defined relative to a random oracle G (right).

Lemma 1.1.14. Let PKE be a PKE scheme. We have that $\delta_{\text{wc}} \leq \delta_{\text{ac}}$ (and they hence are equal) if and only if for all key pairs $(pk, sk) \in \text{supp}(\mathsf{KG})$ there exists a constant $\lambda(pk, sk)$ such that for all messages $m \in \mathcal{M}$,

$$\Pr[\mathsf{Dec}(sk, c) \neq m] = \lambda(pk, sk) \ ,$$

where the probability is taken over $c \leftarrow \mathsf{Enc}(pk, m)$.

In some of our results, we will require from a deterministic scheme that decrypting a ciphertext c results in $\perp$ unless re-encrypting results in the ciphertext c again. We formalise this requirement in the next definition.

Definition 1.1.15 (Rigidity [BP18])**.** We call a deterministic public-key encryption scheme PKE *rigid* if for all key pairs $(pk, sk) \in \text{supp}(\mathsf{KG})$ and all ciphertexts $c \in \mathcal{C}$ it holds that $\mathsf{Dec}(sk, c) = \perp$ or $\mathsf{Enc}(pk, \mathsf{Dec}(sk, c)) = c$.

For perfectly correct schemes, requiring rigidity is equivalent to requiring that for all key pairs $(pk, sk) \in \text{supp}(\mathsf{KG})$, and all ciphertexts c that do not lie in the range of $\mathsf{Enc}(pk, -)$, we have that $\mathsf{Dec}(sk, c) = \perp$.

Correctness of KEMs. We furthermore define correctness for key encapsulation mechanisms. Throughout this thesis, we will only have to account for decryption failure that is triggered by honestly generated ciphertexts. This is the reason why we only give the KEM equivalent of average-case correctness, and simply call it δ-correctness.

Definition 1.1.16. We call KEM *δ-correct* if

$$\Pr\left[\mathsf{Decaps}(sk, c) \neq K\right] \leq \delta \ ,$$

where the probability is taken over $(pk, sk) \leftarrow \mathsf{KG}$ and $(K, c) \leftarrow \mathsf{Encaps}(pk)$.

1.2 Identification Schemes and Signatures

1.2.1 Identification Schemes

We now define syntax and security of canonical identification schemes [AABN02], which are interactive protocols that allow a party (called the prover) to prove to another party (called the verifier) knowledge of a secret key. A natural application of identification schemes is in user identification, however, we will see in Section 1.2.2 that they also give rise to signature schemes.

Definition 1.2.1 (Identification Schemes)**.** An identification scheme ID is defined as a collection of algorithms $\mathsf{ID} = (\mathsf{IG}, \mathsf{Commit}, \mathsf{Respond}, \mathsf{V})$.

- The key generation algorithm IG returns a key pair (pk, sk). We assume that pk defines a challenge space $\mathcal{C}$, a commitment space $\mathcal{W}$ and a response space $\mathcal{Z}$.
- Commit takes as input the secret key sk and returns a commitment $w \in \mathcal{W}$ and a state st.
- $\mathsf{Respond}$ takes as input the secret key sk, a commitment w, a challenge c, and a state st, and returns a response $z \in \mathcal{Z} \cup \{\perp\}$, where $\perp \notin \mathcal{Z}$ is a special symbol indicating failure.
- The deterministic verification algorithm $\mathsf{V}(pk, w, c, z)$ returns 1 (accept) or 0 (reject).

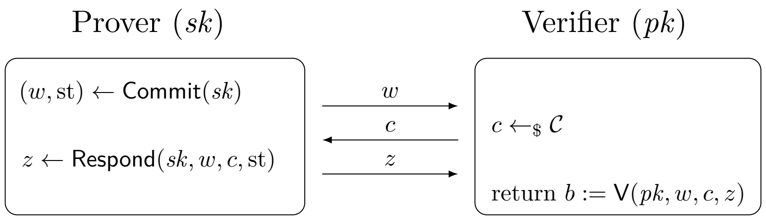

Prover (sk)
Verifier (pk)
$(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
$z \leftarrow \mathsf{Respond}(sk, w, c, \mathrm{st})$
w
c
z
$c \leftarrow_{\$} \mathcal{C}$
return $b := \mathsf{V}(pk, w, c, z)$

Fig. 1.6: Proving knowledge of a secret key via an identification scheme.

Note that during one of our application examples (i.e., in Section 3.3), we will define the response algorithm such that it does not explicitly take a commitment w as input. If needed, it can be assumed that st contains a copy of w.

A *transcript* is a triplet $trans = (w, c, z) \in \mathcal{W} \times \mathcal{C} \times \mathcal{Z}$. It is called *valid* (with respect to public key pk) if $\mathsf{V}(pk, w, c, z) = 1$. In Figure 1.7, we define transcript algorithm

$\mathsf{getTrans}$ that returns the transcript $trans = (w, c, z)$ of an honest interaction between prover and verifier. We furthermore define another transcript algorithm $\mathsf{getTransChall}$ that returns a transcript for a fixed challenge c.

```
Oracle getTrans(sk)
  $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
  $c \leftarrow_\$ \mathcal{C}$
  $z \leftarrow \mathsf{Respond}(sk, w, c, \mathrm{st})$
  return $(w, c, z)$

Oracle getTransChall(sk, c)
  $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
  $z \leftarrow \mathsf{Respond}(sk, w, c, \mathrm{st})$
  return $(w, c, z)$
```

Fig. 1.7: Generating honest transcripts with algorithm $\mathsf{getTrans}$, and generating challenge-dependent transcripts with algorithm $\mathsf{getTransChall}$.

Definition 1.2.2 (Commitment Entropy)**.** We define

$$\gamma(\mathsf{Commit}) := \mathbb{E}[\max_{w'}] \Pr[w = w'] \ ,$$

where the expectation is taken over $(pk, sk) \leftarrow \mathsf{IG}$, and the probability is taken over $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$.

In one of our results (see Section 3.3), the $\mathsf{Respond}$ algorithm is required to reject whenever its challenge input c is malformed. As observed in [AOTZ20], this additional requirement is not too severe, since most practical implementations perform a sanity check on c. We will call this property validity awareness.

Definition 1.2.3 (Validity Awareness)**.** We say that ID is *validity aware* if for all challenges $c \notin \mathcal{C}$, $\mathsf{Respond}(sk, w, c, \mathrm{st}) = \bot$.

(Special) Honest-Verifier Zero Knowledge. We will now formalise the property that honest transcripts do not provide too much information to an adversary, which is also called Honest-Verifier Zero Knowledge (HVZK). Intuitively, an identification scheme is HVZK if honest transcripts can be simulated without knowledge of the secret key. To make this description more formal, we will first introduce the notion of an HVZK simulator. Our definition comes in two flavours: While a (standard) HVZK simulator generates transcripts relative to the public key, a *special* HVZK simulator generates transcripts relative to (the public key and) a particular challenge.

Definition 1.2.4 ((Special) HVZK Simulator)**.** An HVZK *simulator* is an algorithm Sim that takes as input the public key pk and outputs a transcript (w, c, z). A *special* HVZK *simulator* is an algorithm Sim that takes as input the public key pk and a challenge c and outputs a transcript (w, c, z).

Statistical HVZK. Next, we will now reintroduce the definition of statistical HVZK [GMR85], and the definition of *special* statistical HVZK from [AOTZ20].

Definition 1.2.5 ((Special) Statistical HVZK)**.** Assume that ID comes with an HVZK simulator Sim. We say that ID is Δ_{HVZK}*-statistical* HVZK if for any key pair $(pk, sk) \in \mathrm{supp}(\mathsf{IG})$, the distribution of $(w, c, z) \leftarrow \mathsf{Sim}(pk)$ has statistical distance at most Δ_{HVZK} from an honest transcript $(w, c, z) \leftarrow \mathsf{getTrans}(sk)$.

To define *special* statistical HVZK, assume that ID comes with a special HVZK simulator Sim. We say that ID is Δ_{sHVZK}*-statistical* sHVZK if for any key pair $(pk, sk) \in \mathrm{supp}(\mathsf{IG})$ and any challenge $c \in \mathcal{C}$, the distribution of $(w, c, z) \leftarrow \mathsf{getTransChall}(sk, c)$ and the distribution of $(w, c, z) \leftarrow \mathsf{Sim}(pk, c)$ have statistical distance at most Δ_{sHVZK}.

Computational HVZK for multiple transcripts. In our security proofs, we will have to argue that collections of honestly generated transcripts are indistinguishable from collections of simulated ones. Since it is not always clear whether computational HVZK [FS87] implies computational HVZK for *multiple* transcripts, we extend our definition, accordingly: In the multi-HVZK game, the adversary obtains a collection of transcripts (rather than a single one). Similarly, we extend the definition of *special* computational HVZK from [AOTZ20].

Definition 1.2.6 ((Special) Computational Multi-HVZK)**.** Assume that ID comes with an HVZK simulator Sim. We define multi-HVZK games t-HVZK as in Figure 1.8, and the multi-HVZK *advantage function of an adversary* A *against* ID as

$$\mathrm{Adv}_{\mathsf{ID}}^{t\text{-}\mathsf{HVZK}}(\mathsf{A}) := \left|\Pr[t\text{-}\mathsf{HVZK}_{1\,\mathsf{ID}}^{\mathsf{A}} \Rightarrow 1] - \Pr[t\text{-}\mathsf{HVZK}_{0\,\mathsf{ID}}^{\mathsf{A}} \Rightarrow 1]\right| \ .$$

To define *special* HVZK, assume that ID comes with a special HVZK simulator Sim. We define multi-sHVZK game as in Figure 1.8, and the multi-sHVZK *advantage function of an adversary* A *against* ID as

$$\mathrm{Adv}_{\mathsf{ID}}^{t\text{-}\mathsf{sHVZK}}(\mathsf{A}) := \left|\Pr[t\text{-}\mathsf{sHVZK}_{1\,\mathsf{ID}}^{\mathsf{A}} \Rightarrow 1] - \Pr[t\text{-}\mathsf{sHVZK}_{0\,\mathsf{ID}}^{\mathsf{A}} \Rightarrow 1]\right| \ .$$

Following [AOTZ20], we now define subset-revealing identification schemes. Intuitively, an identification scheme is subset-revealing if $\mathsf{Respond}$ responds to a challenge by revealing parts of the state that was computed by Commit, and the response does not depend on sk.

```
GAME t-HVZK_b
$(pk, sk) \leftarrow \mathsf{IG}$
for $i \in \{1, \cdots, t\}$
    $trans_i^0 \leftarrow \mathsf{getTrans}(sk)$
    $trans_i^1 \leftarrow \mathsf{Sim}(pk)$
$b' \leftarrow \mathsf{A}(pk, (trans_i^b)_{1 \leq i \leq t})$
return $b'$
```

```
GAME t-sHVZK_b
$(pk, sk) \leftarrow \mathsf{IG}$
$i := 1$
$b' \leftarrow \mathsf{A}^{\text{GETTRANS}}(pk)$
return $b'$
```

```
GETTRANS(c)
if $i > t$ return $\perp$
$i := i + 1$
$trans^0 \leftarrow \mathsf{getTransChall}(sk, c)$
$trans^1 \leftarrow \mathsf{Sim}(pk, c)$
return $trans^b$
```

Fig. 1.8: Multi-HVZK game and multi-sHVZK game for ID. Both games are defined relative to bit $b \in \{0, 1\}$, and to the number t of transcripts the adversary is given.

Definition 1.2.7 (Subset-revealing ID scheme)**.** Let $\mathsf{ID} = (\mathsf{IG}, \mathsf{Commit}, \mathsf{Respond}, \mathsf{V})$ be an identification protocol. We say that ID is *subset-revealing* if there exists an algorithm DeriveSet that takes as input a challenge c and returns a set $I \subset \{1, \cdots, N\}$ for some natural number N such that for any key pair $(pk, sk) \in \text{supp}(\mathsf{IG})$, any tuple $(w, \text{st}) \in \text{supp}(\mathsf{Commit}(sk))$, and any challenge $c \in \mathcal{C}$ it holds that $\mathsf{Respond}(sk, c, \text{st}) = (\text{st}_i)_{i \in I}$, where $I = \mathsf{DeriveSet}(c)$ and $(\text{st}_n)_{n \in \{1, \cdots, N\}} := \text{st}$.

1.2.2 Signature Schemes

Signature schemes are a digital analogue to handwritten signatures. Similarly to a handwritten one, it should be hard to forge a digital signature, while it should be publicly verifiable that a signature was created by the signer. A generic strategy to obtain a digital signatures is to apply the Fiat-Shamir heuristic which we will reintroduce below. We will also reintroduce the hedging paradigm that was devised in order to immunise signature schemes against biased randomness.

Definition 1.2.8 (Signature Scheme)**.** A digital signature scheme SIG is defined as a triple of algorithms $\mathsf{SIG} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Vrfy})$.

- The key generation algorithm KG returns a key pair (pk, sk). We assume that pk defines the message space $\mathcal{M}$.
- The signing algorithm $\mathsf{Sign}(sk, m)$ returns a signature σ.
- The deterministic verification algorithm $\mathsf{Vrfy}(pk, m, \sigma)$ returns 1 (accept) or 0 (reject).

SECURITY. Following [GMR88], we define UnForgeability under Chosen Message Attacks (UF-CMA) and UnForgeability under Chosen Message Attacks with 0 queries to the signing oracle ($\mathsf{UF\text{-}CMA_0}$, also known as $\mathsf{UF\text{-}KOA}$) advantage functions of an adversary A against SIG as

$$\mathrm{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{SIG}}(\mathsf{A}) := \Pr[\mathsf{UF\text{-}CMA}^{\mathsf{A}}_{\mathsf{SIG}} \Rightarrow 1]$$

and

$$\mathrm{Adv}^{\mathsf{UF\text{-}CMA_0}}_{\mathsf{SIG}}(\mathsf{A}) := \Pr[\mathsf{UF\text{-}CMA_0}^{\mathsf{A}}_{\mathsf{SIG}} \Rightarrow 1] \ ,$$

where games $\mathsf{UF\text{-}CMA}$ and $\mathsf{UF\text{-}CMA_0}$ are given in Figure 1.9.

```
Game $\boxed{\mathsf{UF\text{-}CMA}}$ $\mathsf{UF\text{-}CMA_0}$
$(pk, sk) \leftarrow \mathsf{IG}$
$\boxed{(m^*, \sigma^*) \leftarrow \mathsf{A}^{\mathrm{SIGN}}(pk)}$
$(m^*, \sigma^*) \leftarrow \mathsf{A}(pk)$
if $m^* \in \mathfrak{L}_{\mathcal{M}}$ return $0$
return $\mathsf{Vrfy}(pk, m^*, \sigma^*)$

SIGN($m$)
$\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
$\sigma \leftarrow \mathsf{Sign}(sk, m)$
return $\sigma$
```

Fig. 1.9: Games $\mathsf{UF\text{-}CMA}$ and $\mathsf{UF\text{-}CMA_0}$ for SIG.

The Fiat-Shamir transform [FS87]. To an identification scheme $\mathsf{ID} = (\mathsf{IG}, \mathsf{Commit}, \mathsf{Respond}, \mathsf{V})$ with commitment space $\mathcal{W}$, and random oracle $\mathsf{H} : \mathcal{W} \times \mathcal{M} \rightarrow \mathcal{C}$ for some message space $\mathcal{M}$, we associate

$$\mathsf{FS}[\mathsf{ID}, \mathsf{H}] := \mathsf{SIG} := (\mathsf{IG}, \mathsf{Sign}, \mathsf{Vrfy}) \ ,$$

where algorithms Sign and Vrfy of SIG are defined in Figure 1.10.

In this thesis, we will also consider the modified Fiat-Shamir transform, in which lines 02 and 05 are replaced with $c := \mathsf{H}(w, m, pk)$.

```
Sign($sk, m$)
01 $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
02 $c := \mathsf{H}(w, m)$
03 $z \leftarrow \mathsf{Respond}(sk, w, c, \mathrm{st})$
04 return $\sigma := (w, z)$

Vrfy($pk, m, \sigma = (w, z)$)
05 $c := \mathsf{H}(w, m)$
06 return $\mathsf{V}(pk, w, c, z)$
```

Fig. 1.10: Signing and verification algorithms of $\mathsf{SIG} = \mathsf{FS}[\mathsf{ID}, \mathsf{G}]$.

Hedged signature schemes [BPS16, BT16]. Let $\mathcal{N}$ be any nonce space. To a signature scheme $\mathsf{SIG} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Vrfy})$ with secret key space $\mathcal{SK}$ and signing randomness space $\mathcal{R}_{\mathsf{Sign}}$, and random oracle $\mathsf{G} : \mathcal{SK} \times \mathcal{M} \times \mathcal{N} \rightarrow \mathcal{R}_{\mathsf{Sign}}$, we associate

$$\mathsf{R2H}[\mathsf{SIG}, \mathsf{G}] := \mathsf{SIG}' := (\mathsf{KG}, \mathsf{Sign}', \mathsf{Vrfy}) \ ,$$

where the signing algorithm Sign' of SIG' is defined in Figure 1.11.

```
$\mathsf{Sign}'(sk, m; n)$
$r := \mathsf{G}(sk, m, n)$
$\sigma := \mathsf{Sign}(sk, m; r)$
return $\sigma$
```

Fig. 1.11: Hedged signing algorithm Sign' of $\mathsf{SIG}' = \mathsf{R2H}[\mathsf{SIG}, \mathsf{G}]$.

1.3 Quantum Computation and the Quantum Random Oracle Model (QROM)

In this section, we briefly give some background on quantum computation and current techniques used in the context of security proofs in the quantum random oracle. For a more detailed discussion of the basic definitions introduced in Section 1.3.1, we refer to [NC11].

1.3.1 Basic Definitions

Qubits and quantum registers. We will treat a *qubit* as a vector $|\varphi\rangle$ that lies in the unity sphere of $\mathbb{C}^2$. In more detail, qubits $|\varphi\rangle$ are a linear combination $|\varphi\rangle = \alpha_0 \cdot |0\rangle + \alpha_1 \cdot |1\rangle$ of the two *basis states* (vectors) $|0\rangle$ and $|1\rangle$, such that $\alpha_b \in \mathbb{C}$ for both bits b, and it additionally holds that $|\alpha_0|^2 + |\alpha_1|^2 = 1$. (By $|\cdot|$ we denote the Euclidean norm.)

Similarly, we will treat an n qubit quantum register (or quantum bitstring) as a vector $|\varphi\rangle$ that lies in the unity sphere of $\mathbb{C}^{2^n}$. I.e., $|\varphi\rangle$ is a linear combination $|\varphi\rangle := \sum_{x\in\{0,1\}^n} \alpha_x \cdot |x\rangle$ of the *basis states* (vectors) $|x\rangle$, where $\alpha_x \in \mathbb{C}$ for all $x \in \{0,1\}^n$, and it additionally holds that $\sum_{x\in\{0,1\}^n} |\alpha_x|^2 = 1$.

We call the basis $\{|x\rangle\}_{x\in\{0,1\}^n}$ the *standard orthonormal computational basis.* Classical bitstrings can be interpreted as quantum bitstrings via the mapping $(x' \mapsto \sum_{x\in\{0,1\}^n} [\![x = x']\!] \cdot |x\rangle$.

The quantum bitstring $|\varphi\rangle$ is said to be *in superposition*, and we will say that $|\varphi\rangle$ *contains (classical) bitstring* x if $\alpha_x \neq 0$.

At times, we will generalise the definitions above such that they account for any finite set X, as opposed to only considering a set $\{0,1\}^n$ of bitstrings. Any finite set

X gives rise to its own canonical complex Hilbert space $\mathcal{H}_X$ (also called state space): Space $\mathcal{H}_X$ is completely described by taking the set X as its set of base vectors. Again, a quantum register is a vector that lies in the unity sphere of $\mathcal{H}_X$. We use a subscript to indicate that a vector $|\psi\rangle$ is the state of a quantum register X with Hilbert space $\mathcal{H}_X$, i.e., we write $|\psi\rangle_X$. In this generalised setting, we will at times consider systems that comprise more than one quantum register. E.g., the system might comprise one register with respect to some finite base set X (e.g., an input register), and one register with respect to some finite base set Y (e.g., an output register). Registers X and Y give rise to Hilbert spaces $\mathcal{H}_X$ and $\mathcal{H}_Y$, and the composite system state space is defined as $\mathcal{H}_{X\times Y} := \mathcal{H}_X \otimes \mathcal{H}_Y$. Where it helps simplify notation, we take the liberty to reorder registers, keeping track of them using register subscripts. The base states of $\mathcal{H}_{X\times Y}$ hence are all vectors $|x,y\rangle_{X\times Y} := |x\rangle_X \otimes |y\rangle_Y$.

For a vector $|\psi\rangle \in \mathcal{H}$, we denote the standard Euclidean norm by $\| \, |\psi\rangle \, \|$. Similarly, M_X indicates that a matrix M acting on $\mathcal{H}$ is considered as acting on register X. The only other norm we will require is the trace norm. For a matrix M acting on $\mathcal{H}$, the trace norm $\|M\|_1$ is defined as the sum of the singular values of M.

Given a quantum state $|\varphi\rangle$, we denote by $|\varphi\rangle\!\langle\varphi|$ the orthogonal projection onto the subspace spanned by $|\varphi\rangle$.

One important quantum operation is the quantum extension of the classical CNOT. This is a unitary matrix CNOT acting on two qubits, i.e. on the vector space $\mathbb{C}^2 \otimes \mathbb{C}^2$, as $\mathrm{CNOT}\,|b_1\rangle\,|b_2\rangle = |b_1\rangle\,|b_2 \oplus b_1\rangle$. We sometimes subscript a CNOT gate with control register X and target register Y with $X:Y$, and extend this notation to the case where many CNOT gates are applied, i.e. $\mathrm{CNOT}^{\otimes n}_{X:Y}$ means a CNOT gate is applied to the i-th qubit of the n-qubit registers X and Y for each $i = 1, ..., n$ with the qubits in X being the controls and the ones in Y the targets.

Measurements. Qubits can be measured with respect to a basis. In this thesis, we will usually consider measurements in the standard orthonormal computational basis. We will denote measuring some quantum register $|\varphi\rangle$ in the standard orthonormal computational basis by $x \leftarrow \mathsf{Measure}(|\varphi\rangle)$. After the measurement, the amplitudes are *collapsed*, meaning that all amplitudes are switched to 0 except for the one that belongs to the measurement outcome x, which will be switched to 1. Measuring a quantum register $|\varphi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x \cdot |x\rangle$ will result in the classical bitstring x with probability $|\alpha_x|^2$.

Theorem 1.3.1. (Principle of deferred measurement [NC11, Sct. 4.4]) Measurements can always be moved from an intermediate stage of a quantum circuit to the end of the circuit; if the measurement results are used at any stage of the circuit, then the classically controlled operations can be replaced by conditional quantum operations.

Quantum adversaries and the quantum random oracle model. We will consider security games in the quantum random oracle model (QROM) as counterparts to the respective games in the classical random oracle model. To be more precise, quantum adversaries will be given **quantum** access to all random oracles involved, and **classical** access to all other oracles (e.g., plaintext checking or decapsulation oracles), as the latter represent honest users.

To model quantum oracle access, let X and Y be finite sets, and let $\mathsf{O} : X \to Y$. Following [BDF+11, BBC+98], we usually model quantum access to O via oracle access to the unitary mapping

$$\begin{aligned} U_{\mathsf{O}} :\ & \mathcal{H}_{X\times Y} \longrightarrow \mathcal{H}_{X\times Y} \\ & |x\rangle_X |y\rangle_Y \longmapsto |x\rangle_X |y \oplus \mathsf{O}(x)\rangle_Y \quad . \end{aligned}$$

Furthermore, we model adversaries A with quantum access to O by a sequence U_1, U_{O}, U_2, U_{O}, $\cdots$, U_{O}, U_N, where $U_1, \cdots, U_N$ are unitary transformations. We write $\mathsf{A}^{|\mathsf{O}\rangle}$ to indicate that O is quantum-accessible (contrary to oracles which can only process classical bits).

Zhandry [Zha12b] proved that no quantum algorithm $\mathsf{A}^{|\mathsf{O}\rangle}$, issuing at most q quantum queries to O, can distinguish between a random function $\mathsf{O} : \{0,1\}^m \to \{0,1\}^n$ and a $2q$-wise independent function. This allows us to view quantum random oracles as polynomials of sufficiently large degree. I.e., we can replace quantum access to O with an oracle that evaluates a random polynomial f_{2q} of degree $2q$ over the finite field $\mathbb{F}_{2^n}$. The running time to evaluate f_{2q} is linear in q. In this thesis, we will use this observation in the context of security reductions, where a quantum reduction B against some underlying security property executes a quantum adversary $\mathsf{A}^{|\mathsf{O}\rangle}$, issuing at most q queries to O, against the target security property. The running time of B is $\text{Time}(\mathsf{B}) = \text{Time}(\mathsf{A}) + q \cdot \text{Time}(\mathsf{Sim}(\mathsf{O}))$, where $\text{Time}(\mathsf{Sim}(\mathsf{O}))$ denotes the time it takes to simulate quantum access to O. When using a $2q$-wise independent function in order to (information-theoretically) simulate quantum access to O, we obtain that the running time of B is $\text{Time}(\mathsf{B}) = \text{Time}(\mathsf{A}) + q \cdot \text{Time}(f_{2q})$, and the time $\text{Time}(f_{2q})$ to evaluate f_{2q} is linear in q. Following [SXY18] and [KLS18], we make use of the fact that the second summand of $\text{Time}(\mathsf{B})$, which is quadratic in q, can be further reduced to a term linear in q: As [SXY18] observe, it can be viewed as natural to model B with access to its own additional external oracle O. In this case, the second summand is reduced to $q \cdot \text{Time}(\mathsf{O})$, where $\text{Time}(\mathsf{O})$ now simply denotes the time it takes to evaluate O. Assuming that evaluating a random oracle takes one time unit, we then have $\text{Time}(\mathsf{B}) = \text{Time}(\mathsf{A}) + q$. Throughout this thesis, we will hence endow quantum adversaries B with access to an additional external oracle, and simplify the bounds on running time by dropping Time($\mathsf{Sim}(\mathsf{O})$) altogether.

1.3.2 Query Extraction Arguments: One-Way to Hiding

Extractability. In (classical) random oracle proofs, a common proof strategy is to make use of the observation that A can not distinguish a particular random oracle value $\mathsf{O}(x^*)$ from random unless A queries O on x^*. When aiming to reduce the success probability of A by giving a reduction B to some underlying problem P, learning preimage x^* can often be exploited to generate a solution to P. In the classical random oracle model, this approach usually is formalised by "identical-until-query" games: The game keeps track of all random oracle queries and raises flag QUERY if O ever is queried on x^*. The probability that QUERY occurs is then upper bounded in terms of the success probability of some reduction B, which simulates the game to A until (and wins with some probability if) QUERY occurs. (The probability of B winning if QUERY occurs can depend on the number of oracle queries to O.) This property of the classical random oracle model is sometimes called *extractability*.

In the quantum random oracle model, however, things become slightly more involved: For an oracle that is quantum-accessible, the value x^* is contained in a superposition $|\varphi\rangle$ of (potentially exponentially many) base states. In order to extract a classical value from $|\varphi\rangle$, the reduction would need to measure $|\varphi\rangle$, but this measurement would let the amplitudes of $|\varphi\rangle$ collapse. Answering A's query after execution of a measurement hence would not be true to the original game, and might derail its behaviour.

Quantum counterparts of extractability. Luckily, there have been recent results that showed how to give a quantum equivalent of "random-until-query"-like arguments. Sticking to the established naming convention, we will gather those arguments under the umbrella term "one-way to hiding", as they prove $\mathsf{O}(x^*)$ to be hidden unless it can be inverted.

The first one-way to hiding argument ("original one-way to hiding") was given by Unruh in [Unr14b]. Accounting for the fact that the reduction cannot exploit random oracle queries without potentially disturbing A's behaviour, the argument describes an extractor algorithm that randomly commits to one of the quantum queries a priori (i.e., before starting to execute A) and runs A only until this query, which then is measured to extract a classical value. With a probability dependent on the number of random oracle queries, the extractor will find x^* if A behaves differently when receiving random input (instead of receiving $(x^*, \mathsf{O}(x^*))$). We will formalise this argument in Theorem 1.3.2 below. In [HHK17], we used a slightly more general variant of this argument (called "algorithmic one-way to hiding"), where adversaries receive additional input, and might furthermore have access to additional oracles (which might be defined relative to O). We will formalise this generalisation in Section 2.2. While being the conceptually simplest solutions, the resulting upper bounds are far from tight: They lose a factor of q (the number of random oracle queries), and additionally suffer from a quadratic loss in the extraction advantage.

Given that one-way to hiding is such an important tool when trying to lift random oracle proofs to the quantum random oracle model, several recent publications [AHU19, BHH$^+$19, KSS$^+$20] improved the upper bound by deploying more sophisticated proof techniques. All improvements, however, come with some additional technical restrictions.

In [AHU19], Ambainis et al. generalised original one-way to hiding for multiple preimages and reduced the factor q to $q^{\frac{1}{2}}$, by a technique they called "semi-classical one-way to hiding". Semi-classical oracles are oracles that measure queries, but only partially. (To be more precise, it is only measured whether the query contained a desired preimage. In particular, this means that if a query does not contain such a preimage, the measurement has no effect on the query whatsoever.) We will formalise this argument in Theorems 1.3.3 to 1.3.5 below, in which A has quantum access to either a random oracle O_1, or an oracle O_2 such that O_2 coincides with O_1 everywhere but on some subset $S \subset X$. (In our use cases, S will simply be the set that contains one particular x^*.) It was then shown that there exists an extractor which will find some $x \in S$ (again, with probability dependent on q) if A behaves differently when accessing O_2. The additional technical requirement is that for each classical value x contained in a query to $\mathsf{O} \in \{\mathsf{O}_1, \mathsf{O}_2\}$, the extractor must be able to recognise whether $x \in S$ (Otherwise, another factor of $q^{\frac{1}{2}}$ is lost, and we end up with a factor of q again.)

Subsequent to [AHU19], Bindel et al. [BHH$^+$19] were able to remove the factor q altogether with a technique called "double-sided one-way to hiding". The name was chosen as the extractor must be able to evaluate both oracles O_1 and O_2.

Finally, Kuchta et al. [KSS$^+$20] built on the work of [BHH$^+$19], leading to the same additional requirement. By also considering A's internal workings (as opposed to simply measuring one of the random oracle queries), a reduction was achieved that removed the square root, albeit at the price of reintroducing factor q. The additional requirement is as in [BHH$^+$19].

We give a simplified overview in Figure 1.12, where ϵ denotes the success probability of the respective extractor, or, in the case of [AHU19], the probability of measuring that a query contains the desired preimage. We disregard small constant factors.

Variant	bound (simplified)	additional restrictions
Original [Unr14b]	$q\sqrt{\epsilon}$	
Semi-classical [AHU19]	$\sqrt{q\epsilon}$	✓
Double-sided [BHH+19]	$\sqrt{\epsilon}$	✓
MRM [KSS+20]	$q\epsilon$	✓

Fig. 1.12: Comparison of known one-way to hiding variants.

Original one-way to hiding. We will now restate "original one-way to hiding" [Unr14b, Lemma 5].

Theorem 1.3.2. (Original one-way to hiding) Let $\mathsf{O} : \{0,1\}^n \to \{0,1\}^m$ be a random oracle, and let A be a quantum algorithm with binary output, issuing at most q_{O} (quantum) queries to O. Let B be an oracle algorithm that, on input x^*, does the following: Pick $i \leftarrow_\$ \{1, \cdots, q_{\mathsf{O}}\}$ and $y^* \leftarrow_\$ \{0,1\}^m$, run $\mathsf{A}^{|\mathsf{O}\rangle}(x^*, y^*)$ until (just before) the i-th query, measure the argument of the query in the computational basis, and output the measurement outcome. (When A makes less than i queries, B outputs $\perp \notin \{0,1\}^n$.) Then

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq 2q_{\mathsf{O}} \cdot \sqrt{p_{\text{FIND}}} \ ,$$

where games G_b (for bit b) are defined below and

$$p_{\text{FIND}} := \Pr[x' = x^*]$$

with the probability taken over $x^* \leftarrow_\$ \{0,1\}^n$, and $x' \leftarrow \mathsf{B}^{|\mathsf{O}\rangle}(x^*)$.

```
GAME $G_b$
$x^* \leftarrow_\$ \{0,1\}^n$
$y_0^* := \mathsf{O}(x)$, $y_1^* \leftarrow_\$ \{0,1\}^m$
$b' \leftarrow \mathsf{A}^{|\mathsf{O}\rangle}(x^*, y_b^*)$
```

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

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

The following theorem is a simplification of several statements given in [AHU19, Thm. 1: "Semi-classical O2H"]. While [AHU19] consider adversaries that might execute parallel oracle invocations, and therefore differentiate between query depth d and number of queries q, we use the upper bound $q \geq d$ for the sake of simplicity.

Theorem 1.3.3 (Distinguishing)**.** Let $S \subset X$ be random. Let $\mathsf{O}_1, \mathsf{O}_2 \in Y^X$ be random functions such that $\mathsf{O}_1(x) = \mathsf{O}_2(x)$ for all $x \in X \setminus S$, and let z be a random bitstring. (S, O_1, O_2, and z may have an arbitrary joint distribution.) For $i \in \{1, 2\}$, let

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

and let

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

Then the probability of FIND is the same for both oracles, i.e.,

$$p_{\text{FIND}} = \Pr[b \leftarrow \mathsf{A}^{|\mathsf{O}_2 \setminus \mathsf{S}\rangle}(z) : \text{FIND}] \ . \tag{1.1}$$

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

$$|p_1 - p_2| \leq 2 \cdot \sqrt{(q+1) \cdot p_{\text{FIND}}} \ . \tag{1.2}$$

It furthermore holds that

$$|\sqrt{p_1} - \sqrt{p_2'}| \leq \sqrt{(q+1) \cdot p_{\text{FIND}}} \ , \tag{1.3}$$

where we let

$$p_2' := \Pr[1 \leftarrow \mathsf{A}^{|\mathsf{O} \setminus \mathsf{S}\rangle}(z) \wedge \neg\text{FIND}]$$

for either oracle $\mathsf{O} \in \{\mathsf{O}_1, \mathsf{O}_2\}$.

Unfortunately, it is not always enough to argue that FIND occurs during a game, as our reductions will not always know the set according to which the oracle is punctured. To give an upper bound for p_{FIND} in this setting, we now restate a simplified version of

[AHU19, Thm. 2: "Search in semi-classical oracle"] and [AHU19, Cor. 1].

Theorem 1.3.4 (Extracting unknown preimages)**.** Let $S \subset X$ be random, let O be a random function, and let z be a random bitstring. (S, O, and z may have an arbitrary joint distribution.) Let

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

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

$$p_{\text{FIND}} \leq 4q \cdot \Pr[x \leftarrow \mathsf{B}(z) : x \in S] \ , \tag{1.4}$$

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

If $S := \{x^*\}$ for $x^* \leftarrow_\$ X$, and x^* and z are independent, we have that

$$p_{\text{FIND}} \leq \frac{4q}{|X|} \ . \tag{1.5}$$

We will also use the following theorem, which generalises Theorem 1.3.2 by considering arbitrary $(S, \mathsf{O}_1, \mathsf{O}_2, z)$.

Theorem 1.3.5. [AHU19, Thm. 3: "One-way to hiding, probabilities"] We have that

$$|p_1 - p_2| \leq 2q \cdot \sqrt{\Pr[x \leftarrow \mathsf{B}(z) : x \in S]} \ ,$$

where p_1 and p_2 are defined as in Theorem 1.3.3, and B is defined as in Theorem 1.3.4.

1.3.3 Quantum Search and Distinguishing Problems

In our security proofs, we will sometimes argue that it is hard to find elements of a finite set X that fulfill certain properties, where any element x fulfills this property only with some bounded probability. We will formalise this argument as follows: For $\lambda \in [0, 1]$, let B_λ denote the Bernoulli distribution, i.e., $\Pr[b = 1] = \lambda$ for the bit $b \leftarrow B_\lambda$. Let X be some finite set, and let $F : X \rightarrow \{0, 1\}$ be a random function such that for each $x \in X$, $F(x)$ is distributed according to B_λ. The Generic quantum Search Problem (GSP, [HRS16, Thrm. 1], [Zha12a]) is to find an $x \in X$ satisfying $F(x) = 1$, given quantum access to F.

We will also at times argue that it is hard to detect when random sampling of $x \leftarrow_{\$} X$ is replaced with sampling x in a way such that no element carries the desired property any more. To this end, we will make use of the Generic quantum Distinguishing Problem (GDP): The generic quantum distinguishing problem ([ARU14, Lemma 37: "Preimage search in a random function"] [HRS16, Lem. 3]) is to distinguish quantum access to F from quantum access to the zero function.

To be more precise, we will need slight variations of GSP and GDP: In the generic quantum search (distinguishing) problem with Bounded probabilities GSPB (GDPB), the Bernoulli parameter $\lambda(x)$ may depend on x, but it is upper bounded by a global λ.

Lemma 1.3.6. (Generic search (distinguishing) problem with bounded probabilities) Let X be a finite set, and let $\lambda \in [0, 1]$. For any (unbounded, quantum) algorithm A issuing at most q quantum queries to F,

$$\Pr[\mathsf{GSPB}_{\lambda}^{\mathsf{A}} \Rightarrow 1] \leq 8 \cdot \lambda \cdot (q+1)^2 \ , \tag{1.6}$$

where game GSPB_{λ} is defined in Figure 1.13. Furthermore, for any (unbounded, quantum) algorithm A issuing at most q quantum queries to F,

$$|\Pr[\mathsf{GDPB}_{\lambda,0}^{\mathsf{A}} \Rightarrow 1] - \Pr[\mathsf{GDPB}_{\lambda,1}^{\mathsf{A}} \Rightarrow 1]| \leq 8 \cdot \lambda \cdot (q+1)^2 \quad , \tag{1.7}$$

where games $\mathsf{GDPB}_{\lambda,b}$ (for bit $b \in \{0, 1\}$) are also defined in Figure 1.13.

```
GAME GSPB_λ
$(\lambda(x))_{x \in X} \leftarrow \mathsf{A}_1$
if $\exists x \in X$ s.t. $\lambda(x) > \lambda$
   return 0
for all $x \in X$
   $F(x) \leftarrow B_{\lambda(x)}$
$x \leftarrow \mathsf{A}_2^{|F\rangle}$
return $F(x)$

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

Fig. 1.13: Generic search game GSPB_{λ} and generic distinguishing games $\mathsf{GDPB}_{\lambda,b}$ with bounded maximal Bernoulli parameter $\lambda \in [0, 1]$.

Note that Equation (1.6) was already proven in [KLS17]. Similar to the proof of Equation (1.6), we showed in [HKSU20] that the bound in Equation (1.7) can be reduced to the known bound on GDP by artificially increasing the Bernoulli parameter in order to obtain the dependence on each x.

In Section 3.1, we will consider a scenario win which a register is either in a state $|\gamma_0\rangle$ or in a state $|\gamma_1\rangle$. As a helper theorem, we will now recall that the optimal distinguishing advantage between those states can be upper bounded in terms of the trace distance of their density matrices $|\gamma_0\rangle\langle\gamma_0|$ and $|\gamma_1\rangle\langle\gamma_1|$. Theorem 1.3.7 is a straightforward corollary of [NC11, Thm. 9.1].

Theorem 1.3.7. (Optimal distinguishing advantage) For any quantum algorithm A, we have that

$$|\Pr[1 \leftarrow \mathsf{A}(|\gamma_0\rangle)] - \Pr[1 \leftarrow \mathsf{A}(|\gamma_1\rangle)]| \leq \frac{1}{2} \left\| |\gamma_0\rangle\langle\gamma_0| - |\gamma_1\rangle\langle\gamma_1| \right\|_1 \quad .$$

1.3.4 The Superposition Oracle Formalism

In this section, we present the (simplest version of the) superposition oracle formalism that was introduced in [Zha19], and which we will need in Chapter 3. Superposition oracles are a perfectly correct method to simulate a quantum-accessible random oracle $\mathsf{O} : \{0,1\}^n \to \{0,1\}^m$. Different variants of the superposition oracle have different additional features that make them more useful than the quantum-accessible random oracle model itself. In Chapter 3, we will be considered with the question whether reprogramming of a quantum-accessible oracle O can be done without causing too much of a difference in the behaviour of algorithms that access O, and since we will prove an information-theoretical bound, we only need the simplest version of the superposition oracle formalism.[6] In this basic form, there are three conceptual steps underlying the construction of the superposition oracle, with the last one being key to its usefulness in our analysis of the question stated above:

1. For each input value $x \in \{0,1\}^n$, $\mathsf{O}(x)$ is a random variable uniformly distributed on $\{0,1\}^m$. This variable can be sampled by performing a computational basis measurement of the uniform superposition

$$|\phi_0\rangle := 2^{-m/2} \sum_{y \in \{0,1\}^m} |y\rangle \quad .$$

For a function $\mathsf{O} : \{0,1\}^n \to \{0,1\}^m$, we can store the string $\mathsf{O}(x)$ in a quantum

[6] We will use the fact that in the superposition oracle formalism, reprogramming can be implemented directly by replacing a part of the quantum state that is held by the oracle, instead of using a simulator that sits between the original oracle and the adversary. We will not be bothered with the time that it takes to simulate such a quantum-accessible random oracle, since we will use Theorem 1.3.7 to prove a bound that holds for any adversary, regardless of its running time.

register F_x. In fact, to sample $\mathsf{O}(x)$, we can prepare a register F_x in the state $|\phi_0\rangle$, perform a computational basis measurement, and keep the collapsed post-measurement state. The outcome y of the measurement corresponds to the projector $|y\rangle\langle y|$, and a post-measurement state proportional to

$$|y\rangle\langle y|\,|\phi_0\rangle = 2^{-\frac{m}{2}}\,|y\rangle\,.$$

The complete quantum-accessible random oracle $\mathsf{O} : \{0,1\}^n \to \{0,1\}^m$ can accordingly be sampled by measuring a uniform superposition of all possible value tables: O can be sampled by preparing a register F_x for each input value x as described above, resulting in a state

$$|\Phi_0\rangle_F = \bigotimes_{x\in\{0,1\}^n} |\phi_0\rangle_{F_x} \quad ,$$

where $F = F_{0^n}F_{0^{n-1}1}...F_{1^n}$, followed by a measurement of F in the computational basis.

2. We will later argue that we can delay the measurement by which O is determined until after the algorithm accessing O has finished, leading to the question how to process the adversary's queries while the random oracle still is in superposition. On a high level, we do the same thing as in the usual quantum random oracle formalism, meaning that we answer to quantum queries on registers X and Y by $\oplus$-ing the oracle values into the Y-register, only that now the oracle values are still in superposition: Say a query is issued on $|x\rangle_X\,|\psi\rangle_Y$, where X is the input register and Y is the output register. We answer to the query using quantum CNOT gates, i.e., we answer with a superposition oracle unitary O_{XYF} that acts on the registers X, Y and the oracle register F such that

$$O_{XYF}\,|x\rangle\langle x|_X = |x\rangle\langle x|_X \otimes \left(\mathrm{CNOT}^{\otimes m}\right)_{F_x:Y} \quad .$$

Before we proceed to the third conceptual step, we provide some more details on how transformation O_{XYF} works for readers who are not yet familiar with this formalism. Readers who are already familiar with this formalism can proceed to page 50.

We will now make explicit how unitary O_{XYF} acts on XYF when a query is on a base state $|x^*, y^*\rangle$. Since O_{XYF} is linear, it is thereby completely described.

As a warm-up, let us first assume that registers F_{x^*} and $F_{x^{*c}}$ can be separated in the current oracle state $|\Phi\rangle_F$, i.e., we have that $|\Phi\rangle_F = |\phi\rangle_{F_{x^*}} \otimes |\phi'\rangle_{F_{x^{*c}}}$ for some state $|\phi\rangle_{F_{x^*}} = \sum_{y\in\{0,1\}^m} \alpha_y\,|y\rangle$ and some state $|\phi'\rangle_{F_{x^{*c}}}$. In this case, O_{XYF} works as follows: For any base state $|y\rangle = |y_1 \cdots y_n\rangle$ contained in $|\phi\rangle_{F_{x^*}}$, we can compute

$$|y, y^* \oplus y\rangle_{F_{x^*},Y} = |y, y_1^* \oplus y_1, \cdots, y_n^* \oplus y_n\rangle_{F_{x^*},Y} = \mathrm{CNOT}^{\otimes m} |y, y^*\rangle_{F_{x^*}:Y} \quad .$$

The joint query-oracle state transitions to

$$\begin{aligned} O_{XYF} |x^*, y^*\rangle_{X,Y} \otimes |\Phi\rangle_F &= O_{XYF} |x^*, y^*\rangle_{X,Y} \otimes (|\phi\rangle_{F_{x^*}} \otimes |\phi'\rangle_{F_{x^*c}}) \\ &= |x^*\rangle_X \otimes \left((\mathrm{CNOT}^{\otimes m})_{F_{x^*}:Y} |y^*\rangle_Y \otimes |\phi\rangle_{F_{x^*}}\right) \otimes |\phi'\rangle_{F_{x^*c}} \\ &= |x^*\rangle_X \otimes \left(\sum_{y\in\{0,1\}^m} \alpha_y |y^* \oplus y\rangle_Y \otimes |y\rangle_{F_{x^*}}\right) \otimes |\phi'\rangle_{F_{x^*c}} \quad , \end{aligned}$$

and the oracle answers with the X- and Y-register.

In the general case where the registers cannot necessarily be separated, we have that $\{|y\rangle_{F_{x^*}}\}_y$ constitutes a basis for register F_{x^*}, and that $\{b_i\}_{i\in I} := \{|y\rangle_{F_x}\}_{x\neq x^*,y}$ constitutes a basis for F_{x^*c}. We can therefore write the current oracle state in this basis as $|\Phi\rangle_F = \sum_{y\in\{0,1\}^m,i\in I} \alpha_{y,i} |y\rangle_{F_{x^*}} \otimes |b_i\rangle_{F_{x^*c}}$. When queried on base state $|x^*, y^*\rangle$, the joint state transitions to (and the oracle answers with the X- and Y-register of)

$$\begin{aligned} O_{XYF} |x^*, y^*\rangle_{X,Y} \otimes |\Phi\rangle_F &= O_{XYF} |x^*, y^*\rangle_{X,Y} \otimes (\sum_{y\in\{0,1\}^m,i\in I} \alpha_{y,i} |y\rangle_{F_{x^*}} \otimes |b_i\rangle_{F_{x^*c}}) \\ &= |x^*\rangle_X \otimes \sum_{y\in\{0,1\}^m,i\in I} \alpha_{y,i} |y^* \oplus y\rangle_Y \otimes |y\rangle_{F_{x^*}} \otimes |b_i\rangle_{F_{x^*c}} \quad . \end{aligned}$$

Note that with its queries, the querying algorithm becomes entangled with the oracle state: As an easy example, assume that the algorithm issues a first oracle query on some base state $|x^*, y^*\rangle$. The initial oracle state $|\Phi_0\rangle_F$ is separable into $|\phi_0\rangle_F = |\phi_0\rangle_{F_x^*} \otimes |\phi_0\rangle_{F_{x^*c}}^{\otimes 2^n-1}$, and we have that

$$O_{XYF} |x^*, y^*\rangle_{X,Y} \otimes |\Phi_0\rangle_F = |x^*\rangle_X \otimes \left(2^{-m/2} \sum_{y\in\{0,1\}^m} |y^* \oplus y\rangle_Y \otimes |y\rangle_{F_{x^*}}\right) \otimes |\phi_0\rangle_{F_{x^*c}}^{\otimes 2^n-1} \quad .$$

Since the state $2^{-m/2} \sum_{y\in\{0,1\}^m} |y^* \oplus y, y\rangle$ is not separable, the Y-register has become entangled with the F_{x^*}-register.

In particular, this means that the algorithm that queries the oracle can influence the oracle by measuring the responses it obtains. In the example above, a measurement of the Y-register will also let the F_{x^*}-register collapse. This should not be too surprising, though: With its measurement, the adversary has learned the value of $\mathsf{O}(x^*)$ and it is hence determined from now on.

We now proceed to the last conceptual step of the superposition oracle formalism.

3. Since the matrices $|y\rangle\langle y|_{F_x}$ and $\left(\mathrm{CNOT}^{\otimes m}\right)_{F_x:Y}$ commute, we can delay the measurement that performs the sampling of the random oracle until after the querying algorithm was executed. Queries are hence answered using the unitary O_{XYF}, acting on oracle registers F_x that are all initialised in the uniform superposition state $|\phi_0\rangle$, and only after the querying algorithm has finished, the register F is measured to obtain the concrete random function O.

The formalism, in a nutshell. Due to the observations above, we can equivalently implement a quantum-accessible oracle for a random function $\mathsf{O} : \{0,1\}^n \to \{0,1\}^m$ as follows:

- Initialise: Prepare the initial state

$$|\Phi_0\rangle_F = \bigotimes_{x\in\{0,1\}^n} |\phi_0\rangle_{F_x} .$$

- Responding to oracle queries: A quantum query on registers X and Y is answered using the unitary O_{XYF}.

- Post-processing: Register F is measured to obtain a random function O.

The last step can be omitted whenever the function O is not needed for the evaluation of the adversary's success.

In Chapter 3, we will also need the following helper lemma, which is a reformulation of [AMRS20, Lemma 2]. In a way, Lemma 1.3.8 generalises that when accessing a classical oracle, an adversary issuing q many queries learns at most q oracle values. While superposition queries can of course result in answers that contain many oracle values at once, and while the adversary can become entangled with the oracle, Lemma 1.3.8 states that the joint state of adversary and oracle still is a superposition of states in which at most q many outputs (registers F_x) were touched.

Lemma 1.3.8. Let $|\psi_q\rangle_{AF}$ be the joint adversary-oracle state after an adversary A has made q queries to the superposition oracle with register F. Define the low hamming-weight subspace for register F as the span of all vectors of which at most q many subregisters F_x are not in state $|\phi_0\rangle_{F_x}$. Then $|\psi_q\rangle_{AF}$ is supported by the tensor product of A's registers and this subspace. I.e., $|\psi_q\rangle_{AF}$ can be written as

$$|\psi_q\rangle_{AF} = \sum_{\substack{S\subset\{0,1\}^n\\ |S|\leq q}} |\psi_q^{(S)}\rangle_{AF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n-|S|)}\right)_{F_{S^c}} \quad ,$$

where for any set $R = \{x_1, x_2, ..., x_{|R|}\} \subset \{0,1\}^n$ we have defined $F_R = F_{x_1} F_{x_2} ... F_{x_{|R|}}$, and $|\psi_q^{(S)}\rangle_{AF_S}$ are some unnormalised vectors such that $\langle \phi_0|_{F_x} |\psi_q^{(S)}\rangle_{AF_S} = 0$ for all $x \in S$.

Chapter 2

FO-like Transformations in the (Q)ROM

The notion of indistinguishability against chosen-ciphertext attacks (IND-CCA) [RS92] is now widely accepted as the standard security notion for public-key encryption schemes. While IND-CCA security is in many applications the desired notion of security, it is usually much more difficult to prove than passive (i.e., IND-CPA or OW) security. Thus, several transformations have been suggested that turn a public-key encryption scheme with weaker security properties into an IND-CCA one, generically.

For instance, the aforementioned Fujisaki-Okamoto (FO) transformation [FO99, FO13] yields a hybrid encryption scheme from combining any public-key encryption scheme with any symmetric encryption scheme. If the underlying public-key encryption scheme is OW secure, and the underlying symmetric scheme is one-time secure, then the hybrid scheme is IND-CCA secure in the random oracle model. The REACT and GEM transformations [OP01, CHJ$^+$02] are considerably simpler, but require the underlying asymmetric scheme to satisfy OW-PCA security (see Definition 1.1.5, page 26). A similar transformation was also implicitly used in the "Hashed ElGamal" encryption scheme by Abdalla et al. [ABR01].

Since real-world systems often work with hybrid encryption schemes that are derived from a KEM, the primary goal of Chapter 2 will be to construct IND-CCA secure KEMs. An important step towards this goal was already taken in [Den03], in which several KEMs were constructed and proven IND-CCA secure, including constructions that can be viewed as KEM variants of the FO transformation [Den03, Table 5] and the REACT/GEM transformations [Den03, Table 2].

Limitations of previously known results for FO, REACT, and GEM. Despite their versatility, these transformations exhibit a couple of disadvantages.

- **Non-Tightness.** The security reduction of the FO transformation in the random oracle model is non-tight, i.e., it loses a factor of q, where q is the number of random oracle queries. While the REACT/GEM transformations have a tight security reduction, they require the underlying encryption scheme to be OW-PCA secure. As observed by Peikert [Pei14], many natural lattice-based encryption scheme are not OW-PCA secure due to their decision/search equivalence, and it is not clear how to modify them to be so. In fact, the main technical difficulty is to build an IND-CPA or OW-PCA secure encryption scheme from a scheme that is OW secure such that the security reduction is tight.

- **Correctness errors.** The FO as well as the REACT/GEM transformations require the underlying asymmetric encryption scheme to be perfectly correct, i.e., not having a decryption error. In general, one cannot exclude the fact that decryption errors can be exploited by an active adversary, and in fact, recent research [DVV18, BS20] has confirmed this assessment. Dealing with imperfectly correct schemes hence proves of great importance, as many (but not all) practical lattice-based encryption schemes have a small correctness error, see, e.g., DXL [DXL12], Peikert [Pei14], BCNS [BCNS15], New Hope [ADPS16], Frodo [BCD$^+$16], Lizard [CKLS16], and Kyber [BDK$^+$17].

- **The security model.** The aforementioned constructions were proven secure in the random oracle model, meaning that the proof did not consider quantum adversaries. While a QROM proof for a variant of FO was given in [TU16], it was highly non-tight and again required the underlying scheme to be perfectly correct. Furthermore, the construction in [TU16] introduced a communication overhead of the length of the transmitted plaintext.

These deficiencies were of little or no concern when the FO and REACT/GEM transformations were originally devised. Today, however, we view these deficiencies as acute problems, due to the emergence of large-scale scenarios, in which tight security reductions offer security at significantly lower costs, the increased popularity of lattice-based schemes with correctness errors, and the potential threat of attackers with quantum capabilities.

Organisation of Chapter 2

In section Section 2.1, we offer a modular analysis of FO-like KEM transformations in the random oracle model. To put it more precisely, we provide fine-grained transformations that can be used to turn any passively secure PKE scheme into a KEM that is $\mathsf{IND\text{-}CCA}$ secure in two steps. Intuitively, the first step (called transformation T) achieves several intermediate notions (e.g. $\mathsf{OW\text{-}PCA}$) and tightness properties, depending on properties of the underlying scheme. The second step (called transformation U) comes in different variations, with all of them tightly achieving $\mathsf{IND\text{-}CCA}$ security. In particular, we show that T turns OW security into $\mathsf{OW\text{-}PCA}$ security, and we give a $\mathsf{OW\text{-}PCA} \rightarrow \mathsf{IND\text{-}CCA}$ variant of transformation U such that combining the two yields the KEM equivalent of the original FO transformation. The benefit of this modular approach is not only a conceptual simplification, but also that it results in a larger variety of possible combined transformations (with different requirements and properties). In particular, we can combine two results about our transformations T and U to obtain that the KEM equivalent of the original FO transformation yields $\mathsf{IND\text{-}CCA}$ security from $\mathsf{IND\text{-}CPA}$ security with a *tight* security reduction.

Recall that all previous work on FO-like transformations assumed the underlying scheme to be perfectly correct. It turns out that the possibility of correctness errors can indeed affect the level of the resulting scheme's security: All of our security bounds for the U-transforms include a term relative to the level of the underlying scheme's worst-case correctness (see Definition 1.1.12), and subsequent work [DVV18, BS20] has shown that this term is not a mere artifact of our proof strategy, but indeed reflects the possibility of an adversary learning secret information by deliberately triggering decryption failures.

In Section 2.2, we will revisit transformation T and generalise its analysis such that it accounts for adversaries with quantum capabilities, i.e., we prove its security in the quantum random oracle model. We furthermore show how to modify two of the U-variants such that they achieve $\mathsf{IND\text{-}CCA}$ security in the quantum random oracle model. The results presented in Section 2.1 and Section 2.2 are based on joint work with Dennis Hofheinz and Eike Kiltz, published in [HHK17].

The security proofs given in Section 2.2, however, are highly non-tight, as all of them invoke a quantum query extraction argument which comes with quadratic loss in the extraction probability, and linear loss in the number of oracle queries. Combining the security statement for transformation T with the security statement for either one of the U-transformations, hence, leads to quartic loss in the advantage, rendering the security statement quite meaningless for real-world applications. It was left as an open problem in [HHK17] to derive tighter security reductions. Furthermore, the quantum-secure U-variants from Section 2.2 introduced some communication overhead when compared to their classical counterparts from Section 2.1, which is why real-world implementations generally follow the framework from Section 2.1. In Section 2.3, we therefore revisit one of the combined KEMs from Section 2.1. We give a proof in the quantum random oracle model that comes with better $\mathsf{IND\text{-}CCA}$ bounds than that of Section 2.2, with respect to the underlying security assumption as well as with respect to the probability of decryption failure. Due to a proof of equivalence, the analysis provided in Section 2.3 immediately carries over to another KEM variant from Section 2.1. Unfortunately, the proof strategy does not carry over to the other two variants from Section 2.1. The result presented in Section 2.3 is based on joint work with Eike Kiltz, Sven Schäge, and Dominique Unruh [HKSU20], in which it was generalised to achieve post-quantum secure authenticated key exchange.

What all known security proofs for FO-like transformations have in common is that they require at least worst-case correctness, if not even perfect correctness. As already pointed out, our somewhat conservative definition of worst-case correctness indeed reveals a potential attack venue that should not be neglected. On the other hand, there exist practical schemes which do not naturally meet the requirement of worst-case correctness, while fulfilling the more traditional (and less conservative) definition of average-case correctness. In order to further broaden the applicability of known results for FO-like transformations, we therefore introduce a new transformation in Section 2.4 that achieves average-case correctness from worst-case correctness, and that integrates well into FO if the scheme either is one-way secure and randomness-recoverable, or if it satisfies either $\mathsf{IND\text{-}CPA}$ security or partial one-wayness.

Open problems. Subsequent to the publication of the results from Section 2.2, a lot of independent research [SXY18, JZC$^+$18, JZM19a, JZM19b, BHH$^+$19, KSS$^+$20] has been invested in the open problem (of proving tighter QROM bounds) that was stated in [HHK17]. As of today, there exist tight proofs for deterministic schemes that satisfy disjoint simulatability and perfect correctness (see [SXY18, Thm. 4.2] and [JZM19a, Thm. 5]). For deterministic schemes that do not satisfy disjoint simulatability, all known reductions are still at least quadratically loose [BHH$^+$19] or lose a factor of q [KSS$^+$20]. In the case that the underlying scheme is non-deterministic, all known bounds are roughly of the form[1] (or can be straightforwardly improved to) $\sqrt{q \cdot \text{Adv}_{\mathsf{PKE}}^{\mathsf{IND\text{-}CPA}}(\mathsf{B}) + q^2 \cdot \delta}$,

[1] Note that in order to keep the comparison lucid, we disregarded small constant factors and additional inherent summands that stem from, e.g., the search advantage corresponding to implicit rejection. We furthermore want to mention that [BHH$^+$19, KSS$^+$20] require the T-transformed deterministic scheme to be injective with overwhelming probability.

or respectively, $q \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B})} + q^2 \cdot \delta$, with the exception of [KSS+20] , in which the square root was avoided by proving a bound roughly of the form $q^2 \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B}) + q^2 \cdot \delta$. Given that there exist tight reductions in the ROM when starting from IND-CPA security, and that the last years have seen several refinements of the query extraction arguments that can be deployed, it remains an open question whether the non-tightness of the recently known bounds are proof artefacts, and the bounds hence could be improved further, or whether it is meaningful as it reflects the possibility of a quantum attack.

Furthermore, we note that all constructions either involve a re-encryption step or demand for additional requirements (like rigidity of a deterministic scheme, or OW security in the presence of a quantum-accessible plaintext checking oracle). We view it as an interesting question whether constructions can be found that do not involve a re-encryption step, without making additional requirements and without sacrificing the currently achieved level of efficiency. Given that key encapsulation mechanisms are usually more "lightweight" than public-key encryption schemes, we also view it as an interesting open question whether there exist direct transformations that only assume (and yield) KEMs, while preserving efficiency.

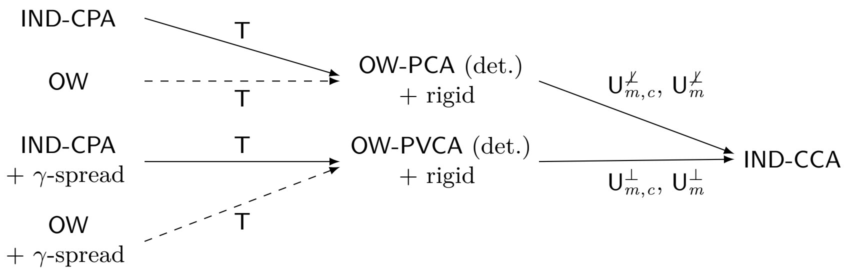
2.1 Modular Constructions in the ROM

In this section, we provide modular variants of the FO transformation that work in two steps, and prove their security in the random oracle model. An overview is given in Figure 2.1.

T: from OW to OW-PCA Security ("Derandomisation"+"re-encryption"). Transformation T is the Encrypt-with-Hash construction from [BBO07], originally proposed in [BHSV98, Sec. 5]: Starting from an encryption scheme PKE and a hash function G, we build an encryption scheme $\mathsf{PKE}' := \mathsf{T}[\mathsf{PKE}, \mathsf{G}]$ by defining

$$\mathsf{Enc}'(pk, m) := \mathsf{Enc}(pk, m; \mathsf{G}(m)) \ ,$$

where $\mathsf{G}(m)$ is used as the random coins for Enc. Note that Enc' is deterministic. $\mathsf{Dec}'(sk, c)$ first decrypts c into m' and rejects by returning $\perp$ if $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) \neq c$ ("re-encryption"). Modeling G as a random oracle, OW-PCA security of PKE' non-tightly reduces to OW security of PKE, and tightly reduces to IND-CPA security of PKE. If PKE furthermore is γ-spread (for sufficiently large γ), then PKE' is even OW-PVCA secure. Recall that OW-PVCA security is OW-PCA security, where the adversary is additionally given access to a validity oracle $\mathrm{VALID}(c)$ that checks c's validity (in the sense that it does not decrypt to $\perp$, see Definition 1.1.7 on page 27). Furthermore, it is


IND-CPA
OW
IND-CPA
+ γ-spread
OW
+ γ-spread
T
T
T
T
OW-PCA (det.)
+ rigid
OW-PVCA (det.)
+ rigid
$\mathsf{U}^{\not\perp}_{m,c}$, $\mathsf{U}^{\not\perp}_{m}$
$\mathsf{U}^{\perp}_{m,c}$, $\mathsf{U}^{\perp}_{m}$
IND-CCA

Transformation	**Security implication**	**ROM Tightness?**	**Requirements**
$\mathsf{PKE}' = \mathsf{T}[\mathsf{PKE}, \mathsf{G}]$ (§2.1.1)	IND-CPA ⇒ OW-PCA	✓	none
	IND-CPA ⇒ OW-PVCA	✓	γ-spread
	OW ⇒ OW-PCA	—	none
	OW ⇒ OW-PVCA	—	γ-spread
$\mathsf{KEM}^{\perp}_{m,c} = \mathsf{U}^{\perp}_{m,c}[\mathsf{PKE}', \mathsf{H}]$ (§2.1.2)	OW-PVCA ⇒ IND-CCA	✓	none
$\mathsf{KEM}^{\not\perp}_{m,c} = \mathsf{U}^{\not\perp}_{m,c}[\mathsf{PKE}', \mathsf{H}]$ (§2.1.2)	OW-PCA ⇒ IND-CCA	✓	none
$\mathsf{KEM}^{\perp}_{m} = \mathsf{U}^{\perp}_{m}[\mathsf{PKE}', \mathsf{H}]$ (§2.1.3)	OW-VCA ⇒ IND-CCA	✓	PKE' det. + rigid
$\mathsf{KEM}^{\not\perp}_{m} = \mathsf{U}^{\not\perp}_{m}[\mathsf{PKE}', \mathsf{H}]$ (§2.1.3)	OW ⇒ IND-CCA	✓	PKE' det. + rigid

Fig. 2.1: Our modular transformations. Top: solid arrows indicate tight reductions, dashed arrows indicate non-tight reductions. Bottom: properties of the transformations.

easy to verify that T achieves rigidity, meaning that ciphertexts either decrypt to $\perp$ or re-encryption yields the original ciphertext, see Definition 1.1.15 on page 32.

$\mathsf{U}^{\not\perp}_{m,c}$: FROM OW-PCA TO IND-CCA SECURITY ("HASHING" + "IMPLICIT REJECTION'). Starting from an encryption scheme PKE' and a hash function H, we build a key encapsulation mechanism $\mathsf{KEM}^{\not\perp}_{m,c} := \mathsf{U}^{\not\perp}_{m,c}[\mathsf{PKE}', \mathsf{H}]$ with "implicit rejection" by defining

$$\mathsf{Encaps}(pk) := (c \leftarrow \mathsf{Enc}'(pk, m), K := \mathsf{H}(m, c)), \tag{2.1}$$

where m is picked at random from the message space.

$$\mathsf{Decaps}^{\not\perp}_{m,c}(sk, c) := \begin{cases} \mathsf{H}(m, c) & m \neq \perp \\ \mathsf{H}(c, s) & m = \perp \end{cases}, \tag{2.2}$$

where $m := \mathsf{Dec}(sk, c)$, and s is a random seed which is contained in sk. Modeling H as a random oracle, IND-CCA security of $\mathsf{KEM}^{\not\perp}_{m,c}$ tightly reduces to OW-PCA security of PKE'. In the context of the FO transformation, implicit rejection was first introduced by Persichetti [Per12, Sec. 5.3].

$\mathsf{U}^{\perp}_{m,c}$: FROM OW-PVCA TO IND-CCA SECURITY ("HASHING"+ "EXPLICIT REJECTION'). Starting from an encryption scheme PKE' and a hash function H, we build a key encapsulation mechanism $\mathsf{KEM}^{\perp}_{m,c} = \mathsf{U}^{\perp}_{m,c}[\mathsf{PKE}', \mathsf{H}]$ with "explicit rejection" which differs from $\mathsf{KEM}^{\not\perp}_{m,c}$ only in decapsulation:

$$\mathsf{Decaps}^{\perp}_{m,c}(sk, c) := \begin{cases} \mathsf{H}(m, c) & m \neq \perp \\ \perp & m = \perp \end{cases}, \tag{2.3}$$

where $m := \mathsf{Dec}(sk, c)$. Modeling H as a random oracle, IND-CCA security of $\mathsf{KEM}^{\perp}_{m,c}$ tightly reduces to OW-PVCA security of PKE'. We remark that transformation $\mathsf{U}^{\perp}_{m,c}$ is essentially [Den03, Table 2], i.e., a KEM variant of the REACT/GEM transformations.

$\mathsf{U}^{\not\perp}_{m}$ ($\mathsf{U}^{\perp}_{m}$): FROM DETERMINISTIC OW (OW-VCA) TO IND-CCA SECURITY. We consider two more variants: Transformation $\mathsf{U}^{\not\perp}_{m}$ ($\mathsf{U}^{\perp}_{m}$) is a variant of $\mathsf{U}^{\not\perp}_{m,c}$ ($\mathsf{U}^{\perp}_{m,c}$), where $K = \mathsf{H}(m, c)$ from Equations (2.1)-(2.3) is replaced by $K = \mathsf{H}(m)$. We prove that in the random oracle model, IND-CCA security of $\mathsf{KEM}^{\not\perp}_{m} := \mathsf{U}^{\not\perp}_{m}[\mathsf{PKE}', \mathsf{H}]$ ($\mathsf{KEM}^{\perp}_{m} := \mathsf{U}^{\perp}_{m}[\mathsf{PKE}', \mathsf{H}]$) tightly reduces to OW (OW-VCA) security of PKE', if the encryption algorithm of PKE' is deterministic and the scheme is rigid. Recall that OW-VCA security is essentially OW security, where the adversary is additionally given access to a validity oracle VALID(c) that checks c's validity, see Definition 1.1.7 on page 27.

THE RESULTING FO TRANSFORMATIONS. Our final transformations $\mathsf{FO}^{\not\perp}_{m,c}$ ("FO with implicit rejection"), $\mathsf{FO}^{\perp}_{m,c}$ ("FO with explicit rejection"), $\mathsf{FO}^{\not\perp}_{m}$ ("FO with implicit rejection, $K = \mathsf{H}(m)$"), $\mathsf{FO}^{\perp}_{m}$ ("FO with explicit rejection, $K = \mathsf{H}(m)$") are defined in the following table. The column indicating ROM tightness refers to the case where the underlying scheme is assumed to be IND-CPA secure.

Transformation	**ROM Tightness?**	**Requirements**
$\mathsf{FO}^{\not\perp}_{m,c}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}^{\not\perp}_{m,c}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}]$	✓	none
$\mathsf{FO}^{\perp}_{m,c}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}^{\perp}_{m,c}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}]$	✓	γ-spread
$\mathsf{FO}^{\not\perp}_{m}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}^{\not\perp}_{m}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}]$	✓	none
$\mathsf{FO}^{\perp}_{m}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}^{\perp}_{m}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}]$	✓	γ-spread

As corollaries of our modular transformation we obtain that IND-CCA security of $\mathsf{FO}^{\not\perp}_{m,c}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}]$, $\mathsf{FO}^{\perp}_{m,c}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}]$, $\mathsf{FO}^{\not\perp}_{m}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}]$, and $\mathsf{FO}^{\perp}_{m}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}]$ non-tightly reduces to the OW security of PKE, and tightly reduces to the IND-CPA security of PKE, in the random oracle model. We remark that transformation $\mathsf{FO}^{\perp}_{m}$ essentially recovers a KEM variant [Den03, Table 5] of the original FO transformation [FO99]. Whereas the explicit rejection variants $\mathsf{FO}^{\perp}_{m,c}$ and $\mathsf{FO}^{\perp}_{m}$ require PKE to be γ-spread, there is no such requirement on the implicit rejection variants $\mathsf{FO}^{\not\perp}_{m,c}$ and $\mathsf{FO}^{\not\perp}_{m}$.

Correctness Error. We stress that all our security reductions also take non-perfect correctness into account. Finding the "right" definition of correctness that is achievable (say, by currently proposed lattice-based encryption schemes) and at the same time sufficient to prove security turned out to be a bit subtle. This is the reason why our definition of correctness (see Section 1.1.3, page 30) deviates from the ones that were previously used. The concrete bounds of $\mathsf{FO}^{\not\perp}_{m,c}$, $\mathsf{FO}^{\perp}_{m,c}$, $\mathsf{FO}^{\not\perp}_{m}$, and $\mathsf{FO}^{\perp}_{m}$ give guidance on the required correctness error of the underlying PKE scheme. Concretely, for "κ bits security", PKE requires a worst-case correctness error of at most $2^{-\kappa}$.

Example Instantiations. In the context of ElGamal encryption, one can apply $\{\mathsf{FO}^{\not\perp}_{m,c}, \mathsf{FO}^{\perp}_{m,c}, \mathsf{FO}^{\not\perp}_{m}, \mathsf{FO}^{\perp}_{m}\}$ to obtain the schemes of [KM03, BLK00, GMMV05] whose $\mathsf{IND\text{-}CCA}$ security non-tightly reduces to the CDH assumption, and tightly reduces to the DDH assumption. Alternatively, one can directly use $\mathsf{U}^{\not\perp}_{m,c}/\mathsf{U}^{\perp}_{m,c}$ to obtain the more efficient schemes of [OP01, CHJ$^+$02, ABR01, Sho04a] whose $\mathsf{IND\text{-}CCA}$ security tightly reduces to the gap-DH (a.k.a. strong CDH) assumption. In the context of deterministic encryption schemes, one can apply $\mathsf{U}^{\not\perp}_{m,c}/\mathsf{U}^{\perp}_{m,c}$ to obtain schemes mentioned in [Sho04a, Den03], whose $\mathsf{IND\text{-}CCA}$ security tightly reduces to one-way security. Finally, in the context of lattices-based encryption (e.g., [Reg05, LPR13]), one can apply $\mathsf{FO}^{\not\perp}_{m,c}$, $\mathsf{FO}^{\perp}_{m,c}$, $\mathsf{FO}^{\not\perp}_{m}$, $\mathsf{FO}^{\perp}_{m}$ to achieve $\mathsf{IND\text{-}CCA}$ security.

Related work. As already pointed out, $\mathsf{FO}^{\perp}_{m} = \mathsf{U}^{\perp}_{m} \circ \mathsf{T}$ is essentially a KEM variant of the Fujisaki-Okamoto transform from [Den03, Table 5]. Further, $\mathsf{U}^{\perp}_{m,c}$ is a KEM variant [Den03] of the GEM/REACT transform [OP01, CHJ$^+$02, ABR01]. Our modular view suggest that the FO transform implicitly contains the GEM/REACT transform, at least the proof technique. With this more general view, the FO transform and its variants remains the only known transformation from $\mathsf{IND\text{-}CPA}$ to $\mathsf{IND\text{-}CCA}$ security. It is an interesting open problem to come up with alternative transformations that get rid of derandomisation or dispense with re-encryption (while preserving efficiency). Note that for the ElGamal encryption scheme, the "twinning" technique [CKS08, CKS09] does exactly this, but it uses non-generic zero-knowledge proofs that are currently not available for all schemes (e.g., for lattice-based schemes).

In concurrent and independent work, [AOP$^+$17] considered the $\mathsf{IND\text{-}CCA}$ security of LIMA, which in our notation can be described as $\mathsf{FO}^{\perp}_{m}[\mathsf{RLWE}, \mathsf{G}, \mathsf{H}]$. Here RLWE is a specific encryption scheme based on lattices associated to polynomial rings from [LPR10], which is $\mathsf{IND\text{-}CPA}$ secure under the Ring-LWE assumption. As the main result, [AOP$^+$17] provides a tight reduction of LIMA's $\mathsf{IND\text{-}CCA}$ security to the Ring-LWE assumption, in the random oracle model. As observed in [AOP+17], the proof exploits "some weakly homomorphic properties enjoyed by the underlying encryption scheme", and therefore does not seem to be applicable to other schemes. The tight security reduction from Ring-LWE is recovered as a special case of our general security results on $\mathsf{FO}^{\perp}_{m}$. We note that the security reduction of [AOP+17] does not take the (non-zero) correctness error of RLWE into account.

Organisation of Section 2.1

In Section 2.1.1, we will define and analyse transformation T that turns any OW secure public-key encryption scheme PKE into a scheme PKE' that is $\mathsf{OW\text{-}PCA}$ secure. We achieve a tight reduction if PKE is $\mathsf{IND\text{-}CPA}$ secure. If PKE furthermore is γ-spread, then the resulting scheme even satisfies the stronger security notion of $\mathsf{OW\text{-}PVCA}$ security.

Next, in Sections 2.1.2 and 2.1.3, we will introduce transformations $\mathsf{U}^{\perp}_{m,c}$, $\mathsf{U}^{\not\perp}_{m,c}$ ($\mathsf{U}^{\perp}_{m}$, $\mathsf{U}^{\not\perp}_{m}$) that transform any $\mathsf{OW\text{-}PVCA}$ ($\mathsf{OW\text{-}PCA}$) secure encryption scheme PKE' into an $\mathsf{IND\text{-}CCA}$ secure KEM. The security reductions are tight. Transformations $\mathsf{U}^{\perp}_{m}$ and $\mathsf{U}^{\not\perp}_{m}$, however, can only be applied to deterministic encryption schemes that are rigid.

Combining T with $\{\mathsf{U}^{\perp}_{m,c}, \mathsf{U}^{\not\perp}_{m,c}, \mathsf{U}^{\perp}_{m}, \mathsf{U}^{\not\perp}_{m}\}$, we provide concrete bounds for the $\mathsf{IND\text{-}CCA}$ security of the resulting KEMs in Section 2.1.4.

2.1.1 Transformation T*: From* OW *and* $\mathsf{IND\text{-}CPA}$ *to* $\mathsf{OW\text{-}PVCA}$

Transformation T transforms a passively secure public-key encryption scheme into an $\mathsf{OW\text{-}PVCA}$ secure one, given that the underlying scheme is γ-spread for sufficiently large γ. If the underlying scheme is not γ-spread, then T still achieves $\mathsf{OW\text{-}PCA}$ security.

The Construction. To a public-key encryption scheme $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ and randomness space $\mathcal{R}$, and random oracle $\mathsf{G} : \mathcal{M} \to \mathcal{R}$, we associate $\mathsf{PKE}' := \mathsf{T}[\mathsf{PKE}, \mathsf{G}]$. The algorithms of $\mathsf{PKE}' = (\mathsf{KG}, \mathsf{Enc}', \mathsf{Dec}')$ are defined in Figure 2.2. Note that Enc' deterministically computers the ciphertext as $c := \mathsf{Enc}(pk, m; \mathsf{G}(m))$. It is easy to verify that PKE' is rigid (see Definition 1.1.15 on page 32): If $m' := \mathsf{Dec}'(sk, c) \neq \perp$, then $\mathsf{Enc}'(pk, m) = \mathsf{Enc}(pk, m'; \mathsf{G}(m')) = c$.

Correctness. The following theorem establishes that if PKE is δ-worst-case correct, then PKE' achieves game-based correctness (see Definition 1.1.13, page 31), in the random oracle model.

```
Enc'(pk, m)
    $c := \mathsf{Enc}(pk, m; \mathsf{G}(m))$
    return $c$

Dec'(sk, c)
    $m' := \mathsf{Dec}(sk, c)$.
    if $m' = \bot$ or $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) \neq c$
        return $\bot$
    else return $m'$
```

Fig. 2.2: OW-PVCA secure encryption scheme $\mathsf{PKE}' = \mathsf{T}[\mathsf{PKE}, \mathsf{G}]$ with deterministic encryption.

Theorem 2.1.1. If PKE is δ_{wc}-worst-case correct, and G is a random oracle, then for any adversary A returning a list of N many distinct messages it holds that

$$\mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{COR\text{-}RO}_N}(\mathsf{A}) \leq N \cdot \delta_{\text{wc}} \ ,$$

where game COR-RO is defined as in Figure 1.5.

Proof. To prove the upper bound, consider an adversary A, playing game COR-RO. We will call a message m *problematic* iff it exhibits a correctness error in PKE', i.e., if $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m; \mathsf{G}(m))) \neq m$. A wins if there exists at least one message in $\mathfrak{L}_{\mathcal{M}}$ that is problematic.

For a fixed key pair $(pk, sk) \in \text{supp}(\mathsf{KG})$, let $\delta'(pk, sk) := \max_{m \in \mathcal{M}} \Pr[\mathsf{Dec}(sk, c) \neq m]$, where the probability is taken over $c \leftarrow \mathsf{Enc}(pk, m)$. Since G outputs independent random values, any message $m \in \mathcal{M}$ is problematic with probability at most $\delta'(pk, sk)$, hence we can take the expectation over $(pk, sk) \leftarrow \mathsf{KG}$ and the union bound to obtain

$$\begin{aligned}\Pr[\mathsf{COR\text{-}RO}_{\mathsf{PKE}'}^{\mathsf{A}} \Rightarrow 1] &= \mathbb{E}\left[\Pr[\mathsf{COR\text{-}RO}_{\mathsf{PKE}'}^{\mathsf{A}} \Rightarrow 1 \mid (pk, sk)]\right] \\ &\leq \mathbb{E}\left[N \cdot \delta'(pk, sk)\right] = N \cdot \delta_{\text{wc}} \ .\end{aligned}$$

□

Non-tight security from OW. The following theorem establishes that OW-PVCA security of PKE'((see Definition 1.1.7, page 27) non-tightly reduces to the OW security of PKE, in the random oracle model, given that PKE is γ-spread (for sufficiently large γ). If PKE is not γ-spread, then PKE' is still OW-PCA secure.

Theorem 2.1.2 (PKE OW $\overset{\text{ROM}}{\Rightarrow}$ PKE' OW-PVCA)**.** Assume PKE to be δ_{wc}-worst-case correct and γ-spread. Then, for any OW-PVCA adversary A that issues at most q_{G} queries to random oracle G, q_{PCO} queries to plaintext checking oracle PCO, and q_{VALID} queries to validity checking oracle VALID, there exists an OW adversary B such that

$$\mathrm{Adv}_{\mathsf{PKE}'}^{\mathsf{OW\text{-}PVCA}}(\mathsf{A}) \leq (q_{\mathsf{G}} + q_{\text{PCO}} + 1) \cdot \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}) + (q_{\mathsf{G}} + q_{\text{PCO}}) \cdot \delta_{\text{wc}} + q_{\text{VALID}} \cdot 2^{-\gamma} \ ,$$

and the running time of B is about that of A.

The main idea of the proof is that since Enc' is deterministic, oracle PCO can be simulated by "re-encryption", and oracle VALID can be simulated by controlling the random oracles. Additional care has to be taken into account for the correctness error.

Per definition, OW-PCA security is OW-PVCA security with $q_{\mathrm{VALID}} := 0$ queries to the validity checking oracle. Hence, the bound of Theorem 2.1.2 in particular shows that PKE' is OW-PCA secure, without requiring PKE to be γ-spread.

Proof. To prove security, let A be an adversary against the OW-PVCA security of PKE', issuing at most q_{G} queries to G, at most q_{PCO} queries to PCO, and at most q_{VALID} queries to VALID. Consider the sequence of games given in Figure 2.3.

```
GAMES $G_0$ - $G_3$
$(pk, sk) \leftarrow \mathsf{KG}$
$m^* \leftarrow_\$ \mathcal{M}$
$c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$
$m' \leftarrow \mathsf{A}^{\mathsf{G},\mathrm{PCO},\mathrm{VALID}}(pk, c^*)$
return $[\![m' = m^*]\!]$

$\mathsf{G}(m)$
if $\exists r$ s. th.$(m, r) \in \mathfrak{L}_{\mathsf{G}}$
    return $r$
if $m = m^*$                                   //$G_3$
    QUERY := true                              //$G_3$
    abort                                      //$G_3$
$r \leftarrow_\$ \mathcal{R}$
$\mathfrak{L}_{\mathsf{G}} := \mathfrak{L}_{\mathsf{G}} \cup \{(m, r)\}$
return $r$

PCO$(m \in \mathcal{M}, c)$
$m' := \mathsf{Dec}(sk, c)$                                              //$G_0$-$G_1$
return $[\![m' = m$ and $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) = c]\!]$   //$G_0$-$G_1$
return $[\![\mathsf{Enc}(pk, m, \mathsf{G}(m)) = c]\!]$                  //$G_2$-$G_3$

VALID$(c \neq c^*)$
$m' := \mathsf{Dec}(sk, c)$                                                          //$G_0$-$G_1$
return $[\![m' \in \mathcal{M}]\!]$ and $[\![\mathsf{Enc}(pk, m'; \mathsf{G}(m')) = c]\!]$   //$G_0$
return $[\![\exists r$ s. th. $(m', r) \in \mathfrak{L}_{\mathsf{G}}$ and $\mathsf{Enc}(pk, m'; r) = c]\!]$   //$G_1$
return $[\![\exists (m, r) \in \mathfrak{L}_{\mathsf{G}}$ s. th. $\mathsf{Enc}(pk, m; r) = c]\!]$   //$G_2$-$G_3$
```

Fig. 2.3: Games G_0-G_3 for the proof of Theorem 2.1.2.

GAME G_0. This is the original OW-PVCA game. Random oracle queries are stored in set $\mathfrak{L}_{\mathsf{G}}$ with the convention that $\mathsf{G}(m) = r$ iff $(m, r) \in \mathfrak{L}_{\mathsf{G}}$. Hence,

$$\Pr[G_0^{\mathsf{A}} \Rightarrow 1] = \mathrm{Adv}_{\mathsf{PKE}'}^{\mathsf{OW\text{-}PVCA}}(\mathsf{A}) \ .$$

GAME G_1. In game G_1, the ciphertext validity oracle VALID$(c \neq c^*)$ is replaced with one that first computes $m' = \mathsf{Dec}(sk, c)$, and returns 1 iff there has occurred a previous query m' to G and re-encryption works, i.e., if there exists an r such that $(m', r) \in \mathfrak{L}_{\mathsf{G}}$ and $\mathsf{Enc}(pk, m'; r) = c$.

Consider a single query VALID(c) and define $m' := \mathsf{Dec}(sk, c)$. It is easy to verify that if VALID$(c) = 0$ in G_0, then VALID$(c) = 0$ also in G_1: If VALID$(c) = 0$ in G_0, then either $m' := \bot$ or $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) \neq c$. If $m' := \bot$, then no query to G on m' can occur. If $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) \neq c$, then VALID$(c)$ returns 0 in game G_1 even if a query to G on m' has occurred.

If VALID$(c) = 1$ in G_0, then VALID$(c) = 0$ in G_1 only if $\mathsf{G}(m')$ was not queried before. The adversary hence has to come up with a ciphertext c such that $c = \mathsf{Enc}(pk, m'; \mathsf{G}(m')) = c$, without knowledge of $\mathsf{G}(m')$. This happens with probability $2^{-\gamma}$, where γ is the parameter from the γ-spreadness of PKE.

By the union bound we obtain

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq q_{\text{VALID}} \cdot 2^{-\gamma}.$$

Game G_2. In game G_2, we replace plaintext checking oracle PCO(m, c) and ciphertext validity oracle VALID(c) by simulations that do not check anymore whether $m = \mathsf{Dec}(sk, c)$. We claim

$$|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \leq (q_{\mathsf{G}} + q_{\text{PCO}}) \cdot \delta_{\text{wc}} \ . \tag{2.4}$$

To show Equation (2.4), observe that both game G_1 and game G_2 make at most q (distinct) queries $\mathsf{G}(m_1), \ldots, \mathsf{G}(m_q)$ to G, where q counts both the explicit queries to G that are issued by B and the implicit queries that are triggered by PCO. Again, we call such a query $\mathsf{G}(m_i)$ *problematic* if it exhibits a correctness error in PKE', i.e., if $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m_i; \mathsf{G}(m_i))) \neq m_i$.

Consider a single query PCO(m, c) (VALID(c)) and define $m' := \mathsf{Dec}(sk, c)$. We first show that if G_1 answers a query to one of the oracles with 1, so does game G_2: If PCO$(m, c) = 1$ in game G_1, then $m = m'$ and $\mathsf{Enc}(pk, m; \mathsf{G}(m)) = \mathsf{Enc}(pk, m'; \mathsf{G}(m')) = c$, hence PCO$(m, c) = 1$ in game G_2. If VALID$(c) = 1$ in game G_1, then G was already queried on m' and $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) = c$, hence VALID$(c) = 1$ in game G_2.

We now show that if G_2 answers a query to one of the oracles with 1, so does game G_1, conditioned on the event that no random oracle query $\mathsf{G}(m_i)$ is problematic: If PCO$(m, c) = 1$ in game G_2, then $\mathsf{Enc}(pk, m; \mathsf{G}(m)) = c$. Since we condition on the event that no random oracle query is problematic, we have $m' = \mathsf{Dec}(sk, c) = m$, hence PCO$(m, c) = 1$ in game G_1. If VALID$(c) = 1$ in game G_2, then G was already queried on some m such that $\mathsf{Enc}(pk, m; \mathsf{G}(m)) = c$. Again, m can not be problematic and hence VALID$(c) = 1$ in game G_2.

We have shown that the two games can only differ if A submits a PCO query (m, c), or a VALID query c, together with a random oracle query $\mathsf{G}(m)$, such that $\mathsf{G}(m)$ is problematic and $c = \mathsf{Enc}(pk, m; \mathsf{G}(m))$. (In this case, G_1 will answer the query with 0, while G_2 will answer with 1.) Clearly, if A makes a problematic query, then there exists an adversary F that wins the correctness game $\mathsf{COR\text{-}RO}$ by returning to game $\mathsf{COR\text{-}RO}$ the list of all queries to G and PCO. Hence, the probability that at least one query $\mathsf{G}(m_i)$ is problematic is at most $(q_{\mathsf{G}} + q_{\mathrm{PCO}}) \cdot \delta_{\mathrm{wc}}$. This proves Equation (2.4).

Game G_3. In Game G_3, we add a flag QUERY in line 09 and abort when it is raised. Hence, G_2 and G_3 only differ if QUERY is raised, meaning that A made a query G on m^*, or, equivalently, $(m^*, \cdot) \in \mathfrak{L}_{\mathsf{G}}$. Due to the difference lemma (Lemma 1.0.1),

$$|\Pr[G_3^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \leq \Pr[\mathrm{QUERY}] \ .$$

We first bound $\Pr[G_3^{\mathsf{A}} \Rightarrow 1]$ by constructing an adversary B_1 in Figure 2.4 against the OW security of the original encryption scheme PKE. B_1 inputs (pk, c^*) for a random, unknown m^*, and $c^* \leftarrow \mathsf{Enc}(pk, m^*)$. Since G_3 aborts if G was queried on m^*, $c^* \leftarrow \mathsf{Enc}(pk, m^*)$ is indistinguishable from $c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$ unless G_3 aborts. Hence, B_1 perfectly simulates game G_3 for A and outputs $m' = m^*$ if A wins in game G_3.

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] = \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}_1) \ .$$

So far, we have established the bound

$$\mathrm{Adv}_{\mathsf{PKE}'}^{\mathsf{OW\text{-}PVCA}}(\mathsf{A}) \leq q_{\mathrm{VALID}} \cdot 2^{-\gamma} + (q_{\mathsf{G}} + q_{\mathrm{PCO}}) \cdot \delta_{\mathrm{wc}} + \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}_1) + \Pr[\mathrm{QUERY}] \ . \quad (2.5)$$

To finally upper bound $\Pr[\mathrm{QUERY}]$, in Figure 2.4 we construct an adversary B_2 against the OW security of the original encryption scheme PKE, that inputs $(pk, c^* \leftarrow \mathsf{Enc}(pk, m^*))$ and perfectly simulates game G_3 for A until QUERY occurs. If flag QUERY is set in G_3, then there exists en entry $(m^*, \cdot) \in \mathfrak{L}_{\mathsf{G}}$, and B_2 returns the correct $m' = m^*$ with probability $1/(q_{\mathsf{G}} + q_{\mathrm{PCO}})$. We just showed

$$\Pr[\mathrm{QUERY}] \leq (q_{\mathsf{G}} + q_{\mathrm{PCO}}) \cdot \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}_2) \ .$$

Combining the latter bound with Equation (2.5) and folding B_1 and B_2 into one single adversary B against the OW security of PKE yields the bound given in the theorem.

□

Tight security from $\mathsf{IND\text{-}CPA}$. Whereas the reduction to OW security in Theo-

```
$\mathsf{B}_1(pk, c^*)$
$m' \leftarrow \mathsf{A}^{\mathsf{G},\mathrm{PCO},\mathrm{VALID}}(pk, c^*)$
return $m'$

$\mathsf{B}_2(pk, c^*)$
$m \leftarrow \mathsf{A}^{\mathsf{G},\mathrm{PCO},\mathrm{VALID}}(pk, c^*)$
$(m', r') \leftarrow_\$ \mathfrak{L}_\mathsf{G}$
return $m'$
```

Fig. 2.4: Adversaries B_1 and B_1 against OW of PKE for the proof of Theorem 2.1.2. Oracles PCO, VALID are defined as in game G_3, and G is defined as in game G_2 of Figure 2.3.

rem 2.1.2 was non-tight, the following theorem establishes that $\mathsf{OW\text{-}PVCA}$ security of PKE' tightly reduces to $\mathsf{IND\text{-}CPA}$ security of PKE, in the random oracle model, given that PKE is γ-spread. If PKE is not γ-spread, then PKE' is still $\mathsf{OW\text{-}PCA}$ secure.

Theorem 2.1.3 (PKE $\mathsf{IND\text{-}CPA}$ $\overset{\text{ROM}}{\Rightarrow}$ PKE' $\mathsf{OW\text{-}PVCA}$)**.** Assume PKE to be δ_{wc}-worst-case correct and γ-spread. Then, for any $\mathsf{OW\text{-}PVCA}$ adversary A that issues at most q_G queries to the random oracle G, q_{PCO} queries to plaintext checking oracle PCO, and q_{VALID} queries to validity checking oracle VALID, there exists an $\mathsf{IND\text{-}CPA}$ adversary B such that

$$\mathrm{Adv}^{\mathsf{OW\text{-}PVCA}}_{\mathsf{PKE}'}(\mathsf{A}) \leq 3 \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B}) + (q_\mathsf{G} + q_{\mathrm{PCO}}) \cdot \delta_{\mathrm{wc}} + q_{\mathrm{VALID}} \cdot 2^{-\gamma} + \frac{2(q_\mathsf{G} + q_{\mathrm{PCO}}) + 1}{|\mathcal{M}|} \ ,$$

and the running time of B is about that of A.

With the same argument as in Theorem 2.1.2, a tight reduction to $\mathsf{OW\text{-}PCA}$ security is implied by Theorem 2.1.3 without requiring PKE to be γ-spread.

Proof. Considering the games of Figure 2.3 from the proof of Theorem 2.1.2, we obtain by Equation (2.5)

$$\begin{aligned}
\mathrm{Adv}^{\mathsf{OW\text{-}PVCA}}_{\mathsf{PKE}'}(\mathsf{A}) \quad &\leq \quad q_{\mathrm{VALID}} \cdot 2^{-\gamma} + (q_\mathsf{G} + q_{\mathrm{PCO}}) \cdot \delta_{\mathrm{wc}} + \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}_1) + \Pr[\mathrm{QUERY}] \\
&\leq \quad q_{\mathrm{VALID}} \cdot 2^{-\gamma} + (q_\mathsf{G} + q_{\mathrm{PCO}}) \cdot \delta_{\mathrm{wc}} + \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{C}) + \frac{1}{|\mathcal{M}|} \\
&\qquad + \Pr[\mathrm{QUERY}] \ ,
\end{aligned}$$

where the last inequation uses Lemma 1.1.6 (see page 26).

In Figure 2.5 we construct an adversary $\mathsf{D} = (\mathsf{D}_1, \mathsf{D}_2)$ against the $\mathsf{IND\text{-}CPA}$ security of the original encryption scheme PKE that wins if flag QUERY is set in G_3. The first adversary D_1 picks two random messages m_0^*, m_1^*. The second adversary D_2 inputs $(pk, c^* \leftarrow \mathsf{Enc}(pk, m_b^*), st)$, for an unknown random bit b, and runs A on (pk, c^*), simulating its view in game G_3. Note that by construction, message m_b^* is uniformly distributed.

```
$\mathsf{D}_1(pk)$
    $st := (m_0^*, m_1^*) \leftarrow_\$ \mathcal{M}^2$
    return $st$

$\mathsf{D}_2(pk, c^*, st)$
    $m' \leftarrow \mathsf{A}^{\mathsf{G},\mathrm{PCO},\mathrm{VALID}}(pk, c^*)$
    $b' := \begin{cases} 0 & |\mathfrak{L}_G(m_0^*)| > |\mathfrak{L}_G(m_1^*)| \\ 1 & |\mathfrak{L}_G(m_1^*)| > |\mathfrak{L}_G(m_0^*)| \\ \leftarrow_\$ \{0,1\} & \text{otherwise} \end{cases}$
    return $b'$
```

Fig. 2.5: Adversary $\mathsf{D} = (\mathsf{D}_1, \mathsf{D}_2)$ against $\mathsf{IND\text{-}CPA}$ for the proof of Theorem 2.1.3. For fixed $m \in \mathcal{M}$, $\mathfrak{L}_G(m)$ is the set of all $(m, r) \in \mathfrak{L}_G$ (if any). Oracles PCO and VALID are defined as in game G_3, and G is defined as in game G_2 of Figure 2.3.

Consider game $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ with random challenge bit b. Let $\mathsf{QUERY}_{\mathsf{G}}$ be the event that A queries random oracle G on m_{1-b}^*. Since m_{1-b}^* is uniformly distributed and independent from A's view, we have $\Pr[\mathsf{QUERY}_{\mathsf{G}}] \le (q_{\mathsf{G}} + q_{\mathrm{PCO}})/|\mathcal{M}|$. For the remainder of the proof we assume $\mathsf{QUERY}_{\mathsf{G}}$ did not happen, i.e., $|\mathfrak{L}_G(m_{1-b}^*)| = 0$.

If QUERY happens, then A queried the random oracle G on m_b^*, which implies that $|\mathfrak{L}_G(m_b^*)| > 0 = |\mathfrak{L}_G(m_{1-b}^*)|$ and therefore $b = b'$. If QUERY does not happen, then A did not query random oracle G on m_b^*. Hence, $|\mathfrak{L}_G(m_b^*)| = |\mathfrak{L}_G(m_{1-b}^*)| = 0$, and $\Pr[b = b'] = 1/2$ since B picks a random bit b'. Overall, we have

$$\begin{aligned} \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{IND\text{-}CPA}}(\mathsf{D}) + \frac{q_{\mathsf{G}} + q_{\mathrm{PCO}}}{|\mathcal{M}|} &\ge \left| \Pr[b = b'] - \frac{1}{2} \right| \\ &= \left| \Pr[\mathrm{QUERY}] + \frac{1}{2}\Pr[\neg\mathrm{QUERY}] - \frac{1}{2} \right| \\ &= \frac{1}{2}\Pr[\mathrm{QUERY}]. \end{aligned}$$

Folding C and D into one single $\mathsf{IND\text{-}CPA}$ adversary B yields the required bound of the theorem. □

Transformation U*: From* OW-PVCA *to* IND-CCA

In this section, we introduce four variants of transformation U, namely $\mathsf{U}_{m,c}^{\perp}$, $\mathsf{U}_{m,c}^{\not\perp}$, $\mathsf{U}_m^{\perp}$, $\mathsf{U}_m^{\not\perp}$, that convert a public-key encryption scheme PKE' into a key encapsulation mechanism KEM. Their differences are summarised in the following table.

Transformation	Rejection method	KEM key	PKE''s requirements
$\mathsf{U}^{\perp}_{m,c}$	explicit	$K = \mathsf{H}(m, c)$	$\mathsf{OW\text{-}PVCA}$
$\mathsf{U}^{\not\perp}_{m,c}$	implicit	$K = \mathsf{H}(m, c)$	$\mathsf{OW\text{-}PCA}$
$\mathsf{U}^{\perp}_{m}$	explicit	$K = \mathsf{H}(m)$	det. + $\mathsf{OW\text{-}VCA}$ + rigid
$\mathsf{U}^{\not\perp}_{m}$	implicit	$K = \mathsf{H}(m)$	det. + OW + rigid

Superscript $^{\perp}$ indicates that decapsulation rejects all inconsistent ciphertext by returning $\perp$ ("explicit rejection"), whereas $^{\not\perp}$ indicates that a pseudorandom key is returned instead ("implicit rejection"). A subscript m, c indicates that the KEM key is derived by hashing message m and ciphertext c, i.e., $K = \mathsf{H}(m, c)$, and a subscript m indicates that the KEM key is derived by only hashing message m, i.e., $K = \mathsf{H}(m)$.

2.1.2 *Transformation* $\mathsf{U}^{\perp}_{m,c}$ *(*$\mathsf{U}^{\not\perp}_{m,c}$*): From* OW-PVCA *(*PCA*) to* IND-CCA

Transformation $\mathsf{U}^{\perp}_{m,c}$: from OW-PVCA to IND-CCA

Transformation $\mathsf{U}^{\perp}_{m,c}$ transforms an OW-PVCA secure public-key encryption scheme into a key encapsulation mechanism that is IND-CCA secure. The $^{\perp}$ in $\mathsf{U}^{\perp}_{m,c}$ means that decapsulation of an invalid ciphertext results in the rejection symbol $\perp$ ("explicit rejection").

The Construction. To a public-key encryption scheme $\mathsf{PKE}' = (\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}')$ with message space $\mathcal{M}$, and a hash function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n$, we associate $\mathsf{KEM}^{\perp}_{m,c} = \mathsf{U}^{\perp}_{m,c}[\mathsf{PKE}', \mathsf{H}]$. The algorithms of $\mathsf{KEM}^{\perp}_{m,c} = (\mathsf{KG}', \mathsf{Encaps}, \mathsf{Decaps}^{\perp}_{m,c})$ are defined in Figure 2.6.

It is easy to verify that if PKE' is δ'_{ac}-average-case correct, then $\mathsf{KEM}^{\perp}_{m,c}$ is δ'_{ac}-correct.

```
Encaps($pk$)
$m \leftarrow_\$ \mathcal{M}$
$c \leftarrow \mathsf{Enc}'(pk, m)$
$K := \mathsf{H}(m, c)$
return $(K, c)$

Decaps$^{\perp}_{m,c}$($sk, c$)
$m' := \mathsf{Dec}'(sk, c)$
if $m' = \perp$ return $\perp$
else return
    $K := \mathsf{H}(m', c)$
```

Fig. 2.6: IND-CCA secure key encapsulation mechanism $\mathsf{KEM}^{\perp}_{m,c} = \mathsf{U}^{\perp}_{m,c}[\mathsf{PKE}', \mathsf{H}]$.

Security. The following theorem establishes that IND-CCA security of $\mathsf{KEM}^{\perp}_{m,c}$ tightly reduces to the OW-PVCA security of PKE', in the random oracle model.

```
GAMES $G_0$ - $G_2$
$(pk, sk) \leftarrow \mathsf{KG}'$
$m^* \leftarrow_\$ \mathcal{M}$
$c^* \leftarrow \mathsf{Enc}'(pk, m^*)$
$K_0^* := \mathsf{H}(m^*, c^*)$
$K_1^* \leftarrow_\$ \{0,1\}^n$
$b \leftarrow_\$ \{0,1\}$
$b' \leftarrow \mathsf{A}^{\textsc{Dec}^{\perp}_{m,c},\mathsf{H}}(pk, c^*, K_b^*)$
if $\mathsf{Dec}'(sk, c^*) \neq m^*$                    //$G_2$
    ERROR := true                                   //$G_2$
    abort                                           //$G_2$
return $[\![b' = b]\!]$
```

```
$\mathsf{H}(m, c)$
if $\exists K$ such that $(m, c, K) \in \mathfrak{L}_H$
    return $K$
$K \leftarrow_\$ \mathcal{K}$
if $\mathsf{Dec}'(sk, c) = m$                           //$G_1$-$G_2$
    if $c = c^*$                                        //$G_2$
        CHAL := true; abort                             //$G_2$
    if $\exists K'$ such that $(c, K') \in \mathfrak{L}_D$   //$G_1$-$G_2$
        $K := K'$                                       //$G_1$-$G_2$
    else                                                //$G_1$-$G_2$
        $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$   //$G_1$-$G_2$
$\mathfrak{L}_H := \mathfrak{L}_H \cup \{(m, c, K)\}$
return $K$
```

```
$\textsc{Dec}^{\perp}_{m,c}(c \neq c^*)$                   //$G_0$
$m' := \mathsf{Dec}'(sk, c)$
if $m' = \perp$ return $\perp$
return $K := \mathsf{H}(m', c)$
```

```
$\textsc{Dec}^{\perp}_{m,c}(c \neq c^*)$                   //$G_1$-$G_2$
if $\exists K$ s. th. $(c, K) \in \mathfrak{L}_D$
    return $K$
if $\mathsf{Dec}'(sk, c) \notin \mathcal{M}$
    return $\perp$
$K \leftarrow_\$ \mathcal{K}$
$\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$
return $K$
```

Fig. 2.7: Games G_0 - G_2 for the proof of Theorem 2.1.4.

Theorem 2.1.4 (PKE' $\mathsf{OW\text{-}PVCA}$ $\overset{\text{ROM}}{\Rightarrow}$ $\mathsf{KEM}^{\perp}_{m,c}$ $\mathsf{IND\text{-}CCA}$)**.** Assume PKE' to be δ'_{ac}-average-case correct. For any $\mathsf{IND\text{-}CCA}$ adversary A against $\mathsf{KEM}^{\perp}_{m,c}$, issuing at most q_D queries to the decapsulation oracle $\textsc{Dec}^{\perp}_{m,c}$ and at most q_{H} queries to the random oracle H, there exists an $\mathsf{OW\text{-}PVCA}$ adversary B against PKE' that issues at most q_{H} queries to its PCO oracle, and q_D queries to its VALID oracle such that

$$\text{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}^{\perp}_{m,c}}(\mathsf{A}) \leq \text{Adv}^{\mathsf{OW\text{-}PVCA}}_{\mathsf{PKE}'}(\mathsf{B}) + \delta'_{\text{ac}} \ ,$$

and the running time of B is about that of A.

The main idea of the proof is to simulate the decapsulation oracle without the secret key. This can be done by answering decryption queries with a random key and then later patch the random oracle, using the plaintext checking oracle PCO provided by the $\mathsf{OW\text{-}PVCA}$ game. Additionally, the ciphertext validity oracle VALID is required to reject decapsulation queries with inconsistent ciphertexts.

Proof. Let A be an adversary against the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM}^{\perp}_{m,c}$, issuing at most q_D queries to $\textsc{Dec}^{\perp}_{m,c}$ and at most q_{H} queries to H. Consider the sequence of games given in Figure 2.7.

Game G_0. Since game G_0 is the original $\mathsf{IND\text{-}CCA}$ game,

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

Game G_1. In game G_1, the oracles H and $\mathrm{Dec}^{\perp}_{m,c}$ are modified such that they make no use of the secret key any longer, except by testing if $\mathsf{Dec}'(sk, c) = m$ for given (m, c) in line 18, and if $\mathsf{Dec}'(sk, c) \in \mathcal{M}$ for given c in line 29. We will use two lists, hash list $\mathfrak{L}_H$ and decapsulation list $\mathfrak{L}_D$, for book-keeping. Existence of an entry $(m, c, K) \in \mathfrak{L}_H$ indicates that H was queried on (m, c) and returned $\mathsf{H}(m, c) := K$, existence of an entry $(c, K) \in \mathfrak{L}_D$ indicates that either $\mathrm{Dec}^{\perp}_{m,c}$ was queried on c (see line 32) and returned K, or that H was queried on (m', c) for $m' := \mathsf{Dec}'(sk, c)$ and returned K (see line 24). In order to show that the view of A is identical in games G_0 and G_1, we have to consider the following cases for a fixed ciphertext c and $m' := \mathsf{Dec}'(sk, c)$:

- Case 1: $m' \notin \mathcal{M}$. Since $\mathsf{Dec}'(sk, c) \notin \mathcal{M}$ is equivalent to $m' = \perp$, $\mathrm{Dec}^{\perp}_{m,c}(c)$ returns $\perp$ in both games. Note that $\mathsf{Dec}'(sk, c) \notin \mathcal{M}$ is also equivalent to $\mathrm{VALID}(c) = 0$.

- Case 2: $m' \in \mathcal{M}$. We will now show that H in game G_1 is "patched", meaning that it is ensured that $\mathrm{Dec}^{\perp}_{m,c}(c) = \mathsf{H}(m', c)$ for all ciphertexts c such that $m' := \mathsf{Dec}'(sk, c) \in \mathcal{M}$. We distinguish two sub-cases: A might either first query H on (m', c), and then query $\mathrm{Dec}^{\perp}_{m,c}$ on c, or the other way round.

    - If H is queried on (m', c) first, it is recognised that $\mathsf{Dec}'(sk, c) = m$ in line 18. Since $\mathrm{Dec}^{\perp}_{m,c}$ was not yet queried on c, no entry of the form (c, K) can already exist in $\mathfrak{L}_D$. Therefore, besides adding $(m, c, K \leftarrow_{\$} \mathcal{K})$ to $\mathfrak{L}_H$, H also adds (c, K) to $\mathfrak{L}_D$ in line 24, thereby defining $\mathrm{Dec}^{\perp}_{m,c}(c) := K = \mathsf{H}(m', c)$.
    - If $\mathrm{Dec}^{\perp}_{m,c}$ is queried on c first, no entry of the form (c, K) exists in $\mathfrak{L}_D$ yet. Therefore, $\mathrm{Dec}^{\perp}_{m,c}$ adds $(c, K \leftarrow_{\$} \mathcal{K})$ to $\mathfrak{L}_D$, thereby defining $\mathrm{Dec}^{\perp}_{m,c}(c) := K$. When queried on (m', c) afterwards, H recognises that $\mathsf{Dec}'(sk, c) = m'$ in line 18 and that an entry of the form (c, K) already exists in $\mathfrak{L}_D$ in line 21. By adding (m, c, K) to $\mathfrak{L}_H$ and returning K, H defines $\mathsf{H}(m', c) := K = \mathrm{Dec}^{\perp}_{m,c}(c)$.

    Note that $\mathsf{Dec}'(sk, c) = m$ is equivalent to $\mathrm{PCO}(m, c) = 1$.

We have shown that A's view is identical in both games and

$$\Pr[G_0^{\mathsf{A}} \Rightarrow 1] = \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \ .$$

GAME G_2. In game G_2, we abort immediately on the event that A queries H on $(\mathsf{Dec}'(sk, c^*), c^*)$. We denote this event by CHAL. Furthermore, we raise flag ERROR in line 09 and abort if c^* exhibits decryption failure, i.e., if $\mathsf{Dec}'(sk, c^*) \neq m^*$. Unless ERROR happens, $\mathsf{H}(m^*, c^*)$ will not be given to A in game G_2; neither through a hash nor a decryption query, meaning bit b is independent from A's view and hence,

$$\Pr[G_2^{\mathsf{A}} \Rightarrow 1] = \frac{1}{2} \ .$$

Due to the difference lemma,

$$\begin{aligned} |\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| &\leq \Pr[\mathrm{ERROR} \vee \mathrm{CHAL}] \\ &= \Pr[\mathrm{ERROR}] + \Pr[\mathrm{CHAL} \wedge \neg\mathrm{ERROR}] \ . \end{aligned}$$

Since m^* was drawn uniformly at random,

$$\Pr[\mathrm{ERROR}] \leq \delta'_{\mathrm{ac}} \ .$$

It remains to bound $\Pr[\mathrm{CHAL} \wedge \neg\mathrm{ERROR}]$. To this end, we construct an adversary B against the $\mathsf{OW\text{-}PVCA}$ security of PKE' in Figure 2.8, simulating G_2 for A.

Note that the simulation is perfect. The event that CHAL occurred (but ERROR did not) implies that A queried $\mathsf{H}(m^*, c^*)$, hence $(m^*, c^*, K') \in \mathfrak{L}_H$ for some K', and B returns $m' = m^*$.

$$\Pr[\mathrm{CHAL} \wedge \neg\mathrm{ERROR}] = \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW\text{-}PVCA}}(\mathsf{B}) \ .$$

Collecting the probabilities yields the desired bound.

□

Transformation $\mathsf{U}_{m,c}^{\not\perp}$: from $\mathsf{OW\text{-}PCA}$ to $\mathsf{IND\text{-}CCA}$

Transformation $\mathsf{U}_{m,c}^{\not\perp}$ is a variant of $\mathsf{U}_{m,c}^{\perp}$ with "implicit rejection" of inconsistent ciphertexts. The $^{\not\perp}$ in $\mathsf{U}_{m,c}^{\not\perp}$ means that, instead of returning $\perp$, decapsulation returns a pseudorandom key. It transforms an $\mathsf{OW\text{-}PCA}$ secure public-key encryption scheme into an $\mathsf{IND\text{-}CCA}$ secure key encapsulation mechanism.

THE CONSTRUCTION. To a public-key encryption scheme $\mathsf{PKE}' = (\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}')$ with message space $\mathcal{M}$, and a random oracle $\mathsf{H} : \{0,1\}^* \to \mathcal{K}$, we associate $\mathsf{KEM}_{m,c}^{\not\perp} = \mathsf{U}_{m,c}^{\not\perp}[\mathsf{PKE}', \mathsf{H}] = (\mathsf{KG}^{\not\perp}, \mathsf{Encaps}, \mathsf{Decaps}_{m,c}^{\not\perp})$. The algorithms of $\mathsf{KEM}_{m,c}^{\not\perp}$ are defined in Figure 2.9. Note that Encaps is the same as in $\mathsf{KEM}_{m,c}^{\perp}$ (Figure 2.6), and that $\mathsf{U}_{m,c}^{\perp}$

```
$\mathsf{B}^{\mathrm{PCO}}(pk, c^*)$
$K^* \leftarrow_\$ \mathcal{K}$
$b' \leftarrow \mathsf{A}^{\mathrm{DEC}^{\perp}_{m,c}, \mathsf{H}}(pk, c^*, K^*)$
if $\exists (m', c', K') \in \mathfrak{L}_H$
      s. th. $\mathrm{PCO}(m', c^*) = 1$
    return $m'$
else abort

$\mathsf{H}(m, c)$
if $\exists K$ such that $(m, c, K) \in \mathfrak{L}_H$
    return $K$
$K \leftarrow_\$ \mathcal{K}$
if $\mathrm{PCO}(m, c) = 1$
    if $\exists K'$ such that $(c, K') \in \mathfrak{L}_D$
        $K := K'$
    else
        $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$
$\mathfrak{L}_H := \mathfrak{L}_H \cup \{(m, c, K)\}$
return $K$
```

Fig. 2.8: Adversary B against $\mathsf{OW\text{-}PVCA}$ for the proof of Theorem 2.1.4, where $\mathrm{DEC}^{\perp}_{m,c}$ is defined as in Game G_2 of Figure 2.7. (Line 29 can be executed via a call to the VALID oracle.)

and $\mathsf{U}^{\not\perp}_{m,c}$ essentially differ in decapsulation: $\mathsf{Decaps}^{\perp}_{m,c}$ from $\mathsf{U}^{\perp}_{m,c}$ rejects if c decrypts to $\perp$, whereas $\mathsf{Decaps}^{\not\perp}_{m,c}$ from $\mathsf{U}^{\not\perp}_{m,c}$ returns a pseudorandom key $K := \mathsf{H}(s, c)$.

Again, it is easy to verify that if PKE' is δ'_{ac}-average-case correct, then $\mathsf{KEM}^{\not\perp}_{m,c}$ is δ'_{ac}-correct.

```
$\mathsf{KG}^{\not\perp}$
$(pk', sk') \leftarrow \mathsf{KG}'$
$s \leftarrow_\$ \mathcal{M}$
$sk := (sk', s)$
return $(pk', sk)$

$\mathsf{Encaps}(pk)$
$m \leftarrow_\$ \mathcal{M}$
$c \leftarrow \mathsf{Enc}'(pk, m)$
$K := \mathsf{H}(m, c)$
return $(K, c)$

$\mathsf{Decaps}^{\not\perp}_{m,c}(sk, c)$
Parse $sk = (sk', s)$
$m' := \mathsf{Dec}'(sk', c)$
if $m' = \perp$
    return $K := \mathsf{H}(s, c)$
else return $K := \mathsf{H}(m', c)$
```

Fig. 2.9: $\mathsf{IND\text{-}CCA}$ secure key encapsulation mechanism $\mathsf{KEM}^{\not\perp}_{m,c} = \mathsf{U}^{\not\perp}_{m,c}[\mathsf{PKE}', \mathsf{H}]$.

Security. The following theorem establishes that $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM}^{\not\perp}_{m,c}$ tightly reduces to the $\mathsf{OW\text{-}PCA}$ security of PKE', in the random oracle model.

Theorem 2.1.5 (PKE' $\mathsf{OW\text{-}PCA}$ $\overset{\mathrm{ROM}}{\Rightarrow}$ $\mathsf{KEM}^{\not\perp}_{m,c}$ $\mathsf{IND\text{-}CCA}$)**.** Assume PKE' to be δ'_{ac}-average-case correct. For any $\mathsf{IND\text{-}CCA}$ adversary A against $\mathsf{KEM}^{\not\perp}_{m,c}$, issuing at most q_D queries to the decapsulation oracle $\mathrm{DEC}^{\not\perp}_{m,c}$ and at most q_{H} queries to the random oracle H, there exists an $\mathsf{OW\text{-}PCA}$ adversary B against PKE' that makes at most q_{H} queries to the PCO oracle such that

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}^{\not\perp}_{m,c}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}) + \frac{q_{\mathsf{H}}}{|\mathcal{M}|} + \delta'_{\mathrm{ac}} \enspace ,$$

and the running time of B is about that of A.

The proof is very similar to the one of Theorem 2.1.4. The difference is the handling

```
GAMES $G_0$ - $G_3$
01 $(pk', sk') \leftarrow \mathsf{KG}'$
02 $s \leftarrow_\$ \mathcal{M}$
03 $sk := (sk', s)$
04 $m^* \leftarrow_\$ \mathcal{M}$
05 $c^* \leftarrow \mathsf{Enc}'(pk', m^*)$
06 $K_0^* := \mathsf{H}(m^*, c^*)$
07 $K_1^* \leftarrow_\$ \{0,1\}^n$
08 $b \leftarrow_\$ \{0,1\}$
09 $b' \leftarrow \mathsf{A}^{\textsc{Dec}^{\not\bot}_{m,c}, \mathsf{H}}(pk', c^*, K_b^*)$
10 if $\mathsf{Dec}'(sk, c^*) \neq m^*$                //$G_3$
11    ERROR := true                                 //$G_3$
12    abort                                         //$G_3$
13 return $[\![b' = b]\!]$

$\textsc{Dec}^{\not\bot}_{m,c}(c \neq c^*)$                  //$G_0$-$G_1$
14 $m' := \mathsf{Dec}'(sk', c)$
15 if $m' = \bot$
16    return $K := \mathsf{H}(s, c)$                //$G_0$
17    return $K := \mathsf{H}'(c)$                  //$G_1$
18 if $m' = s$ return $K := \mathsf{H}'(c)$         //$G_1$
19 return $K := \mathsf{H}(m', c)$

$\mathsf{H}(m, c)$
20 if $\exists K$ s. th. $(m, c, K) \in \mathfrak{L}_H$ return $K$
21 $K \leftarrow_\$ \mathcal{K}$
22 if $m = s$                                       //$G_1$-$G_3$
23    QUERY := true; abort                          //$G_1$-$G_3$
24 if $\mathsf{Dec}'(sk', c) = m$                   //$G_2$-$G_3$
25    if $c = c^*$                                  //$G_3$
26       CHAL := true; abort                        //$G_3$
27    if $\exists K'$ such that $(c, K') \in \mathfrak{L}_D$   //$G_2$-$G_3$
28       $K := K'$                                  //$G_2$-$G_3$
29    else                                          //$G_2$-$G_3$
30       $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$      //$G_2$-$G_3$
31 $\mathfrak{L}_H := \mathfrak{L}_H \cup \{(m, c, K)\}$
32 return $K$

$\textsc{Dec}^{\not\bot}_{m,c}(c \neq c^*)$                  //$G_2$-$G_3$
33 if $\exists K$ s. th. $(c, K) \in \mathfrak{L}_D$
34    return $K$
35 else
36    $K \leftarrow_\$ \mathcal{K}$
37    $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$
38    return $K$
```

Fig. 2.10: Games G_0 - G_3 for the proof of Theorem 2.1.5 . Oracle H' (lines 17 and 18) is an independent internal random oracle that cannot be accessed by A.

of decapsulation queries for inconsistent ciphertexts. Since the OW-PCA experiment does not provide a VALID oracle, the handling of invalid ciphertexts has to be integrated into how we patch the random oracle.

Proof. Let A be an adversary against the IND-CCA security of $\mathsf{KEM}^{\not\bot}_{m,c}$, issuing at most q_D queries to $\textsc{Dec}^{\not\bot}_{m,c}$ and at most q_{H} queries to H. Consider the sequence of games given in Figure 2.10.

Game G_0. Since game G_0 is the original IND-CCA game,

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}^{\not\bot}_{m,c}}(\mathsf{A}) = \left| \Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \frac{1}{2} \right| \quad .$$

Game G_1. In game G_1, we make two changes: First, we raise flag QUERY and abort if $\mathsf{H}(s, \cdot)$ is queried (lines 22 and 23). Second, we make the pseudorandom keys that are returned by $\textsc{Dec}^{\not\bot}_{m,c}$ perfectly random. That is, in $\textsc{Dec}^{\not\bot}_{m,c}(c)$, we replace $K = \mathsf{H}(s, c)$ by $K = \mathsf{H}'(c)$ if $m' := \mathsf{Dec}'(sk', c) = \bot$ (line 17) or if $m' := \mathsf{Dec}'(sk', c) = s$ (line 18), where H' is an independent internal random oracles that cannot be accessed by A. Unless QUERY occurs, A's view is identical in both games: Let c be any query to

$\textsc{Dec}^{\not\perp}_{m,c}$ such that $\mathsf{Dec}'(sk', c) \in \{\perp, s\}$. Since $\textsc{Dec}^{\not\perp}_{m,c}(c)$ still returns a random value, and since $\mathsf{Dec}'(sk', c)$ is unique, the change remains unnoticed by A unless A queries H on (s, c).

Since A's view is independent of (the uniform secret) s unless G_1 aborts due to occurrence of QUERY,

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq \frac{q_{\mathsf{H}}}{|\mathcal{M}|} \ .$$

Game G_2. In game G_2, the oracles H and $\textsc{Dec}^{\not\perp}_{m,c}$ are modified such that $\textsc{Dec}^{\not\perp}_{m,c}$ does not make use of the secret key any longer, except by testing if $\mathsf{Dec}'(sk', c) = m$ for given (m, c) in line 24. We will use two lists, hash list $\mathfrak{L}_H$ and decapsulation list $\mathfrak{L}_D$, for book-keeping. Existence of an entry $(m, c, K) \in \mathfrak{L}_H$ indicates that H was queried on (m, c) and returned $\mathsf{H}(m, c) := K$. Existence of an entry $(c, K) \in \mathfrak{L}_D$ indicates that either H was queried on $(m := \mathsf{Dec}'(sk', c), c)$, or $\textsc{Dec}^{\not\perp}_{m,c}$ was queried on c, and either way, it holds that $\textsc{Dec}^{\not\perp}_{m,c}(c) = K$.

In order to show that the view of A is identical in games G_1 and G_2, consider the following cases for a fixed ciphertext c and $m' := \mathsf{Dec}'(sk', c)$.

- Case 1: $m' \in \{\perp, s\}$. Since H cannot be queried on (m', c) (i.e., $\mathsf{H}(\perp, \cdot)$ is not allowed, and $\mathsf{H}(s, c)$ results in abort), the simulation of H can never add a tuple of the form (c, K) to $\mathfrak{L}_D$. Hence, querying $\textsc{Dec}^{\not\perp}_{m,c}(c)$ in game G_2 will return a uniformly random key, as does Game G_1.

- Case 2: $m' \notin \{\perp, s\}$. We will now show that H in game G_2 is "patched", meaning that it is ensured that $\textsc{Dec}^{\not\perp}_{m,c}(c) = \mathsf{H}(m', c)$ for all valid ciphertexts c with $m' = \mathsf{Dec}'(sk', c) \neq s$. We distinguish two sub-cases: A might either first query H on (m', c), and then query $\textsc{Dec}^{\not\perp}_{m,c}$ on c, or the other way round.
  - If H is queried on (m', c) first, it is recognised that $\mathsf{Dec}'(sk', c) = m'$ in line 24. Since $\textsc{Dec}^{\not\perp}_{m,c}$ was yet not queried on c, no entry of the form (c, K) already exists in $\mathfrak{L}_D$. Therefore, besides adding $(m', c, K \leftarrow_{\$} \mathcal{K})$ to $\mathfrak{L}_H$, H also adds (c, K) to $\mathfrak{L}_D$ in line 30, thereby defining $\textsc{Dec}^{\not\perp}_{m,c}(c) := K = \mathsf{H}(m', c)$.
  - If $\textsc{Dec}^{\not\perp}_{m,c}$ is queried on c first, no entry of the form (c, K) exists in $\mathfrak{L}_D$ yet. Therefore, $\textsc{Dec}^{\not\perp}_{m,c}$ adds $(c, K \leftarrow_{\$} \mathcal{K})$ to $\mathfrak{L}_D$ thereby defining $\textsc{Dec}^{\not\perp}_{m,c}(c) := K$. When queried on (m', c) afterwards, H recognises that $\mathsf{Dec}'(sk', c) = m'$ in line 24 and that an entry of the form (c, K) already exists in $\mathfrak{L}_D$ in line 27. By adding (m', c, K) to $\mathfrak{L}_H$ and returning K, H defines $\mathsf{H}(m', c) := K = \textsc{Dec}^{\not\perp}_{m,c}(c)$.

We have shown that A's view is identical in both games and

$$\Pr[G_1^{\mathsf{A}} \Rightarrow 1] = \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \enspace .$$

Game G_3. From game G_3 on, we proceed identically to the proof of Theorem 2.1.4: In game G_3, we abort immediately (and raise flag CHAL) on the event that A queries H on $(\mathsf{Dec}'(sk, c^*), c^*)$. Furthermore, we raise flag ERROR in line 11 and abort if c^* exhibits decryption failure, i.e., if $\mathsf{Dec}'(sk, c^*) \neq m^*$.

Unless ERROR happens, $\mathsf{H}(m^*, c^*)$ will not be given to A in game G_3; neither through a hash nor a decryption query, meaning bit b is independent from A's view and hence,

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] = \frac{1}{2} \enspace .$$

Due to the difference lemma,

$$\begin{aligned} |\Pr[G_2^{\mathsf{A}} \Rightarrow 1] - \Pr[G_3^{\mathsf{A}} \Rightarrow 1]| &\leq \Pr[\text{ERROR} \vee \text{CHAL}] \\ &= \Pr[\text{ERROR}] + \Pr[\text{CHAL} \wedge \neg\text{ERROR}] \\ &\leq \delta'_{\text{ac}} + \Pr[\text{CHAL} \wedge \neg\text{ERROR}] \enspace . \end{aligned}$$

It remains to bound $\Pr[\text{CHAL} \wedge \neg\text{ERROR}]$. To this end, we construct an adversary B against the $\mathsf{OW\text{-}PCA}$ security of PKE' in Figure 2.11, simulating G_3 for A. Note that the simulation is perfect. The event that CHAL occurred (but ERROR did not) implies that A queried $\mathsf{H}(m^*, c^*)$, hence $(m^*, c^*, K') \in \mathfrak{L}_H$ for some K', and B returns $m' = m^*$.

$$\Pr[\text{CHAL}] = \text{Adv}_{\mathsf{PKE}}^{\mathsf{OW\text{-}PCA}}(\mathsf{B}) \enspace .$$

Collecting the probabilities yields the desired bound.

□

2.1.3 Transformation $\mathsf{U}_m^{\perp}$ ($\mathsf{U}_m^{\not\perp}$): From deterministic OW-VCA *(*OW*) to* IND-CCA

Transformation $\mathsf{U}_m^{\perp}$ is a variant of $\mathsf{U}_{m,c}^{\perp}$ that derives the KEM key as $K = \mathsf{H}(m)$, instead of $K = \mathsf{H}(m, c)$. It transforms an $\mathsf{OW\text{-}VCA}$ secure public-key encryption scheme with rigid deterministic encryption (e.g., one obtained via T from Section 2.1.1) into an

```
$\mathsf{B}^{\mathrm{PCO}}(pk, c^*)$
$K^* \leftarrow_\$ \mathcal{K}$
$s \leftarrow_\$ \mathcal{M}$
$b' \leftarrow \mathsf{A}^{\mathrm{DEC}^{\not\perp}_{m,c},\mathsf{H}}(pk, c^*, K^*)$
if $\exists (m', c', K') \in \mathfrak{L}_H$
  s. th. $\mathrm{PCO}(m', c^*) = 1$
    return $m'$
else
    abort
```

```
$\mathsf{H}(m, c)$
if $\exists K$ s. th. $(m, c, K) \in \mathfrak{L}_H$ return $K$
$K \leftarrow_\$ \mathcal{K}$
if $m = s$ abort
if $\mathrm{PCO}(m, c) = 1$
    if $\exists K'$ s. th. $(c, K') \in \mathfrak{L}_D$
        $K := K'$
    else
        $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$
$\mathfrak{L}_H := \mathfrak{L}_H \cup \{(m, c, K)\}$
return $K$
```

Fig. 2.11: Adversary B against $\mathsf{OW\text{-}PCA}$ for the proof of Theorem 2.1.5. Oracle $\mathrm{DEC}^{\not\perp}_{m,c}$ is defined as in game G_3 of Figure 2.10.

IND-CCA secure key encapsulation mechanism. We also consider an implicit rejection variant $\mathsf{U}^{\not\perp}_m$ that only requires OW security of the underlying encryption scheme PKE'.

The Constructions. To a public-key encryption scheme $\mathsf{PKE}' = (\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}')$ with message space $\mathcal{M}$, and a random oracle $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n$, we associate

$$\mathsf{KEM}^{\not\perp}_m := \mathsf{U}^{\not\perp}_m[\mathsf{PKE}', \mathsf{H}] := (\mathsf{KG}^{\not\perp}, \mathsf{Encaps}_m, \mathsf{Decaps}^{\not\perp}_m)$$

and

$$\mathsf{KEM}^{\perp}_m := \mathsf{U}^{\perp}_m[\mathsf{PKE}', \mathsf{H}] := (\mathsf{KG}', \mathsf{Encaps}_m, \mathsf{Decaps}^{\perp}_m) \ .$$

Algorithm $\mathsf{KG}^{\not\perp}$ is given in Figure 2.9, and the remaining algorithms of $\mathsf{KEM}^{\not\perp}_m$ and $\mathsf{KEM}^{\perp}_m$ are defined in Figure 2.12.

Again, it is easy to verify that if PKE' is δ'_{ac}-average-case correct, then both $\mathsf{KEM}^{\not\perp}_m$ and $\mathsf{KEM}^{\perp}_m$ are δ'_{ac}-correct.

```
$\mathsf{Encaps}_m(pk)$
$m \leftarrow_\$ \mathcal{M}$
$c := \mathsf{Enc}'(pk, m)$
$K := \mathsf{H}(m)$
return $(K, c)$
```

```
$\mathsf{Decaps}^{\not\perp}_m(sk, c)$
Parse $sk = (sk', s)$
$m' := \mathsf{Dec}'(sk', c)$
if $m' = \perp$
    return $K := \mathsf{H}(s, c)$
else return $K := \mathsf{H}(m')$
```

```
$\mathsf{Decaps}^{\perp}_m(sk, c)$
$m' := \mathsf{Dec}'(sk, c)$
if $m' = \perp$ return $\perp$
else return $K := \mathsf{H}(m')$
```

Fig. 2.12: IND-CCA-secure key encapsulation mechanisms $\mathsf{KEM}^{\not\perp}_m = \mathsf{U}^{\not\perp}_m[\mathsf{PKE}', \mathsf{H}]$ and $\mathsf{KEM}^{\perp}_m = \mathsf{U}^{\perp}_m[\mathsf{PKE}', \mathsf{H}]$.

Transformation $\mathsf{U}_m^{\perp}$: from $\mathsf{OW\text{-}VCA}$ to $\mathsf{IND\text{-}CCA}$

Security of $\mathsf{KEM}_m^{\perp}$. The following theorem establishes that $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM}_m^{\perp}$ tightly reduces to the $\mathsf{OW\text{-}VCA}$ security of PKE', in the random oracle model.

Theorem 2.1.6 (PKE' det. + rigid, $\mathsf{OW\text{-}VCA} \overset{\text{ROM}}{\Rightarrow} \mathsf{KEM}_m^{\perp}$ $\mathsf{IND\text{-}CCA}$)**.** Assume PKE' to be deterministic and rigid, and furthermore, let PKE' be δ'_{wc}-worst-case correct. Let G denote the random oracle that PKE' uses (if any). Let $q_{\mathsf{Enc}',\mathsf{G}}$ denote an upper bound on the number of G-queries that Enc' makes upon a single invocation (if any).

For any $\mathsf{IND\text{-}CCA}$ adversary A against $\mathsf{KEM}_m^{\perp}$, issuing at most q_D queries to the decapsulation oracle $\textsc{Dec}_m^{\perp}$ and at most q_{H}, resp. q_{G} queries to its random oracles H and G, there exists an $\mathsf{OW\text{-}VCA}$ adversary B against PKE' issuing at most q_D queries to the VALID oracle, and $q_{\mathsf{G}} + q_{\mathsf{H}} \cdot q_{\mathsf{Enc}',\mathsf{G}}$ many queries to oracle G, and a correctness adversary C such that

$$\mathrm{Adv}_{\mathsf{KEM}_m^{\perp}}^{\mathsf{IND\text{-}CCA}}(\mathsf{A}) \leq \mathrm{Adv}_{\mathsf{PKE}'}^{\mathsf{OW\text{-}VCA}}(\mathsf{B}) + \mathrm{Adv}_{\mathsf{PKE}'}^{\mathsf{COR\text{-}RO}_{q_{\mathsf{H}}}}(\mathsf{C}) \ ,$$

and the running time of B is about that of A.

In principle, the proof is similar to the one of Theorem 2.1.4. The main difference is that we now exploit that Enc' is deterministic and rigid to (implicitly) simulate our own PCO oracle via re-encryption during the proof.

Proof. To show security of $\mathsf{KEM}_m^{\perp}$, let A be an adversary against the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM}_m^{\perp}$, issuing at most q_D queries to $\textsc{Dec}_m^{\perp}$ and at most q_{H} queries to H. Consider the games given in Figure 2.13.

Game G_0. Since game G_0 is the original $\mathsf{IND\text{-}CCA}$ game,

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

Game G_1. In game G_1, the oracles H and $\textsc{Dec}_m^{\perp}$ are changed such that they make no use of the secret key any longer, except for testing if $\mathsf{Dec}'(sk, c) \in \mathcal{M}$ for given c in line 27. Similar to the proofs for $\mathsf{U}_{m,c}^{\not\perp}$ and $\mathsf{U}_{m,c}^{\perp}$, we will use hash list $\mathfrak{L}_H$ and decapsulation list $\mathfrak{L}_D$ for book-keeping. Hash list $\mathfrak{L}_H$ contains all entries (m, K) where H was queried on m and returned $\mathsf{H}(m) := K$. Decapsulation list $\mathfrak{L}_D$ contains all entries (c, K) where either $\textsc{Dec}_m^{\perp}$ was queried on c, or H was queried on some message m such that $c = \mathsf{Enc}'(pk, m)$.

```
GAMES $G_0$ - $G_2$
01 $(pk, sk) \leftarrow \mathsf{KG}'$
02 $m^* \leftarrow_\$ \mathcal{M}$
03 $K_0^* := \mathsf{H}(m^*)$
04 $K_1^* \leftarrow_\$ \{0,1\}^n$
05 $c^* := \mathsf{Enc}'(pk, m^*)$
06 $b \leftarrow_\$ \{0,1\}$
07 $b' \leftarrow \mathsf{A}^{\textsc{Dec}_m^\perp, \mathsf{H}}(pk, c^*, K_b^*)$
08 return $[\![b' = b]\!]$

$\textsc{Dec}_m^\perp(c \neq c^*)$                        // $G_0$
09 $m' := \mathsf{Dec}'(sk, c)$
10 if $m' = \perp$ return $\perp$
11 return $K := \mathsf{H}(m')$

$\mathsf{H}(m)$
12 if $\exists K$ such that $(m, K) \in \mathfrak{L}_H$
13     return $K$
14 if $m = m^*$ and $c^*$ defined                     // $G_2$
15     CHAL := true                                   // $G_2$
16     abort                                          // $G_2$
17 $c' := \mathsf{Enc}'(pk, m)$                       // $G_1$-$G_2$
18 $K \leftarrow_\$ \mathcal{K}$
19 if $\exists K'$ such that $(c', K') \in \mathfrak{L}_D$   // $G_1$-$G_2$
20     $K := K'$                                      // $G_1$-$G_2$
21 else                                               // $G_1$-$G_2$
22     $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c', K)\}$     // $G_1$-$G_2$
23 $\mathfrak{L}_H := \mathfrak{L}_H \cup \{(m, K)\}$
24 return $K$

$\textsc{Dec}_m^\perp(c \neq c^*)$                        // $G_1$-$G_2$
25 if $\exists K$ s. th. $(c, K) \in \mathfrak{L}_D$
26     return $K$
27 if $\mathsf{Dec}'(sk, c) \notin \mathcal{M}$
28     return $\perp$
29 $K \leftarrow_\$ \mathcal{K}$
30 $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$
31 return $K$
```

Fig. 2.13: Games G_0 - G_2 for the proof of Theorem 2.1.6

We note that if the scheme were not assumed to be rigid, then there could exist two distinct ciphertexts $c_1 \neq c_2$ that decrypt to the same message m. In game G_0, $\textsc{Dec}_m^\perp$ would answer both queries with the same value, but this would not necessarily be the case in game G_1. Since we assume the scheme to be rigid, however, we have that if $\mathsf{Dec}'(sk, c_1) = \mathsf{Dec}'(sk, c_2) \neq \perp$, then $c_1 = \mathsf{Enc}'(pk, \mathsf{Dec}'(sk, c_1)) = \mathsf{Enc}'(pk, \mathsf{Dec}'(sk, c_2)) = c_2$.

Let BAD denote the event that $\mathfrak{L}_H$ contains an entry (m, K) such that m exhibits a correctness error, i.e., such that $\mathsf{Dec}'(sk, \mathsf{Enc}'(pk, m)) \neq m$. We will show that the view of A is identical in games G_0 and G_1 unless a query to H occurs on a plaintext that induces a correctness error, i.e., we show that the view only differs if BAD happens.

First, we observe that line 17 will only let H coincide on two distinct messages $m_1 \neq m_2$ if they encrypt to the same ciphertext, meaning that one of the two messages must trigger BAD.

To further analyse game G_1, let c be a query to $\textsc{Dec}_m^\perp$, and let $m' := \mathsf{Dec}'(sk, c)$. We want to show that unless BAD happens, consistency is maintained in game G_1. Here, consistency means that if $m' \neq \perp$, we also have that $\textsc{Dec}_m^\perp(c) = \mathsf{H}(m')$ in game G_1.

Before the queries to $\textsc{Dec}_m^\perp$ on c and to H on m', no entry of the form (c, K)

could already exist in $\mathfrak{L}_D$ unless BAD occurs: Since neither $\textsc{Dec}_m^\perp$ was yet queried on c, nor was H queried yet on m', existence of an entry (c, K) in $\mathfrak{L}_D$ implies that H was already queried on some message $m \neq m'$ such that $\mathsf{Enc}'(pk, m) = c$. Hence, $\mathsf{Dec}'(sk, \mathsf{Enc}'(pk, m)) = \mathsf{Dec}'(sk, c) = m' \neq m$, meaning that m induces a correctness error and BAD happened.

To show $\textsc{Dec}_m^\perp(c) = \mathsf{H}(m')$, we distinguish two sub-cases: A might either first query H on m', then $\textsc{Dec}_m^\perp$ on c, or the other way round.

- If H is queried on m' first, no entry of the form (c, K) already exists in $\mathfrak{L}_D$. Hence, besides adding $(m', K \leftarrow_\$ \mathcal{K})$ to $\mathfrak{L}_H$, H also computes $c' := \mathsf{Enc}'(pk, m')$. Since PKE' is assumed to be rigid and $m' \neq \perp$, we have that $c' = \mathsf{Enc}'(pk, \mathsf{Dec}'(sk, c)) = c$. By adding (c, K) to $\mathfrak{L}_D$ in line 22, H defines $\textsc{Dec}_m^\perp(c) := K = \mathsf{H}(m')$.

- If $\textsc{Dec}_m^\perp$ is queried on c first, it adds $(c, K \leftarrow_\$ \mathcal{K})$ to $\mathfrak{L}_D$, thereby defining $\textsc{Dec}_m^\perp(c) := K$. When queried on m' afterwards, H computes $c' := \mathsf{Enc}'(pk, m') = c$, and recognises that an entry of the form (c, K) already exists in $\mathfrak{L}_D$ in line 19. By adding (m', K) to $\mathfrak{L}_H$ and returning K, H again defines $\mathsf{H}(m') := K = \textsc{Dec}_m^\perp(c)$.

We have shown that A's view is identical in both games unless a correctness error (in the form of BAD) occurs and

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq \Pr[\text{BAD}] \ .$$

We can bound Pr[BAD] with a straightforward reduction to the game-based correctness of PKE'. In this reduction, adversary C simulates game G_0, and adds to its output list each query to H that is issued by A. In total, the list will hence consist of q_{H} many entries. Hence,

$$\Pr[\text{BAD}] \leq \text{Adv}_{\mathsf{PKE}'}^{\mathsf{COR\text{-}RO}_{q_\mathsf{H}}}(\mathsf{C}) \ .$$

<span style="font-variant:small-caps">Game</span> G_2. In game G_2, we abort immediately on the event that A queries H on m^*. We denote this event as CHAL. Due to the difference lemma,

$$|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \leq \Pr[\text{CHAL}] \ .$$

In game G_2, $\mathsf{H}(m^*)$ will not be given to A; neither through a hash nor a decryption query, meaning bit b is independent from A's view and hence,

$$\Pr[G_2^{\mathsf{A}} \Rightarrow 1] = \frac{1}{2} \ .$$

```
$\mathsf{B}^{\mathrm{VALID}}(pk, c^*)$
$K^* \leftarrow_\$ \mathcal{K}$
$b' \leftarrow \mathsf{A}^{\mathrm{DEC}^{\perp}_{m},\mathsf{H}}(pk, c^*, K^*)$
if $\exists (m', K') \in \mathfrak{L}_H$
    s. th. $\mathsf{Enc}'(pk, m') = c^*$
    return $m'$
else abort

$\mathrm{DEC}^{\perp}_{m}(c \neq c^*)$
if $\exists K$ s. th. $(c, K) \in \mathfrak{L}_D$
    return $K$
if $\mathrm{VALID}(c) = 0$
    return $\perp$
$K \leftarrow_\$ \mathcal{K}$
$\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$
return $K$
```

Fig. 2.14: Adversary B against $\mathsf{OW\text{-}VCA}$ for the proof of Theorem 2.1.6, where H is defined as in Game G_1 of Figure 2.13.

It remains to bound Pr[CHAL]. To this end, we construct an adversary B against the $\mathsf{OW\text{-}VCA}$ security of PKE' simulating G_2 for A as in Figure 2.14.

Note that the simulation is perfect until CHAL occurs. The event that CHAL occurred implies that A queried $\mathsf{H}(m^*)$, and hence, $(m^*, K') \in \mathfrak{L}_H$ for some K'. Since Enc' is deterministic, we have that $\mathsf{Enc}'(pk, m^*) = c^*$, and thus B returns m^*.

$$\Pr[\mathrm{CHAL}] = \mathrm{Adv}^{\mathsf{OW\text{-}VCA}}_{\mathsf{PKE}}(\mathsf{B}) \ .$$

Collecting the probabilities yields the required bound.

□

Transformation $\mathsf{U}^{\not\perp}_{m}$: from OW to $\mathsf{IND\text{-}CCA}$

Security of $\mathsf{KEM}^{\not\perp}_{m}$. The following theorem establishes that $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM}^{\not\perp}_{m}$ tightly reduces to the OW security of PKE', in the random oracle model.

Theorem 2.1.7 (PKE' OW $\overset{\mathrm{ROM}}{\Rightarrow}$ $\mathsf{KEM}^{\not\perp}_{m}$ $\mathsf{IND\text{-}CCA}$)**.** Assume PKE' to be deterministic and rigid, and furthermore, let PKE' be δ'_{wc}-worst-case correct. Let G denote the random oracle that PKE' uses (if any), and let $q_{\mathsf{Enc}',\mathsf{G}}$ denote an upper bound on the number of G-queries that Enc' makes upon a single invocation.

For any $\mathsf{IND\text{-}CCA}$ adversary A against $\mathsf{KEM}^{\not\perp}_{m}$, issuing at most q_D queries to the decapsulation oracle $\mathrm{DEC}^{\not\perp}_{m}$ and at most q_{H}, resp. q_{G} queries to its random oracles H and G, there exists an OW adversary B against PKE', issuing at most $q_{\mathsf{G}} + q_{\mathsf{H}} \cdot q_{\mathsf{Enc}',\mathsf{G}}$ many queries to oracle G, and a correctness adversary C such that

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}^{\not\perp}_{m}}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}'}(\mathsf{B}) + \frac{q_{\mathsf{H}}}{|\mathcal{M}|} + \mathrm{Adv}^{\mathsf{COR\text{-}RO}_{q_{\mathsf{H}}}}_{\mathsf{PKE}'}(\mathsf{C}) \ ,$$

and the running time of B is about that of A.

```
GAMES $G_0$ - $G_3$
01 $(pk', sk') \leftarrow \mathsf{KG}'$
02 $s \leftarrow_\$ \mathcal{M}$
03 $m^* \leftarrow_\$ \mathcal{M}$
04 $c^* := \mathsf{Enc}'(pk', m^*)$
05 $K_0^* := \mathsf{H}(m^*)$
06 $K_1^* \leftarrow_\$ \{0,1\}^n$
07 $b \leftarrow_\$ \{0,1\}$
08 $b' \leftarrow \mathsf{A}^{\mathrm{DEC}^{\not\perp}_m, \mathsf{H}}(pk', c^*, K_b^*)$
09 if $\mathsf{Dec}'(sk, c^*) \neq m^*$               //$G_3$
10    ERROR := true                         //$G_3$
11    abort                                 //$G_3$
12 return $[\![b' = b]\!]$

$\mathrm{DEC}^{\not\perp}_m(c \neq c^*)$                      //$G_0$-$G_1$
13 $m' := \mathsf{Dec}'(sk', c)$
14 if $m' = \perp$
15    return $K := \mathsf{H}(s, c)$               //$G_0$
16    return $K := \mathsf{H}'(c)$                 //$G_1$
17 if $m' = s$ return $K := \mathsf{H}'(c)$        //$G_1$
18 return $K := \mathsf{H}(m')$

$\mathsf{H}(m)$
19 if $\exists K$ s. th. $(m, K) \in \mathfrak{L}_H$ return $K$
20 $K \leftarrow_\$ \mathcal{K}$
21 if $m = s$                                 //$G_1$-$G_3$
22    QUERY := true; abort                  //$G_1$-$G_3$
23 if $m = m^*$ and $c^*$ defined                 //$G_3$
24    CHAL := true                          //$G_3$
25    abort                                 //$G_3$
26 $c' := \mathsf{Enc}'(pk', m)$                     //$G_1$-$G_2$
27 if $\exists K'$ such that $(c', K') \in \mathfrak{L}_D$   //$G_1$-$G_2$
28    $K := K'$                               //$G_1$-$G_2$
29 else                                     //$G_1$-$G_2$
30    $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c', K)\}$         //$G_1$-$G_2$
31 $\mathfrak{L}_H := \mathfrak{L}_H \cup \{(m, K)\}$
32 return $K$

$\mathrm{DEC}^{\not\perp}_m(c \neq c^*)$                      //$G_2$-$G_3$
33 if $\exists K$ s. th. $(c, K) \in \mathfrak{L}_D$
34    return $K$
35 else
36    $K \leftarrow_\$ \mathcal{K}$
37    $\mathfrak{L}_D := \mathfrak{L}_D \cup \{(c, K)\}$
38    return $K$
```

Fig. 2.15: Games G_0 - G_3 for the proof of Theorem 2.1.7 . Oracle H' (lines 17 and 18) is an independent internal random oracle that cannot be accessed by A.

The proof is easily obtained by combining the proofs of Theorem 2.1.5 and Theorem 2.1.6, but we include it for the sake of completeness.

Proof. Let A be an adversary against the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM}^{\not\perp}_m$, issuing at most q_D queries to $\mathrm{DEC}^{\not\perp}_m$ and at most q_{H} queries to H. Consider the sequence of games given in Figure 2.15.

GAME G_0. Since game G_0 is the original $\mathsf{IND\text{-}CCA}$ game,

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}^{\not\perp}_m}(\mathsf{A}) = \left| \Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \frac{1}{2} \right| \ .$$

GAME G_1. In game G_1, we make two changes: First, we raise flag QUERY and abort if $\mathsf{H}(s)$ is queried (lines 21 and 22). Second, we make the pseudorandom keys that are returned by $\mathrm{DEC}^{\not\perp}_m$ perfectly random. That is, in $\mathrm{DEC}^{\not\perp}_m(c)$, we replace $K = \mathsf{H}(s, c)$ by $K = \mathsf{H}'(c)$ if $m' := \mathsf{Dec}'(sk', c) = \perp$ (line 16) or if $m' := \mathsf{Dec}'(sk', c) = s$ (line 17), where H' is an independent internal random oracles that cannot be accessed by A. Unless QUERY occurs, A's view is identical in both games: Let c be any query to $\mathrm{DEC}^{\not\perp}_m$

such that $\mathsf{Dec}'(sk', c) \in \{\perp, s\}$. Since $\textsc{Dec}_m^{\not\perp}(c)$ still returns a random value, and since $\mathsf{Dec}'(sk', c)$ is unique, the change remains unnoticed by A unless A queries H on s.

Since A's view is independent of (the uniform secret) s unless G_1 aborts due to occurrence of QUERY,

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq \frac{q_{\mathsf{H}}}{|\mathcal{M}|} \ .$$

Game G_2. In game G_2, the oracles H and $\textsc{Dec}_m^{\not\perp}$ are modified such that $\textsc{Dec}_m^{\not\perp}$ does not make use of the secret key any longer: Again, we will use two lists, hash list $\mathfrak{L}_H$ and decapsulation list $\mathfrak{L}_D$, for book-keeping. Existence of an entry $(m, K) \in \mathfrak{L}_H$ indicates that H was queried on m and returned $\mathsf{H}(m) := K$. Existence of an entry $(c, K) \in \mathfrak{L}_D$ indicates that either H was queried on some message m such that $c = \mathsf{Enc}'(pk', m)$ or $\textsc{Dec}_m^{\not\perp}$ was queried on c, and either way, it holds that $\textsc{Dec}_m^{\not\perp}(c) = K$.

Let BAD denote the event that $\mathfrak{L}_H$ contains an entry (m, K) such that m exhibits a correctness error, i.e., such that $\mathsf{Dec}'(sk', \mathsf{Enc}'(pk', m)) \neq m$.

Similar to the proof of Theorem 2.1.6, we will now show that the view of A is identical in games G_1 and G_2 unless a query to H occurs on a plaintext that induces a correctness error, i.e., we show that the view only differs if BAD happens.

To do so, we have to examine if $\textsc{Dec}_m^{\not\perp}$ and H handle queries consistently in game G_1: In game G_1, it holds that $\textsc{Dec}_m^{\not\perp}(c) = \mathsf{H}(\mathsf{Dec}'(sk', c))$ for all ciphertexts c such that $\mathsf{Dec}'(sk', c) \notin \{\perp, s\}$, and $\textsc{Dec}_m^{\not\perp}(c) = \mathsf{H}'(c)$ for all ciphertexts c such that $\mathsf{Dec}'(sk', c) \in \{\perp, s\}$.

In order to show that the view of A is identical in games G_1 and G_2 unless BAD happens, we fix any ciphertext c and let $m' := \mathsf{Dec}'(sk', c)$.

Similar to the observation made in the proof for Theorem 2.1.5, we first observe that if $m' \in \{\perp, s\}$, the simulation of H can never add a tuple of the form (c, K) to $\mathfrak{L}_D$ as a query to H on s results in abort. Hence, $\textsc{Dec}_{m,c}^{\not\perp}(c)$ will return an independent uniformly random key, like in game G_1, whenever queried on a ciphertext c such that $\mathsf{Dec}'(sk', c) \in \{\perp, s\}$.

It remains to analyse the case where $m' \notin \{\perp, s\}$. We first show that before the query to $\textsc{Dec}_m^{\not\perp}$ on c and the query to H on m', no entry of the form (c, K) could already exist in $\mathfrak{L}_D$ unless BAD occurs: Since neither $\textsc{Dec}_m^{\not\perp}$ was yet queried on c, nor was H queried yet on m', existence of an entry (c, K) in $\mathfrak{L}_D$ implies that H was queried on some message $m \neq m'$ such that $\mathsf{Enc}'(pk', m) = c$. Hence, $\mathsf{Dec}'(sk, \mathsf{Enc}'(pk', m)) = \mathsf{Dec}'(sk, c) = m' \neq m$, meaning that m induces a correctness error and BAD happened.

We will now further analyse the games' behaviour in the case that H was not queried on such an error-inducing message, i.e., conditioned on $\neg$BAD. We will show that

$\textsc{Dec}_m^{\not\perp}(c) = \mathsf{H}(m')$ if $m' \neq \perp$. We distinguish two sub-cases: A might either first query H on m', then $\textsc{Dec}_m^{\not\perp}$ on c, or the other way round.

- If H is queried on m' first, no entry of the form (c, K) already exists in $\mathfrak{L}_D$. Hence, besides adding $(m', K \leftarrow_\$ \mathcal{K})$ to $\mathfrak{L}_H$, H also computes $c' := \mathsf{Enc}'(pk, m')$. Since PKE' is assumed to be rigid and $m' \neq \perp$, we have that $c' = \mathsf{Enc}'(pk, \mathsf{Dec}'(sk, c)) = c$. By adding (c, K) to $\mathfrak{L}_D$ in line 30, H defines $\textsc{Dec}_m^{\not\perp}(c) := K = \mathsf{H}(m')$.
- If $\textsc{Dec}_m^{\not\perp}$ is queried on c first, it adds $(c, K \leftarrow_\$ \mathcal{K})$ to $\mathfrak{L}_D$, thereby defining $\textsc{Dec}_m^{\not\perp}(c) := K$. When queried on m' afterwards, H computes $c' := \mathsf{Enc}'(pk, m') = c$, and recognises that an entry of the form (c, K) already exists in $\mathfrak{L}_D$ in line 27. By adding (m', K) to $\mathfrak{L}_H$ and returning K, H defines $\mathsf{H}(m') := K = \textsc{Dec}_m^{\not\perp}(c)$.

We have shown that A's view is identical in both games unless a correctness error (in the form of BAD) occurs and

$$|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \leq \Pr[\mathrm{BAD}] \ .$$

Again, we can bound Pr[BAD] with a straightforward reduction to the game-based-correctness of PKE' and there exists an adversary C such that

$$\Pr[\mathrm{BAD}] \leq \mathrm{Adv}_{\mathsf{PKE}'}^{\mathsf{COR\text{-}RO}_{q_\mathsf{H}}}(\mathsf{C}) \ .$$

Game G_3. In game G_3, we abort immediately on the event that A queries H on m^*. We denote this event as CHAL. Due to the difference lemma,

$$|\Pr[G_2^{\mathsf{A}} \Rightarrow 1] - \Pr[G_3^{\mathsf{A}} \Rightarrow 1]| \leq \Pr[\mathrm{CHAL}] \ .$$

In game G_3, $\mathsf{H}(m^*)$ will not be given to A; neither through a hash nor a decryption query, meaning bit b is independent from A's view and hence,

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] = \frac{1}{2} \ .$$

It remains to bound Pr[CHAL]. To this end, we construct an adversary B in Figure 2.16 against the OW security of PKE', simulating G_3 for A.

Note that the simulation is perfect until CHAL occurs. The event that CHAL occurred implies that A queried $\mathsf{H}(m^*)$, and hence, $(m^*, K') \in \mathfrak{L}_H$ for some K'. Since Enc' is deterministic, we have that $\mathsf{Enc}'(pk, m^*) = c^*$, and thus B returns m^*.

$$\Pr[\mathrm{CHAL}] = \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}) \ .$$

```
$\mathsf{B}(pk, c^*)$
$K^* \leftarrow_\$ \mathcal{K}$
$b' \leftarrow \mathsf{A}^{\textsc{Dec}_m^{\not\perp},\mathsf{H}}(pk, c^*, K^*)$
if $\exists (m', K') \in \mathfrak{L}_H$
      s. th. $\mathsf{Enc}'(pk, m') = c^*$
   return $m'$
else abort
```

Fig. 2.16: Adversary B against OW for the proof of Theorem 2.1.7, where $\textsc{Dec}_m^{\not\perp}$ and H are defined as in Game G_2 of Figure 2.15.

Collecting the probabilities yields the required bound.

□

2.1.4 Combined FO-like Transformations: The Resulting KEMs

For completeness, we combine transformation T with $\{\mathsf{U}_{m,c}^{\not\perp}, \mathsf{U}_{m,c}^{\perp}, \mathsf{U}_{m}^{\not\perp}, \mathsf{U}_{m}^{\perp}\}$ from the previous sections to obtain four variants of the FO transformation.

To a public-key encryption scheme $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ and randomness space $\mathcal{R}$, and hash functions $\mathsf{G} : \mathcal{M} \to \mathcal{R}$ and $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n$, we associate

$$\begin{aligned}
\mathsf{KEM}_{m,c}^{\perp} &:= \mathsf{FO}_{m,c}^{\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}_{m,c}^{\perp}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}] = (\mathsf{KG}, \mathsf{Encaps}, \mathsf{Decaps}_{m,c}^{\perp}) \\
\mathsf{KEM}_{m,c}^{\not\perp} &:= \mathsf{FO}_{m,c}^{\not\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}_{m,c}^{\not\perp}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}] = (\mathsf{KG}^{\not\perp}, \mathsf{Encaps}, \mathsf{Decaps}_{m,c}^{\not\perp}) \\
\mathsf{KEM}_{m}^{\perp} &:= \mathsf{FO}_{m}^{\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}_{m}^{\perp}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}] = (\mathsf{KG}, \mathsf{Encaps}_m, \mathsf{Decaps}_{m}^{\perp}) \\
\mathsf{KEM}_{m}^{\not\perp} &:= \mathsf{FO}_{m}^{\not\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}_{m}^{\not\perp}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}] = (\mathsf{KG}^{\not\perp}, \mathsf{Encaps}_m, \mathsf{Decaps}_{m}^{\not\perp}) \ .
\end{aligned}$$

Their constituting algorithms are given in Figure 2.17.

We will now show how to concretely bound the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM} \in \{\mathsf{KEM}_{m,c}^{\perp}, \mathsf{KEM}_{m,c}^{\not\perp}, \mathsf{KEM}_{m}^{\perp}, \mathsf{KEM}_{m}^{\not\perp}\}$. To this end, we will first give a simplified presentation of our modular results: We will denote by $\mathsf{OW\text{-}ATK}_t(\mathsf{PKE})$ ($\mathsf{CPA}_t(\mathsf{PKE})$, $\mathsf{CCA}_t(\mathsf{KEM})$) the upper bound on the respective advantage of all adversaries running in time at most t.

As a first step, we gather our results for $\mathsf{PKE}' := \mathsf{T}[\mathsf{PKE}, \mathsf{G}]$. The bounds given in Theorem 2.1.2 and Theorem 2.1.3 share an additive term relative to the underlying scheme's correctness (δ) and spreadness (γ), and differ with regards to the tightness of the reduction to the underlying's scheme level of security. The following table gathers

```
$\mathsf{KG}^{\not\perp}$
$(pk, sk) \leftarrow \mathsf{KG}$
$s \leftarrow_\$ \mathcal{M}$
$sk' := (sk, s)$
return $(pk, sk')$
```

```
$\mathsf{Encaps}(pk)$   $\boxed{\mathsf{Encaps}_m(pk)}$
$m \leftarrow_\$ \mathcal{M}$
$c := \mathsf{Enc}(pk, m; G(m))$
$K := \mathsf{H}(m, c)$
$\boxed{K := \mathsf{H}(m)}$
return $(K, c)$
```

```
$\mathsf{Decaps}^{\perp}_{m,c}(sk, c)$   $\boxed{\mathsf{Decaps}^{\perp}_{m}(sk, c)}$
$m' := \mathsf{Dec}(sk, c)$
if $m' = \perp$ or $c \neq \mathsf{Enc}(pk, m'; \mathsf{G}(m'))$
    return $\perp$
else
    return $K := \mathsf{H}(m', c)$
    $\boxed{\textbf{return } K := \mathsf{H}(m')}$
```

```
$\mathsf{Decaps}^{\not\perp}_{m,c}(sk', c)$   $\boxed{\mathsf{Decaps}^{\not\perp}_{m}(sk', c)}$
Parse $(sk, s) := sk'$
$m' := \mathsf{Dec}(sk, c)$
if $m' = \perp$ or $c \neq \mathsf{Enc}(pk, m'; \mathsf{G}(m'))$
    return $K := \mathsf{H}(s, c)$
else
    return $K := \mathsf{H}(m', c)$
    $\boxed{\textbf{return } K := \mathsf{H}(m')}$
```

Fig. 2.17: Key encapsulation mechanisms $\mathsf{KEM}^{\perp}_{X} = (\mathsf{KG}, \mathsf{Encaps}_X, \mathsf{Decaps}^{\perp}_{X})$, and $\mathsf{KEM}^{\not\perp}_{X} = (\mathsf{KG}^{\not\perp}, \mathsf{Encaps}_X, \mathsf{Decaps}^{\not\perp}_{X})$, where subscript X is either m, c or m, obtained from $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$.

both results.

$\mathsf{OW\text{-}PVCA}_t(\mathsf{PKE}') \leq$	+ additive term	justification
$(q_{\mathsf{G}} + q_{\mathrm{PCO}} + 1) \cdot \mathsf{OW}_t(\mathsf{PKE})$	$(q_{\mathsf{G}} + q_{\mathrm{PCO}}) \cdot \delta_{\mathrm{wc}} + q_{\mathrm{VALID}} \cdot 2^{-\gamma}$	Theorem 2.1.2
$3 \cdot \mathsf{CPA}_t(\mathsf{PKE}) + \frac{2(q_{\mathsf{G}} + q_{\mathrm{PCO}})+1}{\lvert\mathcal{M}\rvert}$	$(q_{\mathsf{G}} + q_{\mathrm{PCO}}) \cdot \delta_{\mathrm{wc}} + q_{\mathrm{VALID}} \cdot 2^{-\gamma}$	Theorem 2.1.3

As a second step, we gather our results for $\mathsf{KEM} := \mathsf{U}[\mathsf{PKE}', \mathsf{H}]$, where $\mathsf{U} \in \{\mathsf{U}^{\perp}_{m,c}, \mathsf{U}^{\not\perp}_{m,c}, \mathsf{U}^{\perp}_{m}, \mathsf{U}^{\not\perp}_{m}\}$. In the respective theorems, it is specified how queries issued by A in the $\mathsf{IND\text{-}CCA}$ game translate to queries issued by B in its game against PKE'. As an example, recall that queries to H in the $\mathsf{IND\text{-}CCA}$ game for $\mathsf{KEM}^{\perp}_{m,c}$ trigger B to query PCO. The following table compares the four results, where the column $q_X \mapsto q_Y$ indicates that for the number q_X of queries that A issues to oracle X, the corresponding adversary B against the underlying scheme makes q_Y queries to oracle Y. Furthermore, $\mathsf{COR\text{-}RO}(\mathsf{PKE}', q)$ denotes the probability that an adversary creates a list of q many entries such that at least one entry exhibits decryption failure with respect to PKE'.

$\mathsf{U} =$	$\mathsf{CCA}_t(\mathsf{KEM}) \leq$	$q_X \mapsto q_Y$	justification
$\mathsf{U}^{\perp}_{m,c}$	$\mathsf{OW\text{-}PVCA}_t(\mathsf{PKE}') + \delta'_{\mathrm{ac}}$	$\mathsf{H} \mapsto \mathrm{PCO}$ $D \mapsto \mathrm{VALID}$	Theorem 2.1.4
$\mathsf{U}^{\not\perp}_{m,c}$	$\mathsf{OW\text{-}PCA}_t(\mathsf{PKE}') + \delta'_{\mathrm{ac}} + \frac{q_{\mathsf{H}}}{\lvert\mathcal{M}\rvert}$	$\mathsf{H} \mapsto \mathrm{PCO}$	Theorem 2.1.5
$\mathsf{U}^{\perp}_{m}$	$\mathsf{OW\text{-}VCA}_t(\mathsf{PKE}') + \mathsf{COR\text{-}RO}(\mathsf{PKE}', q_{\mathsf{H}})$	$\mathsf{H} \mapsto \mathsf{G}$ $D \mapsto \mathrm{VALID}$	Theorem 2.1.6
$\mathsf{U}^{\not\perp}_{m}$	$\mathsf{OW}_t(\mathsf{PKE}') + \mathsf{COR\text{-}RO}(\mathsf{PKE}', q_{\mathsf{H}}) + \frac{q_{\mathsf{H}}}{\lvert\mathcal{M}\rvert}$	$\mathsf{H} \mapsto \mathsf{G}$	Theorem 2.1.7

Combining the tables above, we obtain the following table which provides (simplified) concrete bounds of the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM} \in \{\mathsf{KEM}^{\perp}_{m,c}, \mathsf{KEM}^{\not\perp}_{m,c}, \mathsf{KEM}^{\perp}_{m}, \mathsf{KEM}^{\not\perp}_{m}\}$. Here $q_{\mathsf{RO}} := q_{\mathsf{G}} + q_{\mathsf{H}}$ counts the total number of A's queries to the random oracles G and H, and q_D counts the number of A's decryption queries. We make use of the observation that $\delta_{\mathrm{ac}} \leq \delta_{\mathrm{wc}}$.

KEM	$\mathsf{CCA}_t(\mathsf{KEM}) \leq$
$\mathsf{KEM}^{\perp}_{m,c}$, $\mathsf{KEM}^{\perp}_{m}$	$(q_{\mathsf{RO}} + 1) \cdot \delta_{\mathrm{wc}} + q_D \cdot 2^{-\gamma} + \begin{cases} q_{\mathsf{RO}} \cdot \mathsf{OW}_t(\mathsf{PKE}) \\ 3 \cdot \mathsf{CPA}_t(\mathsf{PKE}) + q_{\mathsf{RO}}/\lvert\mathcal{M}\rvert \end{cases}$
$\mathsf{KEM}^{\not\perp}_{m,c}$,$\mathsf{KEM}^{\not\perp}_{m}$	$(q_{\mathsf{RO}} + 1) \cdot \delta_{\mathrm{wc}} + q_{\mathsf{RO}}/\lvert\mathcal{M}\rvert + \begin{cases} q_{\mathsf{RO}} \cdot \mathsf{OW}_t(\mathsf{PKE}) \\ 3 \cdot \mathsf{CPA}_t(\mathsf{PKE}) \end{cases}$

Concrete parameters. For "κ bits of security", one generally requires that for all adversaries A with advantage $\mathrm{Adv}(\mathsf{A})$ and running time $\mathrm{Time}(\mathsf{A})$, we have

$$\frac{\mathrm{Time}(\mathsf{A})}{\mathrm{Adv}(\mathsf{A})} \geq 2^{\kappa} \quad .$$

The table below gives recommendations for the information-theoretic terms that appear in the concrete security bounds above, namely δ_{wc} (worst-case correctness error of PKE), γ (γ-spreadness of PKE), and $\mathcal{M}$ (message space of PKE).

Term in concrete bound	Minimal requirement for κ bits security
$q_{\mathsf{RO}} \cdot \delta_{\mathrm{wc}}$	$\delta_{\mathrm{wc}} \leq 2^{-\kappa}$
$q_{\mathsf{RO}} \cdot 2^{-\gamma}$	$\gamma \geq \kappa$
$q_{\mathsf{RO}}/\lvert\mathcal{M}\rvert$	$\lvert\mathcal{M}\rvert \geq 2^{\kappa}$

For example, if the concrete security bound contains the term $q_{\mathsf{RO}} \cdot \delta_{\mathrm{wc}}$, then with $\delta_{\mathrm{wc}} \leq 2^{-\kappa}$ one has

$$\frac{\mathrm{Time}(\mathsf{A})}{\mathrm{Adv}(\mathsf{A})} \geq \frac{q_{\mathsf{RO}}}{q_{\mathsf{RO}} \cdot \delta_{\mathrm{wc}}} = \frac{1}{\delta_{\mathrm{wc}}} \geq 2^{\kappa},$$

as required for κ bits security.

2.2 Modular constructions in the QROM

In this section, we will revisit our modular approach from Section 2.1. In order to lift the proof strategies used in Section 2.1 to the quantum random oracle model, we require a slight generalisation of original one-way to hiding. We describe and prove this generalisation in Section 2.2.1.

T: From OW to $\mathsf{OW\text{-}PCA}$ security in the quantum ROM. First, we will first reconsider transformation T from Section 2.1.1 (see Figure 2.2, page 62) in Section 2.2.2. We show that T achieves $\mathsf{OW\text{-}PCA}$ security also in the quantum random oracle model. Since quantum queries to G are in superposition, both our handling of correctness errors and our reduction itself are slightly more involved and yield non-tight bounds, as the former involves a quantum search problem, and the latter involves the extraction argument from Section 2.2.1.

QU_m: From $\mathsf{OW\text{-}PCA}$ to $\mathsf{IND\text{-}CCA}$ security in the quantum ROM. Next, to go from $\mathsf{OW\text{-}PCA}$ to $\mathsf{IND\text{-}CCA}$ in the quantum random oracle model, we modify transformations $\mathsf{U}^{\perp}_m$ and $\mathsf{U}^{\not\perp}_m$ (that were defined in Figure 2.12, on page 76):

We construct a key encapsulation mechanism $\mathsf{QKEM}^{\perp}_m := \mathsf{QU}^{\perp}_m[\mathsf{PKE}', \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}]$ with explicit rejection by defining

$$\mathsf{QEncaps}_m(pk) := ((c \leftarrow \mathsf{Enc}'(pk, m), d := \mathsf{H}'_{\mathsf{conf}}(m)), K := \mathsf{H}(m)) \ ,$$

where m is picked at random from the message space, and

$$\mathsf{QDecaps}^{\perp}_m(sk, (c, d)) := \begin{cases} \mathsf{H}(m') & m' \neq \perp \textbf{ and } \mathsf{H}'_{\mathsf{conf}}(m') = d \\ \perp & m' = \perp \textbf{ or } \mathsf{H}'_{\mathsf{conf}}(m') \neq d \end{cases} ,$$

where $m' := \mathsf{Dec}(sk, c)$. Transformation $\mathsf{QU}^{\perp}_m$ differs from $\mathsf{U}^{\perp}_m$ only in the additional hash value $d = \mathsf{H}'_{\mathsf{conf}}(m)$ included in the ciphertext, which is used for consistency checking. Including this additional value is sometimes called “key confirmation”. Note that $\mathsf{H}'_{\mathsf{conf}}$ is required to have matching domain and range.

Similarly, we construct an implicit rejection variant of the key encapsulation mechanism above, i.e., we define $\mathsf{QKEM}^{\not\perp}_m := \mathsf{QU}^{\not\perp}_m[\mathsf{PKE}', \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}, \mathsf{H}''_{\mathsf{reject}}]$, which differs from $\mathsf{QKEM}^{\perp}_m$ only in decapsulation:

$$\mathsf{QDecaps}^{\not\perp}_m(sk, (c, d)) := \begin{cases} \mathsf{H}(m') & m' \neq \perp \textbf{ and } \mathsf{H}'_{\mathsf{conf}}(m') = d \\ \mathsf{H}''_{\mathsf{reject}}(c, d) & m' = \perp \textbf{ or } \mathsf{H}'_{\mathsf{conf}}(m') \neq d \end{cases} .$$

Modeling H, $\mathsf{H}'_{\mathsf{conf}}$ and $\mathsf{H}''_{\mathsf{reject}}$ as quantum random oracles, we show in Section 2.2.3 that $\mathsf{IND\text{-}CCA}$ security of $\mathsf{QKEM}_m^{\perp}$ and $\mathsf{QKEM}_m^{\not\perp}$ non-tightly reduces to $\mathsf{OW\text{-}PCA}$ security of PKE'.

The resulting FO transformations. Combining T with $\mathsf{QU}_m^{\perp}$ and $\mathsf{QU}_m^{\not\perp}$, in Section 2.2.4 we provide concrete bounds for the $\mathsf{IND\text{-}CCA}$ security of

$$\mathsf{QFO}_m^{\perp}[\mathsf{PKE},\mathsf{G},\mathsf{H},\mathsf{H}'_{\mathsf{conf}}] := \mathsf{QU}_m^{\perp}[\mathsf{T}[\mathsf{PKE},\mathsf{G}],\mathsf{H},\mathsf{H}'_{\mathsf{conf}}]$$

and

$$\mathsf{QFO}_m^{\not\perp}[\mathsf{PKE},\mathsf{G},\mathsf{H},\mathsf{H}'_{\mathsf{conf}}] := \mathsf{QU}_m^{\not\perp}[\mathsf{T}[\mathsf{PKE},\mathsf{G}],\mathsf{H},\mathsf{H}'_{\mathsf{conf}},\mathsf{H}''_{\mathsf{reject}}]$$

in the quantum random oracle model.

As a corollary, we obtain that $\mathsf{IND\text{-}CCA}$ security of both $\mathsf{QFO}_m^{\perp}[\mathsf{PKE},\mathsf{G},\mathsf{H},\mathsf{H}'_{\mathsf{conf}}]$ and $\mathsf{QFO}_m^{\not\perp}[\mathsf{PKE},\mathsf{G},\mathsf{H},\mathsf{H}'_{\mathsf{conf}}]$ reduces to the OW security of PKE. Our transformation $\mathsf{QFO}_m^{\perp}$ essentially recovers a KEM variant of the modified FO transformation that was defined by Targhi and Unruh [TU16].

2.2.1 Algorithmic One-Way to Hiding

In this section, we formalise our slightly more general variant of "original one-way to hiding" (see Theorem 1.3.2, page 43), which we will use during our security proofs in the following sections. To a quantum-accessible oracle O and an algorithm A that has quantum access to O (and that possibly has access to some collection $\mathsf{Oracles} := \{\mathsf{Oracle}_1, \cdots, \mathsf{Oracle}_N\}$ of additional oracles), we associate the following extractor algorithm $\mathsf{EXT}[\mathsf{A}^{|\mathsf{O}\rangle,\mathsf{Oracles}},\mathsf{O}]$, which executes $\mathsf{A}^{|\mathsf{O}\rangle,\mathsf{Oracles}}$ until a randomly chosen quantum query to O, measures this query's input register, and returns the measurement result x'. (If A issues $q_{\mathsf{O},\mathsf{Oracles}}$ queries to its collection $\mathsf{Oracles}$ of additional oracles, these queries are included in the choice which query is measured.)

Theorem 2.2.1. (Algorithmic One-Way to Hiding ($\mathsf{AOW2H}$)) Let $\mathsf{O} : \{0,1\}^n \to \{0,1\}^m$ be a random oracle, and let A be a quantum algorithm with binary output, issuing at most q_{O} explicit (quantum) queries to O, and triggering at most $q_{\mathsf{O},\mathsf{Oracles}}$ queries to O by its queries to $\mathsf{Oracles} = \{\mathsf{Oracle}_1, \cdots, \mathsf{Oracle}_N\}$.

Furthermore, let GenInp be an algorithm that takes as input bitstrings in $\{0,1\}^{n+m}$ and returns some input *inp*. If GenInp does not make any queries to O, we have that

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq 2(q_{\mathsf{O}} + q_{\mathsf{O},\mathsf{Oracles}}) \cdot \sqrt{p_{\text{FIND}}} \ , \tag{2.6}$$

```
$\mathsf{EXT}[\mathsf{A}^{|\mathsf{O}\rangle,\mathsf{Oracles}},\mathsf{O}](inp)$
$i \leftarrow_\$ \{1,\cdots,q_{\mathsf{O}} + q_{\mathsf{O},\mathsf{Oracles}}\}$
Run $\mathsf{A}^{|\mathsf{O}\rangle,\mathsf{Oracles}}(inp)$ until the $i$th query $|\varphi\rangle_X \, \langle\psi\rangle_Y$ to $\mathsf{O}$
if $i >$ number of queries to $\mathsf{O}$
    return $\perp$
else
    $x' \leftarrow \mathsf{Measure}(|\varphi\rangle_X)$
    return $x'$
```

Fig. 2.18: Extractor algorithm EXT for algorithmic one-way to hiding.

where games G_b (for bit b) are defined below and

$$p_{\text{FIND}} := \Pr[x' = x^*]$$

with the probability taken over $x^* \leftarrow_\$ \{0,1\}^n$, $y^* \leftarrow_\$ \{0,1\}^m$, $inp \leftarrow \mathsf{GenInp}(x^*, y^*)$, and $x' \leftarrow \mathsf{EXT}[\mathsf{A}^{|\mathsf{O}\rangle,\mathsf{Oracles}},\mathsf{O}](inp)$.

If all additional oracles are only classically accessible, and none ever triggers a query to O on x^*, then we can replace the upper bound above with

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq 2q_{\mathsf{O}} \cdot \sqrt{p_{\text{FIND}}}\ , \tag{2.7}$$

where the extractor's query choice only considers the q_{O} many explicit queries to O.

```
GAME $G_b$
$x^* \leftarrow_\$ \{0,1\}^n$
$y_0^* := \mathsf{O}(x)$, $y_1^* \leftarrow_\$ \{0,1\}^m$
$inp \leftarrow \mathsf{GenInp}(x^*, y_b^*)$
$b' \leftarrow \mathsf{A}^{|\mathsf{O}\rangle,\mathsf{Oracles}}(inp)$
```

The difference between original and algorithmic one-way to hiding is that the original one-way to hiding lemma ([Unr14b, Lemma 5]) only considers the case that $\mathsf{GenInp}(x^*, y^*) := (x^*, y^*)$, and that no additional oracles $\mathsf{Oracle}_1, \cdots, \mathsf{Oracle}_\mathsf{N}$ are accessed by A. However, Theorem 2.2.1 is a straightforward corollary to [Unr14b, Lemma 5]: A reduction to the original variant is run on input (x^*, y_b^*) and can hence compute $inp \leftarrow \mathsf{GenInp}(x^*, y_b^*)$ on its own (without any additional queries to O). Since the reduction has to provide access to $\mathsf{Oracles}$ (which might happen to be be defined relative to O), the providing of the oracles results in at most $q_{\mathsf{O},\mathsf{Oracles}}$ many additional queries to O.

In order to prove the second statement (the bound given in Equation (2.7)), we have to show how to get rid of the additional summand $q_{\mathsf{O},\mathsf{Oracles}}$ in Equation (2.6).

Intuitively, we can do so because in the view of the additional oracles, x^* is removed from the domain of O. We will now make this argument more formal.

Random oracle O can be dissected into the tuple $(x^*, \mathsf{O}(x^*)$ and its mapping rule on $X \setminus \{x^*\}$. Equivalently, O can be defined by drawing two random oracles O_1, O_2, and letting $\mathsf{O}(x^*) := \mathsf{O}_1(x^*)$ and $\mathsf{O}(x) := \mathsf{O}_2(x)$ anywhere else. As a warm-up, we observe that this equivalent description allow us to give a reduction B that also is run in an $\mathsf{AOW2H}$ game, but here we identify O_1 as B's oracle O', and O_2 as its (only) additional oracle Oracle_1'. (In order to avoid confusion: While we assume all of A's additional oracles to be accessible only classically, we will still model B's additional oracle Oracle_1' as quantum-accessible.) We furthermore identify the input of B with (x^*, inp), where $\mathit{inp} \leftarrow \mathsf{GenInp}(x^*, y_b^*)$ is the input generated according to A's game. Having quantum access to both O_1 and O_2, B can trivially simulate O for A as well as all of A's additional oracles and

$$\Pr[G_b^{\mathsf{A}} \Rightarrow 1] = \Pr[H_b^{\mathsf{B}} \Rightarrow 1],$$

where H_b denotes B's $\mathsf{AOW2H}$ games. (Note that the simulation of O only works since B knows x^*.)

The reduction above, however, still suffers from the classical queries triggered by A's additional oracle queries. As the second step, we therefore change B as follows: Before executing A, B obtains the complete table of O on $X \setminus \{x^*\}$, i.e., the collection $(x, \mathsf{O}_2(x))_{x \neq x^*}$, by querying its additional $\mathsf{Oracle}_1' = \mathsf{O}_2$ on each (classical) value $x \neq x^*$. (Note that our reduction is not required to be bounded, as [Unr14b, Lemma 5] (and consequentially, Equation (2.6)) is information-theoretical and only depends on the number of queries to $\mathsf{O}' = \mathsf{O}_1$.) Clearly, none of these preparation queries trigger a query to B's oracle $\mathsf{O}' = \mathsf{O}_1$. When queried on any additional oracle Oracle_n, B can now use the table to answer consistently, without having to issue any queries to one of its oracles. Since the only queries to O' are now triggered by A's explicit queries to O,

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| = |\Pr[H_0^{\mathsf{B}} \Rightarrow 1] - \Pr[H_1^{\mathsf{B}} \Rightarrow 1]| \leq 2q_{\mathsf{O}} \cdot \sqrt{p_{\mathrm{FIND}}} \ ,$$

and the resulting extractor randomly picks one of A's explicit queries to O.

Note that this argument only works since access to all oracles Oracle_n is assumed to be classical, and their execution is independent of $\mathsf{O}(x^*)$, as otherwise, the table would not be sufficient to simulate the additional oracles.

2.2.2 Transformation T*: from* OW *to* $\mathsf{OW\text{-}PCA}$

Recall transformation T from Figure 2.2 (see page 62). We have shown in Section 2.1.1 that T transforms an OW secure public-key encryption scheme into an $\mathsf{OW\text{-}PCA}$ secure one, in the random oracle, and will now adapt the proof such that it accounts for quantum random oracle queries. Note that $\mathsf{OW\text{-}PVCA}$ security is not required here, since the transformations from the next section only require $\mathsf{OW\text{-}PCA}$ security.

Correctness. Similar to the statement of Theorem 2.1.1, we first establish that if PKE is worst-case correct, then PKE' achieves game-based correctness (see Definition 1.1.13, page 31), in the quantum random oracle model. The handling of decryption failure is slightly more involved, since the random oracle G which determines decryption failure is now searchable with quantum access.

Lemma 2.2.2. If PKE is δ_{wc}-worst-case correct, then for any adversary A, issuing q_{G} (quantum) queries to G and returning one message it holds that

$$\mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{COR\text{-}RO}_1}(\mathsf{A}) \leq 8 \cdot (q_{\mathsf{G}} + 1)^2 \cdot \delta_{\mathrm{wc}} \ .$$

Proof. Consider an (unbounded, quantum) adversary A in game $\mathsf{COR\text{-}RO}$, issuing q_{G} queries to G. For fixed $(pk, sk) \in \mathrm{supp}(\mathsf{KG})$ and message $m \in \mathcal{M}$, we denote by

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

the set of "bad" randomness. We further define

$$\delta(pk, sk, m) := |\mathcal{R}_{\mathrm{bad}}(pk, sk, m)| / |\mathcal{R}| \tag{2.8}$$

as the fraction of bad randomness, and

$$\delta(pk, sk) := \max_{m \in \mathcal{M}} \delta(pk, sk, m) \ .$$

Note that with this notation, we have that $\delta_{\mathrm{wc}} = \mathbb{E}[\delta(pk, sk)]$, where the expectation is taken over $(pk, sk) \leftarrow \mathsf{KG}$.

To upper bound $\Pr[\mathsf{COR\text{-}RO}^{\mathsf{A}} \Rightarrow 1]$, we construct an (unbounded, quantum) adversary A in Figure 2.19 against the generic search problem with bounded probabilities GSPB_λ defined in Figure 1.13 (see page 46).

A runs $(pk, sk) \leftarrow \mathsf{KG}$, and computes the Bernoulli parameters $\lambda(m)$ of its generic search problem as $\lambda(m) := \delta(pk, sk, m)$, which are bounded by $\lambda := \delta(pk, sk) =$

```
$\mathsf{B}_1$
01 $(pk, sk) \leftarrow \mathsf{KG}$
02 for $m \in \mathcal{M}$
03     $\lambda(m) := \delta(pk, sk, m)$
04 return $(\lambda(m))_{m \in \mathcal{M}}$

$\mathsf{B}_2^{\langle F \rangle}$
05 Pick $2q_{\mathsf{H}}$-wise hash $f$
06 $m \leftarrow \mathsf{A}^{|\mathsf{G}\rangle}(pk, sk)$
07 return $m$

$\mathsf{G}(m)$
08 if $F(m) = 0$
09     $\mathsf{G}(m) := \mathsf{Sample}(\mathcal{R} \setminus \mathcal{R}_{\text{bad}}(pk, sk, m); f(m))$
10 else
11     $\mathsf{G}(m) := \mathsf{Sample}(\mathcal{R}_{\text{bad}}(pk, sk, m); f(m))$
12 return $\mathsf{G}(m)$
```

Fig. 2.19: Adversary A, executed in game $\mathsf{GSPB}_{\delta(pk,sk)}$ (with quantum access to F), for the proof of Lemma 2.2.2. $\delta(pk, sk, m)$ is defined in Equation (2.8). f (lines 09 and 11) is an internal $2q_{\mathsf{G}}$-wise independent hash function that cannot be accessed by A. $\mathsf{Sample}(Y)$ is a probabilistic algorithm that returns a uniformly distributed $y \leftarrow_\$ Y$. $\mathsf{Sample}(Y; f(m))$ denotes the deterministic execution of $\mathsf{Sample}(Y)$, using explicitly given randomness $f(m)$.

$\max_{m \in \mathcal{M}} \Pr[\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) \neq m]$.

To analyze A, we first fix (pk, sk). For each $m \in \mathcal{M}$, by the definition of game GSPB_λ, the random variable $F(m)$ is distributed according to $B_{\lambda(m)} = B_{\delta(pk,sk,m)}$. When running A, B_2 provides quantum access to G that is defined relative to F. To be more precise, $\mathsf{G}(m)$ is sampled from the set of bad randomness in line 09 if $F(m) = 1$, and from its complement in line 11 if $F(m) = 0$. Note that by construction, $\mathsf{G}(m)$ is uniformly distributed in $\mathcal{R}$, and G hence is a random oracle.

A wins its game $\mathsf{COR\text{-}RO}$ iff it returns a message m such that $\mathsf{G}(m) \in \mathcal{R}_{\text{bad}}(pk, sk, m)$, as then and only then it will hold that $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m; \mathsf{G}(m))) \neq m$. The condition that $\mathsf{G}(m) \in \mathcal{R}_{\text{bad}}(pk, sk, m)$ is equivalent to $F(m) = 1$, in which case A wins its game GSP_λ. To summarise, conditioned on a fixed (pk, sk), we can apply Lemma 1.3.6 to obtain

$$\Pr[\mathsf{COR\text{-}RO}^{\mathsf{A}} \Rightarrow 1 \mid (pk, sk)] \leq \Pr[\mathsf{GSP}^{\mathsf{A}}_{\delta(pk,sk)} \Rightarrow 1] \leq 8 \cdot \delta(pk, sk) \cdot (q_{\mathsf{G}} + 1)^2 \ ,$$

and by taking the expectation over $(pk, sk) \leftarrow \mathsf{KG}$, we obtain

$$\Pr[\mathsf{COR\text{-}RO}^{\mathsf{A}} \Rightarrow 1] \leq 8 \cdot \delta_{\text{wc}} \cdot (q_{\mathsf{G}} + 1)^2 \ .$$

□

OW-PCA SECURITY FROM OW. The following theorem establishes that OW-PCA security of PKE′ reduces to the OW security of PKE, in the quantum random oracle model.

Theorem 2.2.3 (PKE OW $\overset{\text{QROM}}{\Rightarrow}$ PKE' $\mathsf{OW\text{-}PCA}$)**.** Assume PKE to be δ_{wc}-worst-case correct. For any $\mathsf{OW\text{-}PCA}$ adversary A, issuing at most q_{G} (quantum) queries to random oracle G and q_{PCO} (classical) queries to plaintext checking oracle PCO, there exist OW adversaries B_1 and B_2 such that

$$\begin{aligned}\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}_1) + 2 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}}) \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}_2)} \\ + 8 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}} + 1)^2 \cdot \delta_{\mathrm{wc}} \ ,\end{aligned} \tag{2.9}$$

and the running time of B_1 and B_2 is about that of A. If none of A's queries to PCO is of the form $(m^*, -)$, then we can replace the upper bound above with

$$\begin{aligned}\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{A}) \leq \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}_1) + 2q_{\mathsf{G}} \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}_2)} \\ + 8 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}} + 1)^2 \cdot \delta_{\mathrm{wc}} \ .\end{aligned} \tag{2.10}$$

Below we will prove Equation (2.9), which can be sketched as follows: Similar to the proof of Theorem 2.1.2, our proof first implements the PCO oracle via "re-encryption" in game G_1. Next, we use algorithmic oneway to hiding ($\mathsf{AOW2H}$, see Theorem 2.2.1 on page 88) to argue that we can decouple the challenge ciphertext $c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$ from random oracle G, even if it is quantum-accessible. The decoupling can be upper bounded (non-tightly) in terms of extracting m^* (see B_2), and having a random challenge c^* allows for an ensuing trivial reduction to OW security (see B_1). The approach of decoupling and then extracting via $\mathsf{AOW2H}$ is loosely based on [TU16].

In order to verify Equation (2.10), it remains to show how to drop summand q_{PCO} from the loss in B_2's advantage, assuming that none of A's queries to PCO is of the form $(m^*, -)$. The summand q_{PCO} stems from application of $\mathsf{AOW2H}$, where we identify G with O, m^* with x, and oracle PCO with additional oracle Oracle_1. At the point where we apply $\mathsf{AOW2H}$, the PCO oracle is implemented via "re-encryption". If no query of PCO is of the form $(m^*, -)$, then no query to PCO can trigger a query to G on m^*. Since PCO is only classically accessible, and no query to PCO can trigger a query to G on m^*, we can use the improved bound (Equation (2.7)) and drop summand q_{PCO}.

Proof. Let A be an adversary against the $\mathsf{OW\text{-}PCA}$ security of PKE', issuing at most q_{G} queries to G and q_{PCO} queries to PCO. Consider the sequence of games given in Figure 2.20.

```
GAME $G_0$-$G_3$
01 $m^* \leftarrow_\$ \mathcal{M}$
02 $r^* := \mathsf{G}(m^*)$                                  // $G_0$-$G_1$
03 $r^* \leftarrow_\$ \mathcal{R}$                              // $G_2$-$G_3$
04 $c^* := \mathsf{Enc}(pk, m^*; r^*)$
05 $m' \leftarrow \mathsf{A}^{|\mathsf{G}\rangle,\mathrm{PCO}}(pk, c^*)$                   // $G_1$-$G_2$
06 $m' \leftarrow \mathsf{EXT}[\mathsf{A}^{|\mathsf{G}\rangle,\mathrm{PCO}}, \mathsf{G}](pk, c^*)$        // $G_3$
07 return $[\![m' = m^*]\!]$

PCO($m \in \mathcal{M}, c$)
08 $m' := \mathsf{Dec}(sk, c)$                                 // $G_0$
09 return $[\![m' = m]\!]$
     and $[\![\mathsf{Enc}(pk, m'; \mathsf{G}(m')) = c]\!]$           // $G_0$
10 return $[\![\mathsf{Enc}(pk, m; \mathsf{G}(m)) = c]\!]$         // $G_1$-$G_3$
```

Fig. 2.20: Games G_0 - G_3 for the proof of Theorem 2.2.3.

Game G_0. Since game G_0 is the original OW-PCA game,

$$\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{A}) = \Pr[G_0^{\mathsf{A}} \Rightarrow 1] \ .$$

Game G_1. In game G_1, the plaintext checking oracle PCO is replaced with a simulation that doesn't make use of the secret key anymore. We claim

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq 8 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}} + 1)^2 \cdot \delta_{\mathrm{wc}} \ . \tag{2.11}$$

To show Equation (2.11), first note that both game G_0 and game G_1 proceed identically unless A submits a PCO query (m, c) such that $c = \mathsf{Enc}(pk, m; \mathsf{G}(m))$, but $\mathsf{Dec}(sk, c) \neq m$. We call this event BAD. Since both game G_0 and game G_1 proceed identically conditioned on the event that BAD does not happen,

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq \Pr[\mathrm{BAD}] \ .$$

Similar to the proof of Theorem 2.1.2, one can again show that there exists an adversary F against COR-RO that perfectly simulates games G_0 and G_1 until BAD happens: Since F holds the secret key, it can check for event BAD on each query to $\mathrm{PCO}(m, c)$, and immediately abort A and return m to its game COR-RO if BAD occurs. Note that during this check, re-encryption triggers F to issue an additional query to G. Applying Lemma 2.2.2, we obtain

$$\Pr[\mathrm{BAD}] \leq \Pr[\mathsf{COR\text{-}RO}^{\mathsf{F}}] \leq 8 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}} + 1)^2 \cdot \delta_{\mathrm{wc}} \ .$$

Game G_2. In game G_2, we replace $r^* := \mathsf{G}(m^*)$ with uniform randomness r^* in line 02. Now that r^* is uniformly random, we can trivially construct a first one-way adversary B_1 in Figure 2.21 against the original encryption scheme PKE, simulating game G_2 for

```
$\mathsf{B}_1(pk, c^*)$
    $m' \leftarrow \mathsf{A}^{|\mathsf{G}\rangle,\mathrm{PCO}}(pk, c^*)$
    return $m'$
```

```
$\mathsf{B}_2(pk, c^*)$
    $m' \leftarrow \mathsf{EXT}[\mathsf{A}^{|\mathsf{G}\rangle,\mathrm{PCO}}, \mathsf{G}](pk, c^*)$
    return $m'$
```

Fig. 2.21: Adversaries B_1 and B_2 for the proof of Theorem 2.2.3. Oracle PCO is defined as in game G_2 of Figure 2.20.

A. B_1 outputs $m' = m^*$ if A wins in game G_2.

$$\Pr[G_2^{\mathsf{A}} \Rightarrow 1] = \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}_1) \ .$$

So far, we have shown that

$$\mathrm{Adv}_{\mathsf{PKE}'}^{\mathsf{OW\text{-}PCA}}(\mathsf{A}) \leq 8 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}} + 1)^2 \cdot \delta_{\mathrm{wc}} + \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}_1) \\ + |\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \ .$$

To upper bound $|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]|$, we will apply Theorem 2.2.1 (AOW2H): We identify x with m^*, y with r^*, and define algorithm GenInp in Figure 2.22. Taking into account that A additionally has access to PCO, which triggers exactly one query to G per incovation, we obtain

$$|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \leq 2 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}}) \cdot \sqrt{\Pr[G_3^{\mathsf{A}} \Rightarrow 1]} \ ,$$

where the extractor algorithm EXT used in game G_3 is defined as in Figure 2.18 (see page 89). (Recall that EXT represents execution of A until a randomly chosen query to G, which is then measured to extract a message m'.)

Finally, we construct another one-way adversary B_2 in Figure 2.21 against the original encryption scheme PKE, simulating game G_3 for A.

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] = \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{OW}}(\mathsf{B}_2) \ .$$

```
Algorithm GenInp$(m^*, r^*)$
    $(pk, sk) \leftarrow \mathsf{KG}$
    $c^* := \mathsf{Enc}(pk, m^*; r^*)$
    $inp = (pk, c^*)$
    return $inp$
```

Fig. 2.22: Input generation algorithm GenInp for the proof of Theorem 2.2.3.

□

2.2.3 Transformations $\mathsf{QU}^{\perp}_{m}$, $\mathsf{QU}^{\not\perp}_{m}$*: From* OW-PCA *to* IND-CCA

In this section, we define our quantum variations of $\mathsf{U}^{\perp}_{m}$ and $\mathsf{U}^{\not\perp}_{m}$ (for the original transformations, see Figure 2.12, page 76).

The Constructions. To a public-key encryption scheme $\mathsf{PKE}' = (\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}')$ with message space $\mathcal{M} = \{0,1\}^m$, and hash functions $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n$, $\mathsf{H}'_{\mathsf{conf}} : \mathcal{M} \to \mathcal{M}$, and $\mathsf{H}''_{\mathsf{reject}} : \{0,1\}^* \to \{0,1\}^n$, we associate

$$\mathsf{QKEM}^{\perp}_{m} := \mathsf{QU}^{\perp}_{m}[\mathsf{PKE}', \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}] := (\mathsf{KG}', \mathsf{QEncaps}_m, \mathsf{QDecaps}^{\perp}_{m})$$

and

$$\mathsf{QKEM}^{\not\perp}_{m} := \mathsf{QU}^{\not\perp}_{m}[\mathsf{PKE}', \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}, \mathsf{H}''_{\mathsf{reject}}] := (\mathsf{KG}', \mathsf{QEncaps}_m, DecapsQImpMess) \ .$$

The algorithms of $\mathsf{QKEM}^{\perp}_{m}$ and $\mathsf{QKEM}^{\not\perp}_{m}$ are defined in Figure 2.23.

Like $\mathsf{KEM}^{\perp}_{m}$ and $\mathsf{KEM}^{\not\perp}_{m}$, $\mathsf{QKEM}^{\perp}_{m}$ and $\mathsf{QKEM}^{\not\perp}_{m}$ essentially differ in how they reject: While $\mathsf{QDecaps}^{\perp}_{m}$ rejects by returning $\perp$, $\mathsf{QDecaps}^{\not\perp}_{m}$ uses an additional random oracle $\mathsf{H}''_{\mathsf{reject}}$ to return a random key.

Furthermore, $\mathsf{QKEM}^{\perp}_{m}$ ($\mathsf{QKEM}^{\not\perp}_{m}$) essentially differ from $\mathsf{KEM}^{\perp}_{m}$ ($\mathsf{KEM}^{\not\perp}_{m}$) by including key confirmation value $d = \mathsf{H}'_{\mathsf{conf}}(m)$ in the ciphertext, and using d to check validity of the ciphertext during decapsulation. We stress that hash function $\mathsf{H}'_{\mathsf{conf}}$ has matching domain and range $\mathcal{M} = \{0,1\}^m$.

```
$\mathsf{QEncaps}_m(pk)$
$m \leftarrow_\$ \mathcal{M}$
$c \leftarrow \mathsf{Enc}'(pk, m)$
$d := \mathsf{H}'_{\mathsf{conf}}(m)$
$K := \mathsf{H}(m)$
return $(K, c, d)$

$\boxed{\mathsf{QDecaps}^{\perp}_{m}(sk, c, d)}$  $\mathsf{QDecaps}^{\perp}_{m}(sk, c, d)$
$m' := \mathsf{Dec}'(sk, c)$
if $m' = \perp$ or $\mathsf{H}'_{\mathsf{conf}}(m') \neq d$
    $\boxed{\text{return } \perp}$
    return $K := \mathsf{H}''_{\mathsf{reject}}(c, d)$
else return $K := \mathsf{H}(m')$
```

Fig. 2.23: Key encapsulation mechanisms $\mathsf{QKEM}^{\perp}_{m} = (\mathsf{KG}', \mathsf{QEncaps}_m, \mathsf{QDecaps}^{\perp}_{m})$ and $\mathsf{QKEM}^{\not\perp}_{m} = (\mathsf{KG}', \mathsf{QEncaps}_m, DecapsQImpMess)$.

Security of $\mathsf{QKEM}^{\perp}_{m}$. The following theorem establishes that IND-CCA security of $\mathsf{QKEM}^{\perp}_{m}$ reduces to OW-PCA security of PKE', in the quantum random oracle model.

Theorem 2.2.4 (PKE' OW-PCA $\overset{\text{QROM}}{\Rightarrow}$ $\mathsf{QKEM}^{\perp}_{m}$ IND-CCA)**.** For any quantum adversary A issuing at most q_D (classical) queries to the decapsulation oracle $\textsc{QDecaps}^{\perp}_{m}$, at most q_{H} (quantum) queries to random oracle H, and at most $q_{\mathsf{H}'_{\mathsf{conf}}}$ (quantum) queries to random oracle $\mathsf{H}'_{\mathsf{conf}}$, there exist $\mathsf{OW\text{-}PCA}$ adversaries B_0 and B_1, issuing at most $2q_D q_{\mathsf{H}'_{\mathsf{conf}}}$ queries to oracle PCO, such that

$$\begin{aligned}\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{QKEM}^{\perp}_{m}}(\mathsf{A}) \leq (q_{\mathsf{H}} + q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}_0)} \\ + (q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}_1)} \ , \end{aligned} \tag{2.12}$$

and the running time of B_0 and B_1 is about that of A. If PKE' furthermore is deterministic and rigid, there exist $\mathsf{OW\text{-}PCA}$ adversaries B'_0 and B'_1 such that

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{QKEM}^{\perp}_{m}}(\mathsf{A}) \leq (q_{\mathsf{H}} + q_{\mathsf{H}'_{\mathsf{conf}}}) \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}'_0)} + q_{\mathsf{H}'_{\mathsf{conf}}} \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}'_1)} + \frac{q_D}{2^{m-1}} \ , \tag{2.13}$$

the running time of B'_0 and B'_1 is about that of A, and no query to $\mathrm{QD\small{ECAPS}}^{\perp}_{m}$ will ever trigger either adversary B'_{b} to issue a query to PCO on $(m^*, -)$.

We will first prove Equation (2.12). Similar to the proof of Theorem 2.1.4, the main idea of the proof is to decouple the challenge key and the challenge confirmation value from the challenge message, and to simulate the decapsulation oracle without the secret key. Like in our proof of 2.2.3, decoupling is achieved via usage of $\mathsf{AOW2H}$, yielding a non-tight bound.

Since oracle queries are in superposition, we cannot simply answer decryption queries with random keys and later patch H for $m' := \mathsf{Dec}'(sk, c)$. We circumvent this difficulty by exploiting the fact that we can (information-theoretically) replace key confirmation oracle $\mathsf{H}'_{\mathsf{conf}}$ with a polynomial of sufficiently large degree. With this change, we can compute all potential preimages of d. Using the plaintext checking oracle PCO provided by the $\mathsf{OW\text{-}PCA}$ game, we recognise the correct message (if it exists) and answer decryption queries consistently without the secret key. The idea to use a key confirmation value to achieve recognisability stems from [TU16].

Proof. Let A be an adversary against the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{QKEM}^{\perp}_{m}$, issuing at most q_D (classical) queries to the decapsulation oracle $\mathrm{QD\small{ECAPS}}^{\perp}_{m}$, at most q_{H} (quantum) queries to random oracle H, and at most $q_{\mathsf{H}'_{\mathsf{conf}}}$ (quantum) queries to random oracle $\mathsf{H}'_{\mathsf{conf}}$. Consider the sequence of games given in Figure 2.24.

GAMES $G_{0,b}$. Games $G_{0,0}$ and $G_{0,1}$ describe the $\mathsf{IND\text{-}CCA}$ game in its equivalent "left-or-right" variant:

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{QKEM}^{\perp}_{m}}(\mathsf{A}) = \frac{1}{2} \cdot \left| \Pr[G^{\mathsf{A}}_{0,0} \Rightarrow 1] - \Pr[G^{\mathsf{A}}_{0,1} \Rightarrow 1] \right| \ .$$

```
GAMES $G_{0,b}$, $G_1$
01 $(pk, sk) \leftarrow \mathsf{KG}'$
02 $m^* \leftarrow_\$ \{0,1\}^n$
03 $c^* \leftarrow \mathsf{Enc}'(pk, m^*)$
04 $K_0^* := \mathsf{H}(m^*); K_1^* \leftarrow_\$ \{0,1\}^n$
05 $K^* := K_b^*$                                   //$G_{0,b}$
06 $d^* := \mathsf{H}'_{\mathsf{conf}}(m^*)$          //$G_{0,b}$
07 $(K^*, d^*) \leftarrow_\$ \{0,1\}^{n+m}$            //$G_1$
08 $b' \leftarrow \mathsf{A}^{\textsc{QDecaps}_m^\perp, |\mathsf{H}\rangle, |\mathsf{H}'_{\mathsf{conf}}\rangle}(pk, (c^*, d^*), K^*)$
09 return $b'$
```

```
$\textsc{QDecaps}_m^\perp((c, d) \neq (c^*, d^*))$
10 $m' := \mathsf{Dec}'(sk, c)$
11 if $m' \neq \perp$ and $\mathsf{H}'_{\mathsf{conf}}(m') = d$
12     return $K := \mathsf{H}(m')$
13 else return $\perp$
```

Fig. 2.24: Games $G_{0,b}$ (for bit $b \in \{0,1\}$) and G_1 for the proof of Theorem 2.2.4.

The next two steps are preparation steps to ensure that no query to $\textsc{QDecaps}_m^\perp$ can trigger a query to $\mathsf{H}'_{\mathsf{conf}}$ on m^*.

Game G_1 and games $G_{2,b}$. In game G_1, we replace $(K^* := K_b^*, d^* := \mathsf{H}'_{\mathsf{conf}}(m^*))$ with uniform random (d^*, K^*) in line 07. We have that

$$\begin{aligned}\left|\Pr[G_{0,0}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_{0,1}^{\mathsf{A}} \Rightarrow 1]\right| \leq &\left|\Pr[G_{0,0}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]\right| \\ &+ \left|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_{0,1}^{\mathsf{A}} \Rightarrow 1]\right| \quad .\end{aligned}$$

We will now upper bound each of the two terms by applying a suitable variant of Theorem 2.2.1 ($\mathsf{AOW2H}$).

In the case that $b = 1$ (i.e., in game $G_{0,1}$), K^* was already random and only d^* is changed, hence we need to apply $\mathsf{AOW2H}$ only with respect to $\mathsf{O} := \mathsf{H}'_{\mathsf{conf}}$. We identify x with m^*, y with d^*, and define algorithm GenInp_1 in Figure 2.25.

```
Algorithm $\mathsf{GenInp}_0(m^*, K^*, d^*)$
01 $(pk, sk) \leftarrow \mathsf{KG}'$
02 $c^* \leftarrow \mathsf{Enc}'(pk, m^*)$
03 $inp = (pk, c^*, d^*, K^*)$
04 return $inp$
```

```
Algorithm $\mathsf{GenInp}_1(m^*, d^*)$
05 $(pk, sk) \leftarrow \mathsf{KG}'$
06 $c^* \leftarrow \mathsf{Enc}'(pk, m^*)$
07 $K^* \leftarrow_\$ \{0,1\}^n$
08 $inp = (pk, c^*, d^*, K^*)$
09 return $inp$
```

Fig. 2.25: Input generation algorithms GenInp_0 (left) and GenInp_1 (right) for the proof of Theorem 2.2.4.

With this definition, the $\mathsf{AOW2H}$ game is identical to game $G_{0,1}$ if $y = \mathsf{O}(x) = \mathsf{H}'_{\mathsf{conf}}(m^*)$, and it is identical to game G_1 if y is random. Taking into account that A additionally has access to H (which is independent of $\mathsf{H}'_{\mathsf{conf}}$) and $\textsc{QDecaps}_m^\perp$, which triggers at most one query to $\mathsf{H}'_{\mathsf{conf}}$ per incovation, we obtain

$$|\Pr[G_{0,1}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq 2 \cdot (q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\Pr[G_{2,1}^{\mathsf{A}} \Rightarrow 1]} \ ,$$

where game $G_{2,1}$ is given in Figure 2.26, and the extractor algorithm EXT used in $G_{2,1}$ represents execution of A until a randomly chosen query to $\mathsf{H}'_{\mathsf{conf}}$, which is then measured to extract a message m' (see Figure 2.18, page 89). Note that A also has additional access to H, but H is independent of $\mathsf{H}'_{\mathsf{conf}}$ and does not affect the upper bound.

In the case that $b = 0$ (i.e., in game $G_{0,0}$), both K^* and d^* are changed, hence we need to apply $\mathsf{AOW2H}$ with respect to $\mathsf{O} := \mathsf{H} \times \mathsf{H}'_{\mathsf{conf}}$.[2] We identify x with m^*, y with (K^*, d^*), and define algorithm GenInp_0 also in Figure 2.25. With this definition, the $\mathsf{AOW2H}$ game is identical to game $G_{0,0}$ if $y = \mathsf{O}(x) = (\mathsf{H}(m^*), \mathsf{H}'_{\mathsf{conf}}(m^*))$, and it is identical to game G_1 if y is random. Note that $\text{QD}\textsc{ecaps}_m^{\perp}$ can be equivalently defined such that each invocation triggers at most one query to $\mathsf{H} \times \mathsf{H}'_{\mathsf{conf}}$. Hence

$$|\Pr[G_{0,0}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \leq 2 \cdot (q_{\mathsf{H}} + q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\Pr[G_{2,0}^{\mathsf{A}} \Rightarrow 1]} \ ,$$

where game $G_{2,0}$ is also given in Figure 2.26. (In this case, EXT represents execution of A until a randomly chosen query to H or $\mathsf{H}'_{\mathsf{conf}}$, which is then measured to extract a message m'.)

What we have shown so far is that

$$\mathrm{Adv}_{\mathsf{QKEM}_m^{\perp}}^{\mathsf{IND\text{-}CCA}}(\mathsf{A}) \leq (q_{\mathsf{H}} + q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\Pr[G_{2,0}^{\mathsf{A}} \Rightarrow 1]} + (q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\Pr[G_{2,1}^{\mathsf{A}} \Rightarrow 1]} \ .$$

```
GAMES $G_{2,b}$
$(pk, sk) \leftarrow \mathsf{KG}'$
$(m^*, K^*, d^*) \leftarrow_{\$} \{0,1\}^{2n+m}$
$c^* \leftarrow \mathsf{Enc}'(pk, m^*)$
$m' \leftarrow \mathsf{EXT}[\mathsf{A}^{\text{QD}\textsc{ecaps}_m^{\perp}, |\mathsf{H}\rangle, |\mathsf{H}'_{\mathsf{conf}}\rangle}, \mathsf{H} \times \mathsf{H}'_{\mathsf{conf}}](pk, (c^*, d^*), K^*)$    // $G_{2,0}$
$m' \leftarrow \mathsf{EXT}[\mathsf{A}^{\text{QD}\textsc{ecaps}_m^{\perp}, |\mathsf{H}\rangle, |\mathsf{H}'_{\mathsf{conf}}\rangle}, \mathsf{H}'_{\mathsf{conf}}](pk, (c^*, d^*), K^*)$    // $G_{2,1}$
return $[\![m' = m^*]\!]$
```

Fig. 2.26: Games $G_{2,b}$ for the proof of Theorem 2.2.4. Oracle $\text{QD}\textsc{ecaps}_m^{\perp}$ remains as in Figure 2.24.

G<span>AMES</span> $G_{3,b}$. In games $G_{3,b}$, oracle $\text{QD}\textsc{ecaps}_m^{\perp}$ is changed such that it does not make

[2] Recall that with oracle access to $\mathsf{O}_1 \times \mathsf{O}_2$, queries to O_1 can be answered by appending an additional O_2-register, applying $\mathbb{1} \otimes \mathbb{1} \otimes H_{n_2}$, then $U_{\mathsf{O}_1 \times \mathsf{O}_2}$, then applying $\mathbb{1} \otimes \mathbb{1} \otimes H_{n_2}$ again and returning only the first two original registers. For O_2, replace $\mathbb{1} \otimes H_{n_2}$ with $H_{n_1} \otimes \mathbb{1}$ in the description above.

use of the secret key any longer, except by testing if $\mathsf{Dec}'(sk, c) = m$ for given c and some message candidates $m \in \mathsf{Roots}(\mathsf{H}'_{\mathsf{conf}}(X) - d)$ in line 05. Recall that we can model $\mathsf{H}'_{\mathsf{conf}}$ as a random polynomial $\mathsf{H}'_{\mathsf{conf}}(X)$ of degree $2q_{\mathsf{H}'_{\mathsf{conf}}}$ over $\mathbb{F}_{2^m}$. We can hence define $\mathsf{Roots}(\mathsf{H}'_{\mathsf{conf}}(X) - d)$ as the set of polynomial roots, i.e., all messages $m \in \{0,1\}^m$ such that $\mathsf{H}'_{\mathsf{conf}}(m) = d$. In order to show that the view of A is identical in games $G_{2,b}$ and $G_{3,b}$, consider the following cases for a fixed ciphertext $(c, d) \neq (c^*, d^*)$ and $m' := \mathsf{Dec}'(sk, c)$.

- Case 1: $\textsc{QDecaps}^{\perp}_{m}$ returns $\perp$ in game $G_{2,b}$. We will now show that $\textsc{QDecaps}^{\perp}_{m}$ also returns $\perp$ in game $G_{3,b}$: If $\textsc{QDecaps}^{\perp}_{m}$ does not return $\perp$ in game $G_{3,b}$, then there exists a message $m \in \{0,1\}^m$ such that $\mathsf{H}'_{\mathsf{conf}}(m) = d$ and $m = m'$, hence $m' \neq \perp$ and $\mathsf{H}'_{\mathsf{conf}}(m') = d$, which is exactly the condition that $\textsc{QDecaps}^{\perp}_{m}$ does not return $\perp$ in Game $G_{2,b}$.
- Case 2: $\textsc{QDecaps}^{\perp}_{m}$ does not return $\perp$ in game $G_{2,b}$. We will now show that $\textsc{QDecaps}^{\perp}_{m}$ returns the same value in game $G_{3,b}$ as it returns in game $G_{2,b}$: Since $m' \neq \perp$ and $\mathsf{H}'_{\mathsf{conf}}(m') = d$, m' lies within the roots of $\mathsf{H}'_{\mathsf{conf}}(X) - d$. Since no other root m could satisfy $m = m'$, $\textsc{QDecaps}^{\perp}_{m}(c, d)$ also returns $K = \mathsf{H}(m')$ in game $G_{3,b}$.

We have just shown that in both cases (i.e., for each $b \in \{0,1\}$),

$$\Pr[G^{\mathsf{A}}_{2,b} \Rightarrow 1] = \Pr[G^{\mathsf{A}}_{3,b} \Rightarrow 1] \ .$$

```
QDecaps^⊥_m((c,d) ≠ (c*,d*))   // G_{2,b}
01 $m' := \mathsf{Dec}'(sk, c)$
02 if $m' \neq \perp$ and $\mathsf{H}'_{\mathsf{conf}}(m') = d$
03    return $K := \mathsf{H}(m')$
04 else return $\perp$
```

```
QDecaps^⊥_m((c,d) ≠ (c*,d*))   // G_{3,b}
05 if $\exists m \in \mathsf{Roots}(\mathsf{H}'_{\mathsf{conf}}(X) - d)$ s.t. $\mathsf{Dec}'(sk, c) = m$
06    return $K := \mathsf{H}(m)$.
07 else return $\perp$
```

Fig. 2.27: Oracle $\textsc{QDecaps}^{\perp}_{m}$ in games $G_{2,b}$ - $G_{3,b}$ for the proof of Theorem 2.2.4. The games' main description remains as in Figure 2.26.

It remains to upper bound $\Pr[G^{\mathsf{A}}_{3,0} \Rightarrow 1]$ and $\Pr[G^{\mathsf{A}}_{3,1} \Rightarrow 1]$. To this end, we construct $\mathsf{OW\text{-}PCA}$ adversaries B_0, B_1 against PKE' in Figure 2.28 such that B_b perfectly simulates game $G_{3,b}$ for A.

$$\Pr[G^{\mathsf{A}}_{3,b} \Rightarrow 1] = \mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}_b) \ .$$

Note that both adversaries issue at most $2q_D q_{\mathsf{H}'_{\mathsf{conf}}}$ PCO-queries: For each query (c, d) to $\textsc{QDecaps}^{\perp}_{m}$, both B_0 and B_1 compute the set $\mathsf{Roots}(\mathsf{H}'_{\mathsf{conf}}(X) - d)$ of complex roots. Since $\mathsf{H}'_{\mathsf{conf}}(X) - d$ is a polynomial of degree $2q_{\mathsf{H}'_{\mathsf{conf}}}$, the set has $2q_{\mathsf{H}'_{\mathsf{conf}}}$ elements.

In the worst case, they need to check whether $\mathrm{PCO}(m,c) = 1$ for each element $m \in \mathsf{Roots}(\mathsf{H}'_{\mathsf{conf}}(X) - d)$.

```
$\mathsf{B}_b^{\mathrm{PCO}}(pk, c^*)$
$(d^*, K^*) \leftarrow_\$ \{0,1\}^{n+m}$
$m' \leftarrow \mathsf{EXT}[\mathsf{A}^{\textsc{QDecaps}_m^{\perp}, |\mathsf{H}\rangle, |\mathsf{H}'_{\mathsf{conf}}\rangle}, \mathsf{H} \times \mathsf{H}'_{\mathsf{conf}}](pk, (c^*, d^*), K^*)$    $/\!\!/ \mathsf{B}_0$
$m' \leftarrow \mathsf{EXT}[\mathsf{A}^{\textsc{QDecaps}_m^{\perp}, |\mathsf{H}\rangle, |\mathsf{H}'_{\mathsf{conf}}\rangle}, \mathsf{H}'_{\mathsf{conf}}](pk, (c^*, d^*), K^*)$    $/\!\!/ \mathsf{B}_1$
return $m'$
```

Fig. 2.28: OW-PCA Adversaries B_b (for $b \in \{0,1\}$) for the proof of Theorem 2.2.4. Oracle $\textsc{QDecaps}_m^{\perp}$ is defined as in game $G_{3,b}$ (see Figure 2.27).

$\square$

Proof of Equation (2.13). In order to prove Equation (2.13), we will first sketch how to get rid of summand q_D if the scheme is deterministic and rigid. To this end, we introduce an intermediate game-hop between games $G_{0,b}$ and game G_1, in which we change oracle $\textsc{QDecaps}_m^{\perp}$ such that $\textsc{QDecaps}_m^{\perp}(c,d)$ always returns $\perp$ if $\mathsf{Dec}'(sk, c) = m^*$.

To verify that A's view is identical in both games, let (c,d) be any query such that $\mathsf{Dec}'(sk, c) = m^*$. We will now argue that $\textsc{QDecaps}_m^{\perp}(c,d)$ would have returned $\perp$ in games $G_{0,b}$, anyways, and hence, we only made the behaviour of $\textsc{QDecaps}_m^{\perp}$ explicit for this particular subcase: Since we assume the scheme to be rigid, we have that $c = \mathsf{Enc}'(pk, \mathsf{Dec}'(sk, c))$, and hence, $c = \mathsf{Enc}'(pk, m^*) = c^*$. A query on (c^*, d^*) is forbidden, and a query on $(c^*, d \neq d^* = \mathsf{H}'_{\mathsf{conf}}(m^*))$ would already have been answered with $\perp$ in games $G_{0,b}$.

We have just shown that the games proceed identically, but it is now verified that no query to the classically accessible oracle $\textsc{QDecaps}_m^{\perp}$ could ever trigger a query to H or $\mathsf{H}'_{\mathsf{conf}}$ on m^*. We can therefore apply Theorem 2.2.1, but with the improved bound given in Equation (2.7) (see page 89), to get rid of the summand q_D. Since $G_{0,b}$ and game G_1 behave identically, and our simulation of $\textsc{QDecaps}_m^{\perp}$ in games $G_{3,b}$ was perfect, adversaries B_b can remain unchanged and the summand q_D can be dropped.

Second, we now show how to change our adversaries such that no query to $\textsc{QDecaps}_m^{\perp}$ will ever trigger them to issue a query to PCO on $(m^*, -)$:

We introduce another intermediate game-hop between games $G_{0,b}$ and game $G_{1,b}$, in which we change $\textsc{QDecaps}_m^{\perp}$ such that it always returns $\perp$ if $d = d^*$. In order to recognise this change, A has to query $\textsc{QDecaps}_m^{\perp}$ on a ciphertext $(c \neq c^*, d^*)$ such that $\textsc{QDecaps}_m^{\perp}(c, d^*)$ would not have rejected in games $G_{0,b}$. As shown above, no ciphertext $c \neq c^*$ could decrypt to m^*, since we assume the scheme to be rigid. Since queries on (c^*, d^*) are forbidden, the games can hence only differ if there exists a query to $\text{QD}\textsc{ecaps}_m^{\perp}$ on a ciphertext $(c \neq c^*, d^*)$ such that $m' := \mathsf{Dec}'(sk, c) \neq m$ and $\mathsf{H}'_{\mathsf{conf}}(m') = d^*$. Since the range of $\mathsf{H}'_{\mathsf{conf}}$ is $\mathcal{M}$, this happens with probability at most $q_D/|\mathcal{M}| = q_D/2^m$.

Clearly, we can turn each adversary B_b into an adversary B'_b that rejects during its simulation of $\text{QD}\textsc{ecaps}_m^{\perp}$ whenever queried on a ciphertext of the form (c, d^*).

Security of $\mathsf{QKEM}_m^{\not\perp}$. The following theorem establishes that $\mathsf{IND\text{-}CCA}$ security of $\mathsf{QKEM}_m^{\not\perp}$ reduces to $\mathsf{OW\text{-}PCA}$ security of PKE', in the quantum random oracle model.

Theorem 2.2.5 (PKE' $\mathsf{OW\text{-}PCA}$ $\overset{\text{QROM}}{\Rightarrow}$ $\mathsf{QKEM}_m^{\not\perp}$ $\mathsf{IND\text{-}CCA}$)**.** For any $\mathsf{IND\text{-}CCA}$ quantum adversary A issuing at most q_D (classical) queries to the decapsulation oracle $\text{QD}\textsc{ecaps}_m^{\not\perp}$, at most q_{H} (quantum) queries to random oracle H, at most $q_{\mathsf{H}'_{\mathsf{conf}}}$ (quantum) queries to random oracle $\mathsf{H}'_{\mathsf{conf}}$, (and arbitrarily many (quantum) queries to random oracle $\mathsf{H}''_{\mathsf{reject}}$,= there exist $\mathsf{OW\text{-}PCA}$ adversaries B_0 and B_1, issuing at most $2q_D q_{\mathsf{H}'_{\mathsf{conf}}}$ queries to oracle PCO, such that

$$\begin{aligned}\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{QKEM}_m^{\not\perp}}(\mathsf{A}) \leq & (q_{\mathsf{H}} + q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}_0)} \\ & + (q_{\mathsf{H}'_{\mathsf{conf}}} + q_D) \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}_1)} \ ,\end{aligned}$$

and the running time of B_0 and B_1 is about that of A. If PKE' furthermore is deterministic and rigid, there exist $\mathsf{OW\text{-}PCA}$ adversaries B'_0 and B'_1 such that

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{QKEM}_m^{\not\perp}}(\mathsf{A}) \leq (q_{\mathsf{H}} + q_{\mathsf{H}'_{\mathsf{conf}}}) \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}'_0)} + q_{\mathsf{H}'_{\mathsf{conf}}} \cdot \sqrt{\mathrm{Adv}^{\mathsf{OW\text{-}PCA}}_{\mathsf{PKE}'}(\mathsf{B}'_1)} + \frac{q_D}{2^{m-1}} \ ,$$

the running time of B'_0 and B'_1 is about that of A, and no query to $\text{QD}\textsc{ecaps}_m^{\perp}$ will ever trigger either adversary B'_{b} to issue a query to PCO on $(m^*, -)$.

The proof is almost the same as the one of Theorem 2.2.4: The crucial observation is that in all games used in the proof of Theorem 2.2.4, $\text{QD}\textsc{ecaps}_m^{\perp}$ always knows if a given ciphertext (c, d) is valid or not. (Recall that we achieve validity recognition even without the secret key, by computing all possible preimages of d and using plaintext checking oracle PCO.) If a ciphertext is not valid, our simulation of $\text{QD}\textsc{ecaps}_m^{\perp}$ (correctly) returns $\perp$. In order to prove Theorem 2.2.5, one can hence simply replace $\perp$ with $\mathsf{H}''_{\mathsf{reject}}(c, d)$. Since $\mathsf{H}''_{\mathsf{reject}}$ is an independent random oracle, this change does not trigger any additional queries to either H or $\mathsf{H}'_{\mathsf{conf}}$.

2.2.4 The Resulting KEMs

For completeness, we combine transformation T with $\{\mathsf{QU}_m^{\perp}, \mathsf{QU}_m^{\not\perp}\}$ from the previous sections to obtain two post-quantum secure variants $\mathsf{QFO}_m^{\perp} := \mathsf{QU}_m^{\perp} \circ \mathsf{T}$ and $\mathsf{QFO}_m^{\not\perp} := \mathsf{QU}_m^{\not\perp} \circ \mathsf{T}$ of the FO transformation.

To a public-key encryption scheme $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M} = \{0,1\}^m$ and randomness space $\mathcal{R}$, and hash functions $\mathsf{G} : \mathcal{M} \to \mathcal{R}$, $\mathsf{H} : \{0,1\}^* \to \{0,1\}^n$, $\mathsf{H}'_{\mathsf{conf}} : \mathcal{M} \to \mathcal{M}$, and $\mathsf{H}''_{\mathsf{reject}} : \{0,1\}^* \to \{0,1\}^n$, we associate

$$\begin{aligned}\mathsf{QKEM}_m^{\perp} &:= \mathsf{QFO}_m^{\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}] := \mathsf{QU}_m^{\perp}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}] \\ &= (\mathsf{KG}, \mathsf{QEncaps}_m, \mathsf{QDecaps}_m^{\perp})\end{aligned}$$

and

$$\begin{aligned}\mathsf{QKEM}_m^{\not\perp} &:= \mathsf{QFO}_m^{\not\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}, \mathsf{H}''_{\mathsf{reject}}] := \mathsf{QU}_m^{\not\perp}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}, \mathsf{H}'_{\mathsf{conf}}, \mathsf{H}''_{\mathsf{reject}}] \\ &= (\mathsf{KG}, \mathsf{QEncaps}_m, \mathsf{QDecaps}_m^{\not\perp}) \ .\end{aligned}$$

Their constituting algorithms are given in Figure 2.29.

```
$\mathsf{QEncaps}_m(pk)$
$m \leftarrow_\$ \mathcal{M}$
$c := \mathsf{Enc}(pk, m; \mathsf{G}(m))$
$K := \mathsf{H}(m)$
$d := \mathsf{H}'_{\mathsf{conf}}(m)$
return $(K, c, d)$

$\boxed{\mathsf{QDecaps}_m^{\perp}(sk, c, d)}$   $\mathsf{QDecaps}_m^{\not\perp}(sk, c, d)$
$m' := \mathsf{Dec}(sk, c)$
if $m' = \perp$ or $c \neq \mathsf{Enc}(pk, m'; \mathsf{G}(m'))$ or $\mathsf{H}'_{\mathsf{conf}}(m') \neq d$
    $\boxed{\text{return } \perp}$
    return $K := \mathsf{H}''_{\mathsf{reject}}(c, d)$
else return $K := \mathsf{H}(m')$
```

Fig. 2.29: Key encapsulation mechanisms $\mathsf{QKEM}_m^{\perp} = (\mathsf{KG}, \mathsf{QEncaps}_m, \mathsf{QDecaps}_m^{\perp})$ and $\mathsf{QKEM}_m^{\not\perp}(\mathsf{KG}, \mathsf{QEncaps}_m, \mathsf{QDecaps}_m^{\not\perp})$ obtained from $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$.

We will now show how to concretely bound the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM} \in \{\mathsf{QKEM}_m^{\perp}, \mathsf{QKEM}_m^{\not\perp}\}$, in the quantum random oracle model. Like in Section 2.1.4, we will denote by $\mathsf{OW}_t(\mathsf{PKE})$, $\mathsf{OW\text{-}PCA}_t(\mathsf{PKE}')$ and $\mathsf{CCA}_t(\mathsf{KEM})$ the upper bound on the respective advantage of all adversaries running in time at most t.

As corollaries to theorems 2.2.4 and 2.2.5 (and since $\mathsf{T}[\mathsf{PKE}, \mathsf{G}]$ is rigid), we can upper bound both $\mathsf{IND\text{-}CCA}_t(\mathsf{QKEM}_m^{\perp})$ and $\mathsf{IND\text{-}CCA}_t(\mathsf{QKEM}_m^{\not\perp})$ by

$$\left.\begin{array}{l}\mathsf{IND\text{-}CCA}_t(\mathsf{QKEM}_m^{\perp}) \\ \mathsf{IND\text{-}CCA}_t(\mathsf{QKEM}_m^{\not\perp})\end{array}\right\} \leq \ (q_{\mathsf{H}} + 2 \cdot q_{\mathsf{H}'_{\mathsf{conf}}}) \cdot \sqrt{\mathsf{OW\text{-}PCA}_t(\mathsf{PKE}')} + \frac{q_D}{2^{m-1}} \ ,$$

and we know that no $\mathsf{IND\text{-}CCA}$ adversary will ever trigger a resulting $\mathsf{OW\text{-}PCA}$ adversary to issue a query to PCO on $(m^*, -)$.

As a corollary to Theorem 2.2.3, and only considering adversaries that do not query $\mathrm{PCO}(m^*, -)$, we can upper bound $\mathsf{OW\text{-}PCA}_t(\mathsf{PKE}')$ by

$$\mathsf{OW\text{-}PCA}_t(\mathsf{PKE}') \leq (1 + 2q_{\mathsf{G}}) \cdot \sqrt{\mathsf{OW}_t(\mathsf{PKE})} + 8 \cdot (q_{\mathsf{G}} + q_{\mathrm{PCO}} + 1)^2 \cdot \delta_{\mathrm{wc}} \ .$$

The following table combines the bounds above to give (simplified) concrete bounds of the $\mathsf{IND\text{-}CCA}$ security of $\mathsf{KEM} \in \{\mathsf{QKEM}_m^{\not\perp}, \mathsf{QKEM}_m^{\perp}\}$. Here q_D denotes the number of decryption queries, $q_{\mathsf{RO}} := q_{\mathsf{G}} + q_{\mathsf{H}} + q_{\mathsf{H}'_{\mathsf{conf}}} + q_{\mathsf{H}''_{\mathsf{reject}}}$ counts the total number of (explicit) quantum queries to the respective random oracles, and we used that the oracle queries in the $\mathsf{IND\text{-}CCA}$ game translate to $2q_D q_{\mathsf{H}'_{\mathsf{conf}}}$ many queries to PCO.

KEM	Concrete bound on $\mathsf{IND\text{-}CCA}_t(\mathsf{KEM}) \leq$
$\mathsf{QKEM}_m^{\not\perp}$, $\mathsf{QKEM}_m^{\perp}$	$4q_{\mathsf{RO}}^{\frac{3}{2}} \cdot \sqrt[4]{\mathsf{OW}_t(\mathsf{PKE})} + 12q_{\mathsf{RO}}^2 \cdot q_D \cdot \sqrt{\delta} + \frac{q_D}{2^{m-1}}$

2.3 Tighter Security Bounds in the QROM

Recall that the QU_m-variants from Section 2.2 require a non-standard security notion ($\mathsf{OW\text{-}PCA}$). We view it as desirable to start from a standard notion that can be verified easily. The natural approach would be to apply transformation T before applying a QU_m-variant, as T is proven to turn OW into $\mathsf{OW\text{-}PCA}$ security. This modular approach, however, comes with a significant drawback: The security proofs given in Section 2.2 are highly non-tight, as all of them invoke a non-tight quantum query extraction argument. In particular, combining the security statement for transformation T with the security statement for either one of the QU_m-transformations leads to a quartic loss in the OW advantage, and a factor of $q^{\frac{3}{2}}$.

Furthermore, the QU_m-variants from Section 2.2 introduce some communication overhead by including a key confirmation ciphertext $d := \mathsf{H}'_{\mathsf{conf}}(m)$ of the same length as the message itself. Most real-world proposals are designed such that they do not use key confirmation, meaning that they fit the framework of $\mathsf{FO} = \mathsf{U} \circ \mathsf{T}$ from Section 2.1.4 for some U-variant (see page 84), rather than the framework of $\mathsf{QFO} = \mathsf{QU}_m \circ \mathsf{T}$.

Prior to this result, a modular proof for a variant of $\mathsf{FO}_m^{\not\perp}$ in the quantum random oracle model was already given in [SXY18]: In [SXY18], $\mathsf{FO}_m^{\not\perp}$ is dissected into two transformations TPunc and SXY. Transformation TPunc differs from transformation T in two aspects: First, it does not execute the re-encryption check during decryption. This check is instead shifted to the second transformation SXY, which, apart from doing the re-encryption check, is our transformation $\mathsf{U}^{\not\perp}_{m}$ from Section 2.1.3 (see page 80).[3] Later, we will also briefly discuss other U-variants with an added re-encryption check. To keep our notation concise, we will write $\mathsf{U}^{\circlearrowleft}$ for a U-variant to which the re-encryption check was added. With this notation, we can identify SXY with $\mathsf{U}^{\not\perp\circlearrowleft}_{m}$, which is the notation we will use from now on. Second, transformation TPunc removes a single message $\hat{m}$ from the message space, for reasons we explain below.

In the quantum random oracle model, $\mathsf{IND}\text{-}\mathsf{CCA}$ security of $\mathsf{U}^{\not\perp}_{m}$ with re-encryption (i.e., the security of $\mathsf{U}^{\not\perp\circlearrowleft}_{m}$) tightly reduces to disjoint simulatability of ciphertexts (DS, see Definition 1.1.8 on page 28). DS is naturally satisfied by many code- and lattice-based encryption schemes. If a scheme is $\mathsf{IND}\text{-}\mathsf{CPA}$ secure, simulatability can also be achieved generically by removing any message $\hat{m}$ from the message space, and using $\hat{m}$ to sample fake encryptions. This method is also called "puncturing".

Note, however, that $\mathsf{U}^{\not\perp\circlearrowleft}_{m}$ can only be applied to deterministic schemes (since it re-encrypts). A *deterministic* scheme that satisfies simulatability is achieved by using transformation TPunc, albeit non-tightly. More importantly, the result for $\mathsf{U}^{\not\perp\circlearrowleft}_{m}$ that was given in [SXY18] only considered schemes which are perfectly correct, rendering it inapplicable for many important constructions. It would therefore be desirable to generalise the result such that it also holds for non-perfectly correct schemes, but this generalisation turned out to be less than straightforward: For deterministic schemes, worst-case correctness effectively requires that the scheme is perfectly correct for almost all public keys. Since it is not clear how to give a correctness definition for deterministic encryption schemes that suits known tight proof strategies for $\mathsf{U}^{\not\perp\circlearrowleft}_{m}$, while also being achievable by most lattice-based schemes, we circumvent this difficulty by resorting to a non-modularised proof, i.e., we only consider transformation $\mathsf{FO}^{\not\perp}_{m} = \mathsf{U}^{\not\perp}_{m} \circ \mathsf{T}$: By plugging in $\mathsf{T}[-, \mathsf{G}]$ into $\mathsf{U}^{\not\perp}_{m}$, we can modify random oracle G during the security proof such that the scheme is rendered perfectly correct for a few game-hops. With this trick, the $\mathsf{U}^{\not\perp}_{m}$-portion of the combined proof remains tight.

Our transformation $\mathsf{FO}^{\not\perp}_{m}$ can be applied to any PKE scheme that is both $\mathsf{IND}\text{-}\mathsf{CPA}$ and DS secure. Our reduction is tighter than the one that results from combining those for TPunc and $\mathsf{U}^{\not\perp\circlearrowleft}_{m}$ that were given in [SXY18].

Furthermore, we achieve a better bound with respect to the scheme's correctness than previously known due to a better bound for the generic distinguishing problem. In the case that PKE is not already DS, this requirement can be waived with negligible loss of efficiency: To rely on $\mathsf{IND}\text{-}\mathsf{CPA}$ alone, all that has to be done is to puncture the message space, which we formalise by giving a transformation Punc. A visualisation is given in Figure 2.30.

[3] This means that the KEM now does not only reject if decryption of a ciphertext fails, but also if re-encryption does not yield the ciphertext.

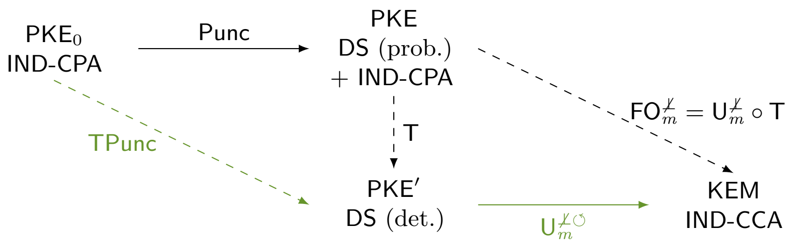

PKE$_0$
IND-CPA
Punc
PKE
DS (prob.)
+ IND-CPA
T
TPunc
$\mathsf{FO}^{\not\perp}_m = \mathsf{U}^{\not\perp}_m \circ \mathsf{T}$
PKE′
DS (det.)
$\mathsf{U}^{\not\perp\circlearrowleft}_m$
KEM
IND-CCA

Fig. 2.30: Comparison of the modular transformation given in [SXY18] (green) with ours. Solid arrows indicate tight reductions, dashed arrows indicate non-tight reductions.

Security of other FO variants. It was shown [BHH+19, Thm. 5] that for any deterministic scheme PKE, $\mathsf{IND\text{-}CCA}$ security of $\mathsf{U}^{\not\perp\circlearrowleft}_m[\mathsf{PKE}', \mathsf{H}]$ is equivalent to $\mathsf{IND\text{-}CCA}$ security of $\mathsf{U}^{\not\perp\circlearrowleft}_{m,c}[\mathsf{PKE}', \mathsf{H}]$. Since T indeed renders the intermediate scheme deterministic, we obtain as a straightforward corollary that $\mathsf{FO}^{\not\perp}_{m,c}[\mathsf{PKE}', \mathsf{GH}]$ is as secure as $\mathsf{FO}^{\not\perp}_m[\mathsf{PKE}', \mathsf{GH}]$. For the variants with explicit reject, however, it is unclear how to deploy a proof strategy similar to ours without either reintroducing key confirmation, or requiring a validity oracle.

Concrete Applications. Our transformation can be applied to any scheme that is $\mathsf{IND\text{-}CPA}$ secure with post-quantum security, e.g., Frodo [NAB+17], Kyber [BDK+17], and Lizard [BI17]. Recall that the additional requirement of DS can be achieved with negligible loss of efficiency. However, in many applications even this negligible loss is inexistent since most of the aforementioned schemes can already be proven DS under the same assumption that their $\mathsf{IND\text{-}CPA}$ security is based upon.

Organisation of Section 2.3

In Section 2.3.1, we show that T achieves deterministic DS from DS and $\mathsf{IND\text{-}CPA}$. Next, in Section 2.3.2, we show that the combined transformation $\mathsf{FO}^{\not\perp}_m = \mathsf{U}^{\not\perp}_m \circ \mathsf{T}$ turns any encryption scheme that is both DS and $\mathsf{IND\text{-}CPA}$ secure into a KEM that is $\mathsf{IND\text{-}CCA}$ secure. The proof is applicable to non-perfectly correct schemes.

We believe that many lattice-based schemes fulfill DS in a natural way[4], but for the sake of completeness, we will show in Section 2.3.3 how transformation Punc can be used to waive the requirement of DS with negligible loss of efficiency.

[4] Fake encryptions could be sampled uniformly random. DS would follow from the LWE assumption, and since LWE samples are relatively sparse, uniform sampling should be disjoint.

2.3.1 Transformation T*: From* DS *and* $\mathsf{IND\text{-}CPA}$ *to deterministic* DS

Recall that T from Section 2.2.2 turns any probabilistic public-key encryption scheme into a deterministic one. In this section, we will now show that T turns any scheme that is both DS and $\mathsf{IND\text{-}CPA}$ secure into a deterministic scheme that is DS. Our security proof for T is tighter than the known proof for TPunc (see [SXY18, Theorem 3.3]) due to our use of the semi-classical O2H theorem.

The Construction. Take an encryption scheme $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ and randomness space $\mathcal{R}$. Assume PKE to be additionally endowed with a sampling algorithm $\mathsf{fakeEnc}$ that takes as input a public key and returns a fake ciphertext. To PKE and random oracle $\mathsf{G} : \mathcal{M} \to \mathcal{R}$, we associate $\mathsf{PKE}' = \mathsf{T}[\mathsf{PKE}, \mathsf{G}]$, where the algorithms of PKE' are defined in Figure 2.2 (see page 62), and we add to the description of PKE' the sampling algorithm $\mathsf{fakeEnc}$.

The following lemma states that combined $\mathsf{IND\text{-}CPA}$ and DS security of PKE imply the DS security of PKE'.

Lemma 2.3.1 (DS security of PKE')**.** If PKE is ϵ-disjoint, so is PKE'. For all adversaries A issuing at most q_{G} (quantum) queries to G, there exist an adversary $\mathsf{B}_{\mathsf{IND}}$ and an adversary B_{DS} such that

$$\begin{aligned}\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE}'}(\mathsf{A}) &\leq \mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE}}(\mathsf{B}_{\mathsf{DS}}) + 2 \cdot \sqrt{2q_{\mathsf{G}} \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B}_{\mathsf{IND}}) + \frac{4q_{\mathsf{G}}^2}{|\mathcal{M}|}} \\ &\leq \mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE}}(\mathsf{B}_{\mathsf{DS}}) + 2 \cdot \sqrt{2q_{\mathsf{G}} \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B}_{\mathsf{IND}})} + \frac{4q_{\mathsf{G}}}{\sqrt{|\mathcal{M}|}} \quad ,\end{aligned}$$

and the running time of each adversary is about that of A.

Proof. It is straightforward to prove disjointness since $\mathsf{Enc}'(pk, \mathcal{M})$ is a subset of $\mathsf{Enc}(pk, \mathcal{M}; \mathcal{R})$.

Let A be a DS adversary against PKE'. Consider the sequence of games given in Figure 2.31. Per definition,

$$\begin{aligned}\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE}'}(\mathsf{A}) &= |\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_1^{\mathsf{A}} \Rightarrow 1]| \\ &\leq |\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| + |\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \quad .\end{aligned}$$

To upper bound $|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]|$, consider adversary B_{DS} against the disjoint simulatability of the underlying scheme PKE, given in Figure 2.32. B_{DS} runs in the time that is required to run A and to simulate G for q_{G} queries. Since B_{DS} perfectly

```
Games $G_0$-$G_3$
01 $pk \leftarrow \mathsf{KG}$
02 $m^* \leftarrow_\$ \mathcal{M}$
03 $c^* \leftarrow \mathsf{fakeEnc}(pk)$                  // $G_0$
04 $c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$     // $G_1$
05 $c^* \leftarrow \mathsf{Enc}(pk, m^*)$                 // $G_2$
06 $b' \leftarrow \mathsf{A}^{|\mathsf{G}\rangle}(pk, c^*)$
07 return $b'$
```

Fig. 2.31: Games G_0 - G_2 for the proof of Lemma 2.3.1.

simulates game G_0 if run with a fake ciphertext as input, and game G_2 if run with an encryption $c \leftarrow \mathsf{Enc}(pk, m^*)$ of a random message m^*,

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| = \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{DS}}(\mathsf{B}_{\mathsf{DS}}) \ .$$

It remains to upper bound $|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]|$. We claim that there exists an adversary $\mathsf{B}_{\mathsf{IND}}$ such that

$$|\Pr[G_1^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \leq 2 \cdot \sqrt{2q_{\mathsf{G}} \cdot \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{IND\text{-}CPA}}(\mathsf{B}_{\mathsf{IND}}) + \frac{4q_{\mathsf{G}}^2}{|\mathcal{M}|}} \ .$$

To prove this claim, we will first introduce an intermediate game $G_{1.5}$ in Figure 2.33.

```
$\mathsf{B}_{\mathsf{DS}}(pk, c)$
01 $b' \leftarrow \mathsf{A}^{|\mathsf{G}\rangle}(pk, c)$
02 return $b'$

$\mathsf{B}_{\mathsf{IND},1}(pk)$
03 $m^* \leftarrow_\$ \mathcal{M}$
04 return $(0, m^*, \mathrm{st} := m^*)$

$\mathsf{B}_{\mathsf{IND},2}(pk, c^*, \mathrm{st} := m^*)$
05 FIND := 0
06 $b' \leftarrow \mathsf{A}^{|\mathsf{G} \setminus \{\mathsf{m}^*\}\rangle}(pk, c^*)$
07 return FIND

$\mathsf{G} \setminus \{\mathsf{m}^*\} \, |\psi\rangle$
08 $|\phi, b\rangle := \mathsf{O}^{\mathsf{SC}}_{\{\mathsf{m}^*\}} \, |\psi, 0\rangle$
09 if $b = 1$
10     FIND := 1
11 return $U_{\mathsf{G}} \, |\phi\rangle$
```

Fig. 2.32: Adversaries B_{DS} and $\mathsf{B}_{\mathsf{IND}}$ for the proof of Lemma 2.3.1.

Game $G_{1.5}$. In game $G_{1.5}$, we replace oracle access to G with oracle acess to G' in line 08, where G' is defined as follows: we pick a uniformly random r^* in line 04 and let $\mathsf{G}'(m) := \mathsf{G}(m)$ for all $m \neq m^*$, and $\mathsf{G}'(m^*) := r^*$. Note that this change also affects the challenge ciphertext c^* since it is now defined relative to this new r^*, i.e., we now have $c^* = \mathsf{Enc}(pk, m^*; \mathsf{G}'(m^*))$. Since r^* is uniformly random and G is a random oracle, so is G', and since we kept c^* consistent, this change is purely conceptual and

$$\Pr[G_1^{\mathsf{A}} \Rightarrow 1] = \Pr[G_{1.5}^{\mathsf{A}} \Rightarrow 1] \ .$$

```
Games $G_1$-$G_2$
$pk \leftarrow \mathsf{KG}$
$m^* \leftarrow_\$ \mathcal{M}$
$r^* := \mathsf{G}(m^*)$                        $/\!\!/ G_1$
$r^* \leftarrow_\$ \mathcal{R}$                         $/\!\!/ G_{1.5}$-$G_2$
$\mathsf{G}' := \mathsf{G}_1^{m^* \mapsto r^*}$                  $/\!\!/ G_{1.5}$
$c^* := \mathsf{Enc}(pk, m^*; r^*)$
$b' \leftarrow \mathsf{A}^{|\mathsf{G}\rangle}(pk, c^*)$              $/\!\!/ G_1$, $G_2$
$b' \leftarrow \mathsf{A}^{|\mathsf{G}'\rangle}(pk, c^*)$             $/\!\!/ G_{1.5}$
return $b'$
```

Fig. 2.33: Games G_1 - G_2 and intermediate game $G_{1.5}$ for the proof of Lemma 2.3.1.

Game G_2. In game G_2, we switch back to oracle access to G, but without changing c^*, meaning that we have now decoupled the ciphertext from $\mathsf{G}(m^*)$.

To upper bound $|\Pr[G_{1.5}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]|$, we will use one-way to hiding with semi-classical oracles (see Theorem 1.3.3, page 44). Intuitively, the first part of O2H states that if oracles G and G' only differ on point m^*, the probability of an adversary being able to tell G and G' apart is directly related to m^* being detectable in its random oracle queries. Detecting m^* is formalised by game G_3 in Figure 2.34, in which the input register of each random oracle query is measured with respect to projector $|m^*\rangle\langle m^*|$, thereby collapsing the query to a superposition such that the input register only contains m^* (and flag FIND is switched to **true**) or that it does not contain m^* at all. We denote this process by a call to oracle $\mathsf{O}^{\mathsf{SC}}_{\{\mathsf{m}^*\}}$. Identifying O_1 with G, O_2 with G', S with $\{m^*\}$ and z with $(pk, c^* := \mathsf{Enc}(pk, m^*; r^*))$, we can now apply Equation (1.2) of Theorem 1.3.3 to obtain

$$|\Pr[G_{1.5}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_2^{\mathsf{A}} \Rightarrow 1]| \leq 2 \cdot \sqrt{q_{\mathsf{G}} \cdot \Pr[G_3^{\mathsf{A}} \Rightarrow 1]} \ .$$

```
Game $G_3$-$G_4$
FIND := 0
$pk \leftarrow \mathsf{KG}$
$m^* \leftarrow_\$ \mathcal{M}$
$c^* \leftarrow \mathsf{Enc}(pk, m^*)$           $/\!\!/ G_3$
$c^* \leftarrow \mathsf{Enc}(pk, 0)$            $/\!\!/ G_4$
$b' \leftarrow \mathsf{A}^{|\mathsf{G} \setminus \{\mathsf{m}^*\}\rangle}(pk, c^*)$
return FIND

$\mathsf{G} \setminus \{\mathsf{m}^*\} |\psi\rangle$
$|\phi, b\rangle := \mathsf{O}^{\mathsf{SC}}_{\{\mathsf{m}^*\}} |\psi, 0\rangle$
if $b = 1$
    FIND := 1
return $U_{\mathsf{G}} |\phi\rangle$
```

Fig. 2.34: Games G_3 - G_4 for the proof of Lemma 2.3.1.

Game G_4. In game G_4, $c^* \leftarrow \mathsf{Enc}(pk, m^*)$ is replaced with an encryption of 0. Since in game G_4, (pk, c^*) is independent of m^*, we can apply Equation (1.5) of Theorem 1.3.4

that upper bounds the probability of finding an independent point m^*, relative to the number of queries and the size of the search space $\mathcal{M}$: We obtain

$$\Pr[G_4^{\mathsf{A}} \Rightarrow 1] \leq \frac{4q_{\mathsf{G}}}{|\mathcal{M}|} \ .$$

To finally upper bound $|\Pr[G_3^{\mathsf{A}} \Rightarrow 1] - \Pr[G_4^{\mathsf{A}} \Rightarrow 1]|$, consider adversary $\mathsf{B}_{\mathsf{IND}}$ against the $\mathsf{IND\text{-}CPA}$ security of PKE, given in Figure 2.32. $\mathsf{B}_{\mathsf{IND}}$ runs in the time that is required to run A, and to simulate oracle G (and perform a measurement) for q_{G} many queries. Since $\mathsf{B}_{\mathsf{IND}}$ perfectly simulates game G_3 if run with an encryption of m^*, and game G_4 if run with an encryption of 0,

$$|\Pr[G_3^{\mathsf{A}} \Rightarrow 1] - \Pr[G_4^{\mathsf{A}} \Rightarrow 1]| = 2 \cdot \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{IND\text{-}CPA}}(\mathsf{B}_{\mathsf{IND}}) \ .$$

Collecting the probabilities yields

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] \leq 2 \cdot \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{IND\text{-}CPA}}(\mathsf{B}_{\mathsf{IND}}) + \frac{4q_{\mathsf{G}}}{|\mathcal{M}|} \ .$$

□

2.3.2 *Transformation* $\mathsf{FO}_m^{\not\perp}$*: From* DS *and* $\mathsf{IND\text{-}CPA}$ *to* $\mathsf{IND\text{-}CCA}$

We will now show that the combined transformation $\mathsf{FO}_m^{\not\perp} = \mathsf{U}_m^{\not\perp} \circ \mathsf{T}$ from turns any encryption scheme that is both DS and $\mathsf{IND\text{-}CPA}$ secure into a KEM that is $\mathsf{IND\text{-}CCA}$ secure.

The Construction. To $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ and randomness space $\mathcal{R}$, and random oracles $\mathsf{H} : \mathcal{M} \to \mathcal{K}$, $\mathsf{G} : \mathcal{M} \to \mathcal{R}$, we associate $\mathsf{KEM} = \mathsf{FO}_m^{\not\perp}[\mathsf{PKE}, \mathsf{G}, \mathsf{H}] := \mathsf{U}_m^{\not\perp}[\mathsf{T}[\mathsf{PKE}, \mathsf{G}], \mathsf{H}]$, where the algorithms of KEM are given in Figure 2.17 (see page 85). We slightly generalise the construction: Instead of sampling the rejection seed s from $\mathcal{M}$, we now sample it from $\{0,1\}^\ell$ for some integer ℓ.

Before we proceed to the security statement for KEM, we recall helper lemma [SXY18, Lem. 2.2] that will allow us to replace access to $\mathsf{H}(s, \cdot)$ with access to a new, independent random oracle H'.

Lemma 2.3.2. Let ℓ be an integer, and let $\mathsf{H} : \{0,1\}^\ell \times X \to Y$ and $\mathsf{H}' : X \to Y$ be two independent random oracles. For any (possibly unbounded) quantum adversary A, issuing at most q_{H} many (quantum) queries to H, we have that

$$|\Pr_{s\leftarrow_\$\{0,1\}^\ell}[1 \leftarrow \mathsf{A}^{|\mathsf{H}\rangle,|\mathsf{H}(s,\cdot)\rangle}] - \Pr[1 \leftarrow \mathsf{A}^{|\mathsf{H}\rangle,|\mathsf{H}'\rangle}]| \leq q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}} \ .$$

Security of KEM. The following theorem establishes that $\mathsf{IND\text{-}CCA}$ security of KEM reduces to DS and $\mathsf{IND\text{-}CPA}$ security of PKE, in the quantum random oracle model. Its proof is quite similar to the proof that one could obtain by combining the modular proofs in [SXY18], except for the fact that it is able to handle correctness errors.

Theorem 2.3.3 (PKE DS + $\mathsf{IND\text{-}CPA}$ $\overset{\text{QROM}}{\Rightarrow}$ KEM $\mathsf{IND\text{-}CCA}$)**.** Assume PKE to be δ-worst-case correct, and furthermore assume that PKE comes with a sampling algorithm $\mathsf{fakeEnc}$ such that PKE is ϵ_{dis}-disjoint. Then, for any (quantum) $\mathsf{IND\text{-}CCA}$ adversary A issuing at most q_D (classical) queries to the decapsulation oracle $\text{Dec}_m^{\not\perp}$, at most q_{H} quantum queries to H, and at most q_{G} quantum queries to G, there exist (quantum) adversaries B_{DS} and $\mathsf{B}_{\mathsf{IND}}$ such that

$$\begin{aligned}\text{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}(\mathsf{A}) \leq \text{Adv}^{\mathsf{DS}}_{\mathsf{PKE}}(\mathsf{B}_{\mathsf{DS}}) + 2 \cdot \sqrt{2q_{\mathsf{RO}} \cdot \text{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B}_{\mathsf{IND}})} \\ + 16 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \delta + \frac{4q_{\mathsf{RO}}}{\sqrt{|\mathcal{M}|}} + \epsilon_{\text{dis}} + q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}} \quad ,\end{aligned}$$

where q_{RO} counts the total number of random oracle queries, and the running time of B_{DS} and $\mathsf{B}_{\mathsf{IND}}$ is about that of A.

Proof. Let A be an adversary against the $\mathsf{IND\text{-}CCA}$ security of KEM, issuing at most q_D queries to $\text{Dec}_m^{\not\perp}$, at most q_{H} queries to the quantum random oracle H, and at most q_{G} queries to the quantum random oracle G. Consider the sequence of games given in Figure 2.35.

Game G_0. Since game G_0 is the original $\mathsf{IND\text{-}CCA}$ game,

$$\text{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}(\mathsf{A}) = |\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - 1/2| \ .$$

Game G_1. In game G_1, we change the way how oracle $\text{Dec}_m^{\not\perp}$ rejects implicitly: Whenever decryption or re-encryption fails, the oracle now returns $K := \mathsf{H}'(c)$, see line 17, where H' is an independent random oracle. Clearly, this game-hop can be simulated by a distinguisher with oracle access to H and $\mathsf{O} \in \{\mathsf{H}(s,\cdot), \mathsf{H}'\}$, by using its oracle O to reject implicitly. Applying Lemma 2.3.2, we obtain

$$|\Pr[G_0^{\mathsf{A}} = 1] - \Pr[G_1^{\mathsf{A}} = 1]| \leq q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}} \ .$$

```
GAMES $G_0$ - $G_4$
01 $b \leftarrow_\$ \{0,1\}$
02 $(pk, sk) \leftarrow \mathsf{KG}$
03 $\mathsf{H_q} \leftarrow_\$ \mathcal{K}^{\mathcal{C}}$                          $/\!\!/ G_2$ - $G_4$
04 $\mathsf{H} := \mathsf{H_q}(\mathsf{Enc}(pk, -; \mathsf{G}(-)))$        $/\!\!/ G_2$ - $G_4$
05 $m^* \leftarrow \mathcal{M}$
06 $c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$           $/\!\!/ G_0$ - $G_2$
07 $c^* \leftarrow \mathsf{fakeEnc}(pk)$                    $/\!\!/ G_3$ - $G_4$
08 $K_0^* := \mathsf{H}(m^*)$                            $/\!\!/ G_0$ - $G_1$
09 $K_0^* := \mathsf{H_q}(c^*)$                           $/\!\!/ G_2$ - $G_3$
10 $K_0^* \leftarrow_\$ \mathcal{K}$                                 $/\!\!/ G_4$
11 $K_1^* \leftarrow_\$ \mathcal{K}$
12 $b' \leftarrow \mathsf{A}^{\textsc{Dec}^{\not\perp}_m, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk, c^*, K_b^*)$
13 return $[\![b' = b]\!]$

$\textsc{Dec}^{\not\perp}_m(c \neq c^*)$                    $/\!\!/ G_0$ - $G_1$
14 $m' := \mathsf{Dec}(sk, c)$
15 if $m' = \perp$
   or $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) \neq c$
16     return $K := \mathsf{H}(s, c)$              $/\!\!/ G_0$
17     return $K := \mathsf{H}'(c)$                $/\!\!/ G_1$
18 else
19     return $K := \mathsf{H}(m')$

$\textsc{Dec}^{\not\perp}_m(c \neq c^*)$                    $/\!\!/ G_2$ - $G_4$
20 return $K := \mathsf{H_q}(c)$
```

Fig. 2.35: Games G_0 - G_4 for the proof of Theorem 2.3.3.

Game G_2. In game G_2, we change the game twofold: First, we plug in encryption into random oracle H, i.e., we draw a new random oracle $\mathsf{H_q} \leftarrow_\$ \mathcal{K}^{\mathcal{C}}$ in line 03 and define H in line 04 by letting

$$\mathsf{H}(m) := \mathsf{H_q}(\mathsf{Enc}(pk, \mathsf{m}; \mathsf{G}(\mathsf{m}))) \ .$$

We also make this change explicit for K_0^* in line 09. (Note that as before, we have that $K_0^* := \mathsf{H}(m^*)$.) Second, we change oracle $\textsc{Dec}^{\not\perp}_m$ such that it always returns $K := \mathsf{H_q}(c)$.

We claim that

$$|\Pr[G_0^{\mathsf{A}} = 1] - \Pr[G_1^{\mathsf{A}} = 1]| \leq 16 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \delta \ . \qquad (2.14)$$

In order to prove Equation (2.14), we introduce a sequence of intermediate games in Figure 2.36: We first replace access to G with a modification $\mathsf{G}_{pk,sk}$ that renders the scheme perfectly correct (in game $G_{1.1}$). Second, we plug encryption into the random oracle (game $G_{1.2}$), with the result that $\textsc{Dec}^{\not\perp}_m(c)$ returns $K := \mathsf{H}(\mathsf{Dec}(sk, c)) = \mathsf{H_q}(c)$ for all valid ciphertexts c. Third, we change $\textsc{Dec}^{\not\perp}_m(c)$ such that it always returns $\mathsf{H_q}(c)$ (game $G_{1.3}$). Switching back to a truly random oracle G, we arrive at game G_2.

Game $G_{1.1}$. In game $G_{1.1}$, we enforce that no decryption failure can occur. More formally, for fixed (pk, sk) and message $m \in \mathcal{M}$, we let

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

denote the set of "bad" randomness. We replace random oracle G in line 04 with an oracle $\mathsf{G}_{pk,sk}$ that only samples from "good" randomness: Let f be a $2q$-wise independent hash function, where q counts the number of all queries to G that are triggered by A,

```
GAMES $G_1$ - $G_2$
$(pk, sk) \leftarrow \mathsf{KG}$
$b \leftarrow_\$ \{0,1\}$
$\mathsf{G} \leftarrow_\$ \mathcal{R}^{\mathcal{M}}$                        // $G_1$, $G_2$
$\mathsf{G} := \mathsf{G}_{pk,sk}$                          // $G_{1.1}$ - $G_{1.3}$
$\mathsf{H_q} \leftarrow_\$ \mathcal{K}^{\mathcal{C}}$                            // $G_{1.2}$ - $G_2$
$\mathsf{H} := \mathsf{H_q}(\mathsf{Enc}(pk, -; \mathsf{G}(-)))$     // $G_{1.2}$ - $G_2$
$m^* \leftarrow \mathcal{M}$
$c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$
$K_0^* := \mathsf{H}(m^*)$
$K_1^* \leftarrow_\$ \mathcal{K}$
$b' \leftarrow \mathsf{A}^{\textsc{Dec}_m^{\not\perp}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk, c^*, K_b^*)$
return $[\![b' = b]\!]$

$\textsc{Dec}_m^{\not\perp}(c \neq c^*)$                  // $G_1$ - $G_{1.2}$
$m' := \mathsf{Dec}(sk, c)$
if $m' = \perp$
or $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) \neq c$
    return $K := \mathsf{H}'(c)$
else
    return $K := \mathsf{H}(m')$

$\textsc{Dec}_m^{\not\perp}(c \neq c^*)$                  // $G_{1.3}$ - $G_2$
return $K := \mathsf{H_q}(c)$
```

Fig. 2.36: Games G_1 and G_2, and intermediate games $G_{1.1}$ to $G_{1.3}$, for the proof of Theorem 2.3.3. .

and let $\mathsf{Sample}(Y)$ be a probabilistic algorithm that returns a uniformly distributed $y \leftarrow_\$ Y$. We now define $\mathsf{G}_{pk,sk}$ by

$$\mathsf{G}_{pk,sk}(m) := \mathsf{Sample}(\mathcal{R} \setminus \mathcal{R}_{\text{bad}}(pk, sk, m); f(m)) \ ,$$

where $\mathsf{Sample}(Y; f(m))$ denotes the deterministic execution of $\mathsf{Sample}(Y)$, using the given randomness $f(m)$. Since $2q$-wise independent hash functions are indistinguishable from random oracles for up to q queries, $\mathsf{G}_{pk,sk}(m)$ indeed is identical to uniformly sampling "good" randomness.

In order to upper bound $|\Pr[G_1^{\mathsf{A}} = 1] - \Pr[G_{1.1}^{\mathsf{A}} = 1]|$, we further define

$$\delta(pk, sk, m) := {}^{|\mathcal{R}_{\text{bad}}(pk,sk,m)|}/_{|\mathcal{R}|}$$

as the fraction of bad randomness, and $\delta(pk, sk) := \max_{m \in \mathcal{M}} \delta(pk, sk, m)$. With this notation, we have that $\delta = \mathbf{E}[\max_{m \in \mathcal{M}} \delta(pk, sk, m)]$, where the expectation is taken over $(pk, sk) \leftarrow \mathsf{KG}$.

We now construct an (unbounded, quantum) adversary B against the generic distinguishing problem with bounded probabilities GDPB (see Lemma 1.3.6) in Figure 2.37. B draws a key pair $(pk, sk) \leftarrow \mathsf{KG}$ and computes the parameters $\lambda(m)$ of the generic distinguishing problem as $\lambda(m) := \delta(pk, sk, m)$, which are bounded by $\lambda := \delta(pk, sk)$. To analyse B, we first fix (pk, sk). In the case that B is run in game $\mathsf{GDPB}_{\lambda(pk,sk),1}$, the random variable $\mathsf{F}(m)$ is bernoulli-distributed according to $B_{\lambda(m)} = B_{\delta(pk,sk,m)}$ for each message $m \in \mathcal{M}$. It is easy to verify that in this case, the random variable $\mathsf{G}(m)$ defined in line 21 if $\mathsf{F}(m) = 0$ and in line 23 if $\mathsf{F}(m) = 1$ is uniformly distributed in $\mathcal{R}$. Hence, G is a random oracle and $\mathsf{B}^{|\mathsf{F}\rangle}$ perfectly simulates game G_1 if executed in game

$\mathsf{GDPB}_{\lambda(pk,sk),1}$. In the case that B is run in game $\mathsf{GDPB}_{\lambda(pk,sk),0}$, the random variable $\mathsf{F}(m)$ is always 0 and B provides access to $\mathsf{G}_{pk,sk}$, thereby perfectly simulating game $G_{1.1}$.

$$|\Pr[G_1^{\mathsf{A}} = 1] - \Pr[G_{1.1}^{\mathsf{A}} = 1]| = |\Pr[\mathsf{GDPB}_{\lambda,1}^{\mathsf{B}} = 1] - \Pr[\mathsf{GDPB}_{\lambda,0}^{\mathsf{B}} = 1]| \ .$$

Since B issues at most $q_{\mathsf{G}} + q_D + 1$ queries to F, we can apply Lemma 1.3.6 to obtain

$$|\Pr[\mathsf{GDPB}_{\lambda,1}^{\mathsf{B}} = 1] - \Pr[\mathsf{GDPB}_{\lambda,0}^{\mathsf{B}} = 1]| \le 8 \cdot (q_{\mathsf{G}} + q_D + 2)^2 \cdot \delta \ .$$

```
$\mathsf{B}_1 = \mathsf{B}_1'$
01 $(pk, sk) \leftarrow \mathsf{KG}$
02 for $m \in \mathcal{M}$
03     $\lambda(m) := \delta(pk, sk, m)$
04 return $(\lambda(m))_{m \in \mathcal{M}}$

$\mathsf{B}_2^{|\mathsf{F}\rangle}$ and $\mathsf{B}_2'^{|\mathsf{F}\rangle}$
05 Pick $2q$-wise hash $f$
06 $b \leftarrow_\$ \{0,1\}$
07 $\mathsf{H} := \mathsf{H_q}(\mathsf{Enc}(pk, -; \mathsf{G}(-)))$      $/\!\!/\mathsf{B}_2'$
08 $m^* \leftarrow \mathcal{M}$
09 $c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$
10 $K_0^* := \mathsf{H}(m^*)$                      $/\!\!/\mathsf{B}_2$
11 $K_0^* := \mathsf{H_q}(c^*)$                     $/\!\!/\mathsf{B}_2'$
12 $K_1^* \leftarrow_\$ \mathcal{K}$
13 $b' \leftarrow \mathsf{A}^{\mathrm{Dec}_m^{\not\perp}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk, c^*, K_b^*)$
14 return $[\![b' = b]\!]$

$\mathrm{Dec}_m^{\not\perp}(c \neq c^*)$                  $/\!\!/$Adversary $\mathsf{B}$
15 $m' := \mathsf{Dec}'(sk, c)$
16 if $m' = \perp$
   or $\mathsf{Enc}(pk, m'; \mathsf{G}(m')) \neq c$
17     return $K := \mathsf{H}'(c)$
18 else return $K := \mathsf{H}(m')$

$\mathrm{Dec}_m^{\not\perp}(c \neq c^*)$                  $/\!\!/$Adversary $\mathsf{B}'$
19 return $K := \mathsf{H_q}(c)$

$\mathsf{G}(m)$
20 if $\mathsf{F}(m) = 0$
21     $\mathsf{G}(m) := \mathsf{Sample}(\mathcal{R} \setminus \mathcal{R}_{\text{bad}}(pk, sk, m); f(m))$
22 else
23     $\mathsf{G}(m) := \mathsf{Sample}(\mathcal{R}_{\text{bad}}(pk, sk, m); f(m))$
24 return $\mathsf{G}(m)$
```

Fig. 2.37: Adversaries B and B' for the proof of Theorem 2.3.3, executed in game $\mathsf{GDPB}_{\delta(pk,sk)}$ with access to F. Note that B (B') can provide quantum access to random oracle H ($\mathsf{H} := \mathsf{H_q}(\mathsf{Enc}(pk, -; \mathsf{G}(-)))$) since they are unbounded.

Game $G_{1.2}$. In game $G_{1.2}$, we plug in encryption into random oracle H by picking a random oracle $\mathsf{H_q} \leftarrow_\$ \mathcal{K}^{\mathcal{C}}$ in line 05 and letting $\mathsf{H} := \mathsf{H_q}(\mathsf{Enc}(pk, -; \mathsf{G}(-)))$ in line 06. Note that with this definition we have that $K_0^* := \mathsf{H}(m^*) = \mathsf{H_q}(c^*)$, and that $\mathrm{Dec}_m^{\not\perp}$ returns $K := \mathsf{H}(m') = \mathsf{H_q}(c)$ for valid ciphertexts c.

Since $\mathsf{G} = \mathsf{G}_{pk,sk}$ only samples good randomness, the deterministic encryption algorithm $\mathsf{Enc}(pk, -; \mathsf{G}(-))$ is rendered perfectly correct and hence, injective. Since $\mathsf{Enc}(pk, -; \mathsf{G}(-))$ is injective, H still is uniformly random and A's view is identical in both games.

$$\Pr[G_{1.1}^{\mathsf{A}} = 1] = \Pr[G_{1.2}^{\mathsf{A}} = 1] \ .$$

Game $G_{1.3}$. In game G_3, we change oracle $\textsc{Dec}_m^{\not\perp}$ such that it always returns $K := \mathsf{H_q}(c)$. We will now argue that this change does not affect A's view:

For valid ciphertexts c, we already had that $\textsc{Dec}_m^{\not\perp}(c) = \mathsf{H}(m') = \mathsf{H_q}(c)$ in game G_2, the response of $\textsc{Dec}_m^{\not\perp}$ could thus only differ for ciphertexts c that are not valid. It is easy to verify that since $\mathsf{G} = \mathsf{G}_{pk,sk}$ only samples good randomness, no non-valid ciphertext could lie in the range of $\mathsf{Enc}(pk, -; \mathsf{G}(-))$. But if c does not lie in the range of $\mathsf{Enc}(pk, -; \mathsf{G}(-))$, then oracle $\textsc{Dec}_m^{\not\perp}(c)$ returns in both games a random value that can not possibly correlate to any random oracle query to H, therefore $\textsc{Dec}_m^{\not\perp}(c)$ is a random value independent of all other input to A in both games. We have shown that A's view is identical in both games and

$$\Pr[G_{1.2}^{\mathsf{A}} = 1] = \Pr[G_{1.3}^{\mathsf{A}} = 1] \ .$$

Game $G_{1.4}$. In game $G_{1.4}$, we switch back to using a truly random oracle G. Consider adversary B' given in Figure 2.37. Since B' issues at most $q_{\mathsf{G}} + q_{\mathsf{H}} + 1$ queries to F, we can apply the same reasoning as for the gamehop from game G_1 to $G_{1.1}$ to obtain

$$|\Pr[G_{1.4}^{\mathsf{A}} = 1] - \Pr[G_2^{\mathsf{A}} = 1]| \leq 8 \cdot (q_{\mathsf{G}} + q_{\mathsf{H}} + 2)^2 \cdot \delta \ .$$

Combining the bounds proves Equation (2.14). The rest of the proof proceeds similiar to the proof in [SXY18], aside from the fact that we consider the particular scheme $\mathsf{T}[\mathsf{PKE}, \mathsf{G}]$ instead of a generic deterministic encryption scheme.

Game G_3. In game G_3, we replace the challenge ciphertext c^* with a fake encryption in line 07. Consider the adversary $\mathsf{C_{DS}}$ against the disjoint simulatability of $\mathsf{T}[\mathsf{PKE}, \mathsf{G}]$ given in Figure 2.38. $\mathsf{C_{DS}}$ runs in the time that is required to run A and to simulate $\mathsf{H_q}(\mathsf{Enc}(pk, -; \mathsf{G}(-)))$ for q_{H} many queries. Since $\mathsf{C_{DS}}$ perfectly simulates game G_2 if run on input $c^* := \mathsf{Enc}(pk, m^*; \mathsf{G}(m^*))$ for a random message m^*, and game G_3 if run with a fake ciphertext,

$$|\Pr[G_2^{\mathsf{A}} = 1] - \Pr[G_3^{\mathsf{A}} = 1]| = \mathrm{Adv}^{\mathsf{DS}}_{\mathsf{T}[\mathsf{PKE},\mathsf{G}]}(\mathsf{C_{DS}}) \ .$$

Since $\mathsf{C_{DS}}$ issues at most $q_{\mathsf{RO}} = q_{\mathsf{G}} + q_{\mathsf{H}}$ many queries to G, we can apply Lemma 2.3.1 to obtain that there exist an adversary $\mathsf{B_{DS}}$ and an adversary $\mathsf{B_{IND}}$ such that

$$\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{T}[\mathsf{PKE},\mathsf{G}]}(\mathsf{C_{DS}}) \leq \mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE}}(\mathsf{B_{DS}}) + 2 \cdot \sqrt{2q_{\mathsf{RO}} \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B_{IND}})} + \frac{4q_{\mathsf{RO}}}{\sqrt{|\mathcal{M}|}} \ .$$

```
$\mathsf{C_{DS}}^{|\mathsf{G}\rangle}(pk, c^*)$
$b \leftarrow_\$ \{0,1\}$
$K_0^* := \mathsf{H_q}(c^*)$
$K_1^* \leftarrow_\$ \mathcal{K}$
$b' \leftarrow \mathsf{A}^{\textsc{Dec}_m^{\not\bot}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk, c^*, K_b^*)$
$\mathbf{return}\ [\![b' = b]\!]$

$\textsc{Dec}_m^{\not\bot}(c \neq c^*)$
$\mathbf{return}\ K := \mathsf{H_q}(c)$
```

Fig. 2.38: Adversary $\mathsf{C_{DS}}$ against the disjoint simulatability of $\mathsf{T}[\mathsf{PKE}, \mathsf{G}]$ for the proof of Theorem 2.3.3. Oracle H is defined as in games G_2 and G_3.

Game G_4. In game G_4, we replace K_0^* with a uniformly random key in line 10. Since both K_0^* and K_1^* are independent of all other input to A in game G_4,

$$\Pr[G_4^{\mathsf{A}} \Rightarrow 1] = 1/2 \ ,$$

and it remains to upper bound $|\Pr[G_3^{\mathsf{A}} = 1] - \Pr[G_4^{\mathsf{A}} = 1]|$. Since A's view only differs if any of the oracle answers actually contains $\mathsf{H_q}(c^*)$, and since queries to $\textsc{Dec}_m^{\not\bot}$ on c^* are explicitly forbidden, it is sufficient to analyse whether A could trigger a query to $\mathsf{H_q}$ containing c^* via one of its queries to H. We know that the input register of these queries only contain superpositions of the form $\sum_m \alpha_m |\mathsf{Enc}(pk, m; \mathsf{G}(m))\rangle$, which can not contain the fake ciphertext c^* unless it lies in the range of $\mathsf{Enc}(pk, -; \mathsf{G}(-))$. Since we assume PKE to be ϵ_{dis}-disjoint,

$$|\Pr[G_3^{\mathsf{A}} = 1] - \Pr[G_4^{\mathsf{A}} = 1]| \leq \epsilon_{\text{dis}} \ .$$

□

2.3.3 *Transformation* $\mathsf{FO}_m^{\not\bot} \circ \mathsf{Punc}$*: From* IND-CPA *to* IND-CCA

In this section, we show that the requirement of disjoint simulatability can be waived with negligible loss of efficiency: To rely on IND-CPA alone, all that has to be done is to puncture the message space at one message, and use this message to sample fake encryptions. We formalise this below by defining transformation Punc. Punc achieves (probabilistic) simulatability and maintains IND-CPA security. Note that we do not consider *disjoint* simulatability, as it will turn out that disjointness is not needed in order to achieve IND-CCA security of $\mathsf{FO}_m^{\not\bot} \circ \mathsf{Punc}$.

The Construction. To a public-key encryption scheme $\mathsf{PKE}_0 = (\mathsf{KG}_0, \mathsf{Enc}_0, \mathsf{Dec}_0)$ with message space $\mathcal{M}_0$, and to a message $\hat{m} \in \mathcal{M}_0$, we associate $\mathsf{PKE} := \mathsf{Punc}[\mathsf{PKE}_0, \hat{m}] := (\mathsf{KG}_0, \mathsf{Enc}_0, \mathsf{Dec}_0)$ with message space $\mathcal{M} := \mathcal{M}_0 \setminus \{\hat{m}\}$. We furthermore define sampling algorithm fakeEnc in Figure 2.39.

```
fakeEnc($pk$)
$c \leftarrow \mathsf{Enc}_0(pk, \hat{m})$
return $c$
```

Fig. 2.39: Fake encryption sampling algorithm fakeEnc.

Given that PKE differs from PKE_0 only in the minimally restricted message space, it is easy to verify that PKE (tightly) inherits IND-CPA security and δ-worst-case correctness from PKE_0. The following lemma furthermore states that IND-CPA security of PKE_0 implies simulatability of PKE.

Lemma 2.3.4 (DS of Punc)**.** For all adversaries A, there exists an IND-CPA adversary B such that

$$\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE}}(\mathsf{A}) = 2 \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}_0}(\mathsf{B}) \ .$$

Proof. Let A be an adversary against DS of PKE, and consider the IND-CPA adversary $\mathsf{B} := (\mathsf{B}_1, \mathsf{B}_2)$ against PKE_0 given in Figure 2.40. If B is run in game $\mathsf{IND\text{-}CPA}_{\mathsf{PKE}_0}$ with $b = 0$, it runs A with an encryption of a message that was randomly picked from $\mathcal{M}$, and if B is run in game $\mathsf{IND\text{-}CPA}_{\mathsf{PKE}_0}$ with $b = 1$, it runs A with a fake ciphertext, hence

$$\mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{B}) = \frac{1}{2} \cdot \mathrm{Adv}^{\mathsf{DS}}_{\mathsf{PKE}}(\mathsf{A}) \ .$$

```
B_1($pk$)
$m \leftarrow_\$ \mathcal{M}_0 \setminus \{\hat{m}\}$
return $(m, \hat{m})$

B_2($pk, c$)
$b' \leftarrow \mathsf{A}(pk, c)$
return $b'$
```

Fig. 2.40: IND-CPA adversary $\mathsf{B} = (\mathsf{B}_1, \mathsf{B}_2)$ for the proof of Lemma 2.3.4.

□

We can now combine Lemma 2.3.4 with Lemma 2.3.1 to obtain that combining T with Punc achieves (deterministic) simulatability from IND-CPA security.

Corollary 2.3.5 (DS of $\mathsf{T} \circ \mathsf{Punc}$)**.** For all adversaries A issuing at most q_{G} queries to G, there exist two adversaries B_1 and B_2 such that

$$\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{T}[\mathsf{Punc}[\mathsf{PKE}_0, \hat{m}], \mathsf{G}]}(\mathsf{A}) \leq 2 \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}_0}(\mathsf{B}_1) + 2 \cdot \sqrt{2 q_{\mathsf{G}} \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}_0}(\mathsf{B}_2)} + \frac{4 q_{\mathsf{G}}}{\sqrt{|\mathcal{M}| - 1}} \ ,$$

and the running time of each adversary is about that of B.

The following theorem establishes that plugging in transformation Punc (before $\mathsf{FO}^{\not\perp}_m$) achieves IND-CCA security from IND-CPA security alone, as long as PKE is γ-spread (see Definition 1.1.2).

Theorem 2.3.6 (IND-CCA security of $\mathsf{FO}^{\not\perp}_m \circ \mathsf{Punc}$.)**.** Assume PKE_0 to be δ-worst-case correct and γ-spread, and let $\hat{m} \in \mathcal{M}$. Let $\mathsf{KEM} := \mathsf{FO}^{\not\perp}_m[\mathsf{Punc}[\mathsf{PKE}_0, \hat{m}], \mathsf{G}, \mathsf{H}]$. Then, for any (quantum) IND-CCA adversary A issuing at most q_D (classical) queries to the decapsulation oracle $\textsc{Dec}^{\not\perp}_m$, at most q_{H} quantum queries to H, and at most q_{G} quantum queries to G, there exist CPA adversaries B_1 and B_2 against PKE_0 such that

$$\begin{aligned}\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}(\mathsf{A}) \leq{}& 2 \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}_0}(\mathsf{B}_1) + 2 \cdot \sqrt{2 q_{\mathsf{RO}} \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}_0}(\mathsf{B}_2)} \\ &+ 24 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \delta + \frac{4 q_{\mathsf{RO}}}{\sqrt{|\mathcal{M}| - 1}} + 2^{-\gamma} + q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}} \ ,\end{aligned}$$

where q_{RO} counts the total number of random oracle queries, and the running time of B_1 and B_2 is about that of A.

Proof. In order to prove Theorem 2.3.6, we revisit the proof of Theorem 2.3.3 and show how we can modify it such that it works for $\mathsf{Punc}[\mathsf{PKE}_0, \hat{m}]$ without having to rely on the disjointess property.

Executing the first 3 game-hops (see Figure 2.35), we achieve that $\textsc{Dec}^{\not\perp}_m(c)$ always returns $\mathsf{H}_{\mathsf{q}}(c)$, and that the challenge ciphertext c^* is replaced with a fake encryption, i.e., an encryption of $\hat{m}$.

Since PKE_0 is δ-correct, so is $\mathsf{Punc}[\mathsf{PKE}, \hat{m}]$, hence

$$\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}(\mathsf{A}) \leq |\Pr[G_2^{\mathsf{A}} \Rightarrow 1] - 1/2| + q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}} + 16 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \delta \ ,$$

and according to Corollary 2.3.5, there exist CPA adversaries B_1 and B_2 against PKE_0 such that

$$|\Pr[G_2^{\mathsf{A}} = 1] - \Pr[G_3^{\mathsf{A}} = 1]| \leq 2 \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}_0}(\mathsf{B}_1) + 2 \cdot \sqrt{2 q_{\mathsf{RO}} \cdot \mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}_0}(\mathsf{B}_2)} + \frac{4 q_{\mathsf{RO}}}{\sqrt{|\mathcal{M}| - 1}} \ .$$

In order to justify that we can replace the real key K_0^* with random like in game G_4 of Figure 2.35, we will now show how to circumvent the disjointness requirement: We introduce an intermediate game $G_{3.1}$ in Figure 2.41, in which we replace G with $\mathsf{G}_{pk,sk}$ yet once more. Since the games can be simulated perfectly by an adversary against $\mathsf{GDPB}_{\delta(pk,sk)}$, we once more obtain

$$|\Pr[G_3^{\mathsf{A}} = 1] - \Pr[G_{3.1}^{\mathsf{A}} = 1]| \leq 8 \cdot (q_{\mathsf{RO}} + 2)^2 \cdot \delta \ .$$

```
**GAMES** $G_3$ - $G_4$
$b \leftarrow_\$ \{0,1\}$
$(pk, sk) \leftarrow \mathsf{KG}$
$\mathsf{G} := \mathsf{G}_{pk,sk}$                    $/\!\!/ G_{3.1}$ - $G_4$
$\mathsf{H} := \mathsf{H_q}(\mathsf{Enc}_0(pk, -; \mathsf{G}(-)))$
$c^* \leftarrow \mathsf{Enc}_0(pk, \hat{m})$
$K_0^* := \mathsf{H_q}(c^*)$                   $/\!\!/ G_3$ - $G_{3.1}$
$K_0^* \leftarrow_\$ \mathcal{K}$                              $/\!\!/ G_4$
$K_1^* \leftarrow_\$ \mathcal{K}$
$b' \leftarrow \mathsf{A}^{\textsc{Dec}_{\hat{m}}^{\not\perp}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk, c^*, K_b^*)$
**return** $[\![b' = b]\!]$
```

Fig. 2.41: Games G_3 and G_4, and intermediate game $G_{3.1}$, for the proof of Theorem 2.3.6.

Game G_4. In game G_4, we replace K_0^* with a uniformly random key. Since both K_0^* and K_1^* are independent of all other input to A in game G_4,

$$\Pr[G_4^{\mathsf{A}} \Rightarrow 1] = 1/2 \ .$$

and it remains to upper bound $|\Pr[G_{3.1}^{\mathsf{A}} = 1] - \Pr[G_4^{\mathsf{A}} = 1]|$. Again, the view of A can only differ if A triggers a query to $\mathsf{H_q}$ containing c^* via one of its queries to H, which is only possible if there exists a message m such that $c^* = \mathsf{Enc}(pk, m; \mathsf{G}(m))$. Assume that such a message m exists. We distinguish two sub-cases: $m \neq \hat{m}$ or $m = \hat{m}$.

- Existence of a message $m \neq \hat{m}$ such that $c^* = \mathsf{Enc}(pk, m; \mathsf{G}(m))$ implies that c^* exhibits decryption failure: Since $\mathsf{G} = \mathsf{G}_{pk,sk}$ only samples good randomness, it is implied that $\mathsf{Dec}_0(sk, c^*) = m \neq \hat{m}$. We can hence upper bound the probability of this case by δ.

- In the case that $m = \hat{m}$, we have that $c^* = \mathsf{Enc}_0(pk, \hat{m}; \mathsf{G}(\hat{m}))$. Since c^* is a random encryption, and PKE is γ-spread, we can upper bound the probability of this case by $2^{-\gamma}$.

$$|\Pr[G_{3.1}^{\mathsf{A}} = 1] - \Pr[G_4^{\mathsf{A}} = 1]| \leq \delta + 2^{-\gamma} \ .$$

Collecting the bounds, we can upper bound

$$|\Pr[G_3^{\mathsf{A}} = 1] - \Pr[G_4^{\mathsf{A}} = 1]| \leq 8 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \delta + 2^{-\gamma}$$

□

2.4 Transformation ACWC: Turning Average-Case into Worst-Case Correctness

In this section, we show how to convert average-case correct schemes (recall Definition 1.1.11, page 30) into ones that are worst-case correct, assuming that they come with randomness recovery and invertible encryption (see Definitions 1.1.3 and 1.1.4, page 25).

Our motivation stems from the fact that there exist passively secure schemes that only come with average-case-correctness, but fulfill the requirement above, with one example being the NTTRU scheme given in [LS19][5].

As shown in Section 1.1.3, worst-case and average-case correctness do not generally coincide: The former definition can be strictly stronger than the latter, and equivalency is given if and only if the decryption error is independent of the message (see Lemma 1.1.14 on page 32). As pointed out in [LS19, Section 2.2], independence of the message can be achieved for LPR-Style schemes, but this is not the case for NTRU-style schemes. Since worst-case correctness is required in order to safely apply FO-like transformations, it seems to be crucial for schemes like NTTRU to find a way to deal with this gap.

In fact, to bypass the correctness issue, [LS19] already included a transformation that is quite similar in spirit to the construction we will define below. The transformation given in [LS19], however, results in a small communicative overhead, and more importantly, its formal treatment in the quantum random oracle model was left as an open problem. We therefore now introduce a length-preserving alternative, which we then analyse in the (quantum) random oracle model.

The Construction. Let $\mathsf{PKE}_0 = (\mathsf{KG}_0, \mathsf{Enc}_0, \mathsf{Dec}_0)$ be a public-key encryption scheme with message space $\mathcal{M}_0$ and randomness space $\mathcal{R}_0$. Assume that PKE_0 comes with randomness recovery (see Definition 1.1.3, page Definition 1.1.3). To PKE_0 and random oracle $\mathsf{F} : \mathcal{R}_0 \to \mathcal{M}_0$, we associate $\mathsf{PKE} := \mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}]$. The algorithms of $\mathsf{PKE} = (\mathsf{KG} := \mathsf{KG}_0, \mathsf{Enc}, \mathsf{Dec})$ are defined in Figure 2.42.

In Section 2.4.1, we will prove that ACWC indeed achieves worst-case from average-case correctness.

In Section 2.4.2, we will prove that ACWC achieves IND-CPA security from OW security, assuming that the underlying scheme comes with invertible encryption. Hence, transformation ACWC can be plugged into our implicitly rejecting KEM transformations from Section 2.1.4, resulting in a OW to IND-CCA transformation which only requires

[5] The passively secure NTTRU construction computes ciphertexts as $\mathsf{Enc}(pk, m; r) := pk \cdot r + m$. We can hence define $\mathsf{Inv}(pk, c, r) = c - pk \cdot r$. As discussed in [LS19, Section 3.1], the key generation algorithm can be modified with negligible loss in efficiency such that pk is invertible. We can hence define $\mathsf{Rec}(pk, m, c) = (c - m) \cdot pk^{-1}$.

```
$\mathsf{Enc}(pk, m)$
    $r \leftarrow_\$ \mathcal{R}_0$
    $c := \mathsf{Enc}_0(pk, m \oplus \mathsf{F}(r); r)$
    return $c$

$\mathsf{Dec}(sk, c)$
    $M' := \mathsf{Dec}_0(sk, c)$
    if $M' = \perp$ return $\perp$
    $r' := \mathsf{Rec}(sk, M', c)$
    return $m' := M' \oplus \mathsf{F}(r')$
```

Fig. 2.42: Worst-case correct encryption scheme $\mathsf{PKE} = \mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}]$.

average-case correctness. If PKE_0 is γ-spread, then so is $\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}]$, and we can also apply the variants with explicit rejection.

While our reduction has a linear loss in q, the number of random oracle queries, we will now argue that this loss does not imply worse IND-CCA bounds than previously known bounds for most application scenarios:

1. First, the bound for combining ACWC with any of the FO-like transformations FO from Section 2.1.4 is as tight as the one for FO alone: While all FO-like transformations also achieve IND-CCA security when the scheme is only one-way secure, the proof then loses a factor of q. Intuitively, the loss is simply shifted from T to ACWC.

2. Second, note that ACWC does not require the underlying scheme to be deterministic. There exist tightly secure IND-CCA conversions for schemes that are only one-way secure, as long as they are deterministic and either rigid (recall Section 2.1.3) or perfectly correct (see [Den03, Theorem 4] and [BP18, SXY18]). For one-way secure schemes that are neither deterministic nor perfectly correct, however, we do not know of any IND-CCA conversions coming with better bounds than our FO-bounds (i.e., where the bound does not lose a factor of q).

In Section 2.4.3, we revisit transformation ACWC in the QROM. While a generalisation of Section 2.4.2 to the QROM can be achieved in a straightforward manner by applying a (non-tight) quantum query extraction argument, this simple approach would result in suboptimal IND-CCA bounds when plugging ACWC into $\mathsf{FO}_m^{\not\perp}$, as a (non-tight) query extraction bound already appears in the bound for $\mathsf{FO}_m^{\not\perp}$. We will show that this nested extraction argument can be avoided as follows: There already exist modular security proofs for $\mathsf{FO}_m^{\not\perp}$ (see, e.g., Section 2.3 and [BHH$^+$19]), in which the underlying scheme is required to be IND-CPA secure. We will show how to integrate ACWC into the modularisation from [BHH$^+$19] such that the nested extraction bound can be avoided. For the sake of comparability, we will furthermore show how to adapt the proof in [BHH$^+$19] for schemes that are only OW secure, at the price of losing an additional factor of $2\sqrt{q}$. We do not know of any IND-CCA conversions that come with tighter bounds in the QROM than the one we achieve with this adaption, assuming one-way secure schemes that are neither deterministic nor perfectly correct, and our reduction for $\mathsf{T} \circ \mathsf{ACWC}$ has essentially the same loss as our OW reduction for T alone.

In conclusion, when working with a OW secure scheme that is neither perfectly correct nor deterministic, our resulting $\mathsf{IND\text{-}CCA}$ bounds are as tight as the $\mathsf{IND\text{-}CCA}$ bounds for previously known conversions, in the ROM as well as in the QROM.

During our integration of ACWC into the modularisation from [BHH+19], the crucial step is to simply reprogram both oracles F and G at once, which can furthermore easily be carried over to the proof that $\mathsf{T}[\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}], \mathsf{G}]$ is disjoint simulatable. For completeness, we make the bounds for the KEM resulting from applying $\mathsf{FO}_m^{\not\perp} \circ \mathsf{ACWC}$ explicit in Section 2.4.4. As a corollary, we observe that authenticated key exchange can also be achieved from schemes that are OW secure and average-case correct, at the price of an additional factor $\sqrt{2q}$.

Lastly, we will discuss in Section 2.4.5 how the requirement of randomness recovery can be avoided. The strategy that we came up with, however, requires the scheme to fulfill a stronger security notion (partial one-wayness or $\mathsf{IND\text{-}CPA}$ security).

2.4.1 Proof of Worst-Case Correctness

Theorem 2.4.1. Assume PKE_0 to be δ_0-average-case-correct. Then PKE is δ-worst-case-correct, where

$$\delta = \delta_0 + \sqrt{\frac{\log(|\mathcal{M}_0|)}{|\mathcal{R}_0|}} + |\mathcal{M}_0|^{-1.88} \ .$$

Proof. We want to upper bound $\delta = \mathbb{E}_{\mathsf{KG},\mathsf{F}} \max_{m \in \mathcal{M}_0} \Pr[\mathsf{Dec}(\mathsf{Enc}(m)) \neq m]$, where the expectation is taken over the internal randomness of KG and the choice of random oracle F, and the probability is taken over the internal randomness of Enc.

We will first fix an arbitrary key pair $(pk, sk) \in \mathrm{supp}(\mathsf{KG})$ and claim an upper bound for

$$\delta(pk, sk) := \mathbb{E}_{\mathsf{F}} \max_{m \in \mathcal{M}_0} \Pr[\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) \neq m] \ ,$$

where the expectation is taken only over the choice of random oracle F. Taking the expectation over the internal randomness of KG, the claimed bound is proven.

In order to upper bound $\delta(pk, sk)$, we will first rewrite the average-case correctness term of the underlying scheme PKE_0: Note that by letting

$$\delta_0(pk, sk) := \Pr_{m,r}[\mathsf{Dec}_0(sk, \mathsf{Enc}_0(pk, m; r)) \neq m]$$

for uniform message m and randomness r, we have that

$$\delta_0 = \mathbb{E}_{\mathsf{KG}_0}\, \delta_0(pk, sk) \ .$$

We claim that for all key pairs $(pk, sk) \in \operatorname{supp}(\mathsf{KG} = \mathsf{KG}_0)$,

$$\delta(pk, sk) \leq \delta_0(pk, sk) + \sqrt{\frac{\log(|\mathcal{M}_0|)}{|\mathcal{R}_0|}} + |\mathcal{M}_0|^{-1.88} \ . \tag{2.15}$$

Taking the expectation over the internal randomness of $\mathsf{KG} = \mathsf{KG}_0$ then yields the bound claimed in Theorem 2.4.1, it hence remains to prove the upper bound given in Equation (2.15). For the rest of the proof, we hence consider the key pair $(pk, sk) \in \operatorname{supp}(\mathsf{KG})$ to be fixed.

With respect to this key pair, we define the set of bad message-randomness combinations for the underlying scheme: We let

$$\mathrm{BAD}_0(pk, sk) := \{(m, r) \in \mathcal{M}_0 \times \mathcal{R}_0 \mid \mathsf{Dec}_0(sk, \mathsf{Enc}_0(pk, m; r)) \neq m\} \ .$$

It is easy to verify that m exhibits decryption failure with respect to the transformed scheme PKE, i.e., that $\mathsf{Dec}(\mathsf{Enc}(m; r)) \neq m$, only if $(m \oplus \mathsf{F}(r), r) \in \mathrm{BAD}_0(pk, sk)$. Note that here we require that recovering the wrong randomness occurs only if decryption failure occurs. Thus,

$$\delta(pk, sk) \leq \mathbb{E}_{\mathsf{F}} \max_{m \in \mathcal{M}_0} \Pr[(m \oplus \mathsf{F}(r), r) \in \mathrm{BAD}_0(pk, sk)] \ .$$

To further upper bound the right-hand side, we define

$$\delta(\mathsf{F}) := \max_{m \in \mathcal{M}_0} \Pr[(m \oplus \mathsf{F}(r), r) \in \mathrm{BAD}_0(pk, sk)] \ .$$

With this notation, it remains to upper bound $\mathbb{E}_{\mathsf{F}}[\delta(\mathsf{F})]$. Relative to any positive real t, $\mathbb{E}_{\mathsf{F}}[\delta(\mathsf{F})]$ can be split into three summands:

$$\begin{aligned}
\mathbb{E}_{\mathsf{F}}[\delta(\mathsf{F})] &= \sum_{\mathsf{F}} \Pr[\mathsf{F}] \cdot \delta(\mathsf{F}) \\
&= \sum_{\mathsf{F}:\delta(\mathsf{F}) < \delta_0(pk,sk)+t} \Pr[\mathsf{F}] \cdot \delta(\mathsf{F}) \ + \sum_{\mathsf{F}:\delta(\mathsf{F}) \geq \delta_0(pk,sk)+t} \Pr[\mathsf{F}] \cdot \delta(\mathsf{F}) \\
&\leq \delta_0(pk, sk) + t \ + \Pr_{\mathsf{F}}[\delta(\mathsf{F}) \geq \delta_0(pk, sk) + t] \ .
\end{aligned}$$

In order to upper bound the probability in the last line, we now define

$$\delta(\mathsf{F},m) := \Pr_r[(m \oplus \mathsf{F}(r),r) \in \mathrm{BAD}_0(pk,sk)] = \mathbb{E}_r\, \delta(\mathsf{F},m,r) \ ,$$

where

$$\delta(\mathsf{F},m,r) := [\![(m \oplus \mathsf{F}(r),r) \in \mathrm{BAD}_0(pk,sk)]\!] \ .$$

Applying a union bound, we obtain

$$\begin{aligned}\Pr_{\mathsf{F}}[\delta(\mathsf{F}) \geq \delta_0(pk,sk)+t] &= \Pr_{\mathsf{F}}[\max_{m\in\mathcal{M}_0} \delta(\mathsf{F},m) \geq \delta_0(pk,sk)+t] \\ &= \Pr_{\mathsf{F}}[\exists\ m \text{ s.th. } \delta(\mathsf{F},m) \geq \delta_0(pk,sk)+t] \\ &\leq |\mathcal{M}_0| \cdot \max_{m\in\mathcal{M}_0} \Pr_{\mathsf{F}}[\delta(\mathsf{F},m) \geq \delta_0(pk,sk)+t] \ .\end{aligned}$$

We claim that for all positive reals t and all messages m it holds that

$$\Pr_{\mathsf{F}}[\delta(\mathsf{F},m) \geq \delta_0(pk,sk)+t] \leq \exp(-2\cdot|\mathcal{R}_0|\cdot t^2) \ . \tag{2.16}$$

Assuming that Equation (2.16) holds, we obtain that

$$\mathbb{E}_{\mathsf{F}}[\delta(\mathsf{F})] \leq \delta_0(pk,sk)+t+|\mathcal{M}_0|\cdot\exp(-2\cdot|\mathcal{R}_0|\cdot t^2) \ .$$

Letting $t := \sqrt{\frac{\log(|\mathcal{M}_0|)}{|\mathcal{R}_0|}}$ and taking into account that

$$|\mathcal{M}_0|\cdot\exp(-2\cdot\log(|\mathcal{M}_0|)) = |\mathcal{M}_0|\cdot|\mathcal{M}_0|^{-2\cdot\log(e)} < |\mathcal{M}_0|^{-1.88}$$

yields the upper bound claimed in Equation (2.15), it hence remains to prove Equation (2.16).

In the following, we consider m to be fixed. For any $r \in \mathcal{R}_0$, we can define a random variable X_r by picking F uniformly at random and returning $\delta(\mathsf{F},m,r)$. Since (X_r) is a collection of independent variables, and the support of $X_r - \mathbb{E}_r X_r$ lies in an interval of length 1 for each r, we can apply Hoeffding's inequality to obtain

$$\Pr_{\mathsf{F}}\left[\sum_r (X_r - \mathbb{E}_{\mathsf{F}}[X_r]) \geq t\cdot|\mathcal{R}_0|\right] \leq \exp\left(-2\cdot|\mathcal{R}_0|\cdot t^2\right) \ .$$

We can now rewrite

$$\Pr_{\mathsf{F}}\left[\sum_r (X_r - \mathbb{E}_{\mathsf{F}}[X_r]) \geq t \cdot |\mathcal{R}_0|\right] = \Pr_{\mathsf{F}}\left[\sum_r \frac{1}{|\mathcal{R}_0|} X_r - \sum_r \frac{1}{|\mathcal{R}_0|} \mathbb{E}_{\mathsf{F}} X_r \geq t\right]$$
$$= \Pr_{\mathsf{F}}\left[\mathbb{E}_r[\delta(\mathsf{F}, m, r)] - \mathbb{E}_{\mathsf{F},r}[\delta(\mathsf{F}, m, r)] \geq t\right]$$
$$= \Pr_{\mathsf{F}}\left[\delta(\mathsf{F}, m) \geq \mathbb{E}_{\mathsf{F},r}[\delta(\mathsf{F}, m, r)] + t\right] \ .$$

Since

$$\mathbb{E}_{\mathsf{F},r}[\delta(\mathsf{F}, m, r)] = \mathbb{E}_r[\mathbb{E}_{\mathsf{F}}[\delta(\mathsf{F}, m, r)]]$$
$$= \mathbb{E}_r[\mathbb{E}_{\tilde{m}}[[\![(m \oplus \tilde{m}, r) \in \mathrm{BAD}_0(pk, sk)]\!]]]$$
$$= \mathbb{E}_r[\mathbb{E}_{m'}[[\![(m', r) \in \mathrm{BAD}_0(pk, sk)]\!]]] = \delta_0(pk, sk) \ ,$$

we have just proven the upper bound claimed in Equation (2.16). □

2.4.2 From OW *to* IND-CPA*, in the ROM*

The following theorem states that transformation ACWC turns any OW secure scheme into one that is IND-CPA secure.

Theorem 2.4.2 (PKE_0 OW/CPA $\overset{\text{ROM}}{\Rightarrow}$ PKE IND-CPA)**.** Assume PKE_0 to come with invertible encryption (see Definition 1.1.4). For any IND-CPA adversary A against PKE that issues at most q queries to random oracle F, there exist a OW adversary B against PKE_0 such that

$$\mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{A}) \leq \frac{q_1}{|\mathcal{R}|} + q_2 \cdot \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}_0}(\mathsf{B}) \ ,$$

where q_1 is the number of queries issued by A_1, and q_2 is the number of queries issued by A_2, and the running time of B is about that of A.

Proof. Consider an adversary A playing the IND-CPA game for PKE, issuing at most q queries to F, and the sequence of games given in Figure 2.43.

Game G_0. Since game G_0 is the original IND-CPA game,

$$\mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE}}(\mathsf{A}) = |\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \frac{1}{2}| \ .$$

Game G_1. In game G_1, we indicate which part of the adversary is run by setting a to

```
GAMES $G_0$ - $G_3$
01 $(pk, sk) \leftarrow \mathsf{KG}_0$
02 $b \leftarrow_\$ \{0,1\}$
03 $r^* \leftarrow_\$ \mathcal{R}$
04 $a := 1$                                   $/\!\!/ G_1$-$G_3$
05 $(m_0^*, m_1^*, st) \leftarrow \mathsf{A}_1^{\mathsf{F}}(pk)$
06 $a := 2$                                   $/\!\!/ G_1$-$G_3$
07 $\tilde{m}^* := \mathsf{F}(r^*)$           $/\!\!/ G_0$-$G_1$
08 $M^* := m_b^* \oplus \tilde{m}^*$          $/\!\!/ G_0$-$G_1$
09 $M^* \leftarrow_\$ \mathcal{M}$            $/\!\!/ G_2$-$G_3$
10 $c^* := \mathsf{Enc}_0(pk, M^*; r^*)$
11 $b' \leftarrow \mathsf{A}_2^{\mathsf{F}}(pk, c^*, st)$
12 return $[\![b' = b]\!]$

$\mathsf{F}(r)$
13 if $\exists m$ such that $(r, m) \in \mathfrak{L}_{\mathsf{F}}$
14     return $m$
15 if $r = r^*$ and $a = 1$
16     $\mathrm{QUERY}_1$ := true             $/\!\!/ G_1$-$G_3$
17     ABORT                                  $/\!\!/ G_1$-$G_3$
18 if $r = r^*$ and $a = 2$
19     return $M^* \oplus m_b^*$              $/\!\!/ G_2$
20     $\mathrm{QUERY}_2$ := true             $/\!\!/ G_3$
21     ABORT                                  $/\!\!/ G_3$
22 $m \leftarrow_\$ \mathcal{M}$
23 $\mathfrak{L}_{\mathsf{F}} := \mathfrak{L}_{\mathsf{F}} \cup \{(r, m)\}$
24 return $m$
```

Fig. 2.43: Games G_0 - G_3 for the proof of Theorem 2.4.2. .

1 before running A_1 (see line 04), and to 2 right after A_1 puts out its challenge messages (line 06). We raise flag QUERY_1 and abort if random oracle F is ever queried on r^* by A_1, i.e., if random oracle F is ever queried on r^* while a is still set to 1 (lines 16 and 17). Since both games proceed identically unless QUERY_1 was risen,

$$|\Pr[G_0^{\mathsf{A}} = 1] - \Pr[G_1^{\mathsf{A}} = 1]| \leq \Pr[\mathrm{QUERY}_1] \ ,$$

and since A_1's input is independent of r^*,

$$\Pr[\mathrm{QUERY}_1] \leq \frac{q_1}{|\mathcal{R}|} \ .$$

Game G_2. In game G_2, we make two changes: First, instead of defining the challenge plaintext as $M^* := m_b^* \oplus \mathsf{F}(r^*)$, we pick message M^* uniformly at random (line 09). Second, we change the random oracle such that it is kept consistent, rendering this change purely conceptual: We let $\mathsf{F}(r^*) := M^* \oplus m_b^*$ (line 19). Since M^* is uniformly random, so is $\mathsf{F}(r^*) = M^* \oplus m_b^*$, hence F remains uniformly random. Furthermore, it is easy to verify that c^* still is an encryption of $m_b^* \oplus \mathsf{F}(r^*)$, therefore

$$\Pr[G_1^{\mathsf{A}} \Rightarrow 1] = \Pr[G_2^{\mathsf{A}} \Rightarrow 1] \ .$$

Game G_3. In game G_3, we raise flag QUERY_2 and abort if A_2 ever queries F on r^* (lines 20 and 21). With this change, M^* is rendered independent of m_b^* as long as the game does not abort, therefore b is independent of A's view and

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] = \frac{1}{2} \ .$$

Since both games proceed identically unless QUERY_2 occurs, and since QUERY_2 only occurs if QUERY_1 did not occur,

$$|\Pr[G_2^{\mathsf{A}} = 1] - \Pr[G_3^{\mathsf{A}} = 1]| \leq \Pr[\text{QUERY}_2] = \Pr[\text{QUERY}_2 \wedge \neg\text{QUERY}_1] \ .$$

Collecting the probabilities, we obtain

$$\text{Adv}_{\mathsf{PKE}}^{\mathsf{IND\text{-}CPA}}(\mathsf{A}) \leq \frac{q_1}{|\mathcal{R}|} + \Pr[\text{QUERY}_2 \wedge \neg\text{QUERY}_1] \ .$$

To upper bound $\Pr[\text{QUERY}_2 \wedge \neg\text{QUERY}_1]$, we construct adversary B against the OW security of PKE_0 in Figure 2.44.

Adversary B is run on input (pk, c^*), where $c^* \leftarrow \mathsf{Enc}_0(pk, M^*; r^*)$ for some random message M^* and some uniform randomness r^*. B keeps track of the random oracle queries issued by A_2 (see line 13). After running A_2, it picks one of these queries at random (see line 04), and uses it to compute its oneway-guess as $M' := \mathsf{Inv}(pk, c^*, r')$ in line 05. If $r' = r^*$, we have that $M' = \mathsf{Inv}(pk, \mathsf{Enc}_0(pk, M^*; r^*), r^*) = M^*$. Since B perfectly simulates game G_2 until QUERY_2 occurs, and wins with probability $1/q_2$ if QUERY_2 occurs,

$$\Pr[\text{QUERY}_2 \wedge \neg\text{QUERY}_1] \leq q_2 \cdot \text{Adv}_{\mathsf{PKE}_0}^{\mathsf{OW}}(\mathsf{B}) \ .$$

```
B(pk, c*)
01 $i := 0$
02 $(m_0^*, m_1^*, st) \leftarrow \mathsf{A}_1^{\mathsf{F}}(pk)$
03 $b' \leftarrow \mathsf{A}_2^{\mathsf{F}}(pk, c^*, st)$
04 $r' \leftarrow_\$ \mathfrak{L}_{\mathcal{R}}$
05 $M' := \mathsf{Inv}(pk, c^*, r')$
06 return $M'$
```

```
F(r)
07 $i{+}{+}$
08 if $\exists m$ such that $(r, m) \in \mathfrak{L}_{\mathsf{F}}$
09     return $m$
10 $m \leftarrow_\$ \mathcal{M}$
11 $\mathfrak{L}_{\mathsf{F}} := \mathfrak{L}_{\mathsf{F}} \cup \{(r, m)\}$
12 if $i > q_1$
13     $\mathfrak{L}_{\mathcal{R}} := \mathfrak{L}_{\mathcal{R}} \cup \{r\}$
14 return $m$
```

Fig. 2.44: Adversary B for the proof of Theorem 2.4.2.

□

2.4.3 $\mathsf{T} \circ \mathsf{ACWC}$: *From* OW *to* OW *and* DS*, in the QROM*

In this section, we show that the combined transformation $\mathsf{T} \circ \mathsf{ACWC}$ achieves OW security, in the QROM. Since OW security was the intermediate notion in the modularisation of $\mathsf{FO}_m^{\not\perp} = \mathsf{U}_m^{\not\perp} \circ \mathsf{T}$ that was given in [BHH+19], we can hence integrate transformation ACWC into the analysis of $\mathsf{FO}_m^{\not\perp}$ from [BHH+19]. The integration technique is fairly straightforward: Instead of decoupling the challenge ciphertext only from G, it is at the same time decoupled from F.

Using the same integration technique, we then show that $\mathsf{TPunc}[\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}], \mathsf{G}]$ is simulatable; we can hence integrate transformation ACWC into the analysis of $\mathsf{FO}_m^{\not\perp}$ from Section 2.3.3 to obtain a KEM that is $\mathsf{IND}\text{-}\mathsf{CCA}$ secure even for schemes that are less than worst-case correct. We want to stress that without ACWC, a generalisation of Section 2.3.3 towards OW security would not have been straightforward, as simulatability of Punc requires $\mathsf{IND}\text{-}\mathsf{CPA}$ security.

In order to make our OW bound for $\mathsf{T} \circ \mathsf{ACWC}$ more comparable with the bound that was achieved in [BHH+19] for T alone, we want to stress that it assumed $\mathsf{IND}\text{-}\mathsf{CPA}$ security. We will now first recall the security statement for T from [BHH+19] in Theorem 2.4.3, and then give an adaption of Theorem 2.4.3 for OW secure schemes in Theorem 2.4.4. Afterwards, we show that $\mathsf{T} \circ \mathsf{ACWC}$ achieves OW with bounds that are not significantly worse than T alone. In conclusion, if a scheme is OW secure (and comes with randomness recovery), then applying $\mathsf{FO}_m^{\not\perp} \circ \mathsf{ACWC}$ does not yield significantly worse bounds than applying $\mathsf{FO}_m^{\not\perp}$ alone with respect to the advantage against the underlying scheme.

We believe that our integration technique can be applied to any existing QROM proof for an FO-like transformation $\mathsf{U} \circ \mathsf{T}$ that uses semi-classical one-way to hiding techniques, e.g., [JZM19b, KSS+20]. If the original proof assumes OW security (or can be adapted, accordingly), the bounds will only differ in terms of a search probability, similar to how the bound in Theorem 2.4.5 differs from Theorem 2.4.4. Going over all existing proofs, however, is beyond the scope of this thesis.

Theorem 2.4.3. [BHH+19, Theorem 1] For any adversary A issuing at most q_G (quantum) queries to G, there exists an adversary B such that

$$\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{T}[\mathsf{PKE},\mathsf{G}]}(\mathsf{A}) \leq (q_\mathsf{G} + 2) \cdot (\mathrm{Adv}^{\mathsf{IND}\text{-}\mathsf{CPA}}_{\mathsf{PKE}}(\mathsf{B}) + \frac{8q_\mathsf{G} + 1}{|\mathcal{M}|}) \ ,$$

and the running time of B is about that of A.

We now show that OW security can be achieved from OW instead of $\mathsf{IND}\text{-}\mathsf{CPA}$ security, at the cost of losing the factor $4q_\mathsf{G}$. Since the proof step for $\mathsf{U}_m^{\not\perp}[\mathsf{PKE}', \mathsf{H}]$ in [BHH+19] relates the $\mathsf{IND}\text{-}\mathsf{CCA}$ advantage to the square root of the OW advantage against PKE', the combined $\mathsf{IND}\text{-}\mathsf{CCA}$ bound loses a factor of $2\sqrt{q_\mathsf{G}}$ when assuming OW instead of $\mathsf{IND}\text{-}\mathsf{CPA}$.

Theorem 2.4.4. For any adversary A issuing at most q_{G} (quantum) queries to G, there exist an adversary B_{OW} such that

$$\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{T}[\mathsf{PKE},\mathsf{G}]}(\mathsf{A}) \leq 4q_{\mathsf{G}} \cdot (q_{\mathsf{G}} + 2) \cdot (\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B})) \enspace ,$$

and the running time of B is about that of A.

Proof. In the proof of [BHH$^+$19, Theorem 1], it was shown that

$$\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{T}[\mathsf{PKE},\mathsf{G}]}(\mathsf{A}) \leq (q_{\mathsf{G}} + 2) \cdot \Pr[\mathrm{FIND}_{m^*}] \enspace .$$

Unlike an IND-CPA reduction, an OW reduction does not know the challenge plaintext(s). Since it cannot simulate the punctured oracle $\mathsf{G} \setminus \{\mathsf{m}^*\}$, it will instead provide access to G, pick a query at random, measure its input register and return the outcome as its one-way solution. More formally, we can apply Equation (1.4) of Theorem 1.3.4 to obtain that there exists a OW adversary B_{OW} such that

$$\Pr[\mathrm{FIND}_{m^*}] \leq 4q_{\mathsf{G}} \cdot \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}_{\mathsf{OW}}) \enspace .$$

□

As a relatively simple example for how ACWC can be integrated into former QROM proofs, we will now show how to integrate ACWC into the proof for [BHH$^+$19, Theorem 1] such that the bound does not get significantly worse than the bound given in Theorem 2.4.4. Intuitively, it only differs from the bound in Theorem 2.4.4 in terms of a search probability for the masking value $\tilde{m}^*$.

Theorem 2.4.5. Assume PKE_0 to come with invertible encryption. For any adversary A issuing at most q_{G} (quantum) queries to G and q_{F} (quantum) queries to F, there exists an adversary B such that

$$\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{T}[\mathsf{ACWC}[\mathsf{PKE}_0,\mathsf{F}],\mathsf{G}]}(\mathsf{A}) \leq 4q_{\mathsf{F}} \cdot (q_{\mathsf{RO}} + 2) \cdot \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}) + \frac{4(q_{\mathsf{G}} + 1)(q_{\mathsf{RO}} + 2)}{|\mathcal{M}_0|} \enspace ,$$

where q_{RO} counts the number of all queries to G and F, and the running time of B is about that of A.

Proof. Let F' be the oracle such that $\mathsf{F}'(r^*)$ is sampled uniformly at random, but coincides with F anywhere but on r^*. Similarly, let G' be reprogrammed randomly on m^*, and coincides with G everywhere else. We claim that

$$\Pr[\mathsf{A} \text{ wins }] \leq (q_{\mathsf{G}} + q_{\mathsf{F}} + 2) \cdot \Pr[\mathrm{FIND}_{m^*} \vee \mathrm{FIND}_{r^*}] \enspace , \tag{2.17}$$

where FIND_{m^*} denotes the event that the input register of one of A's queries to $\mathsf{G}' \setminus \{\mathsf{m}^*\}$ collapsed to m^*, and FIND_{r^*} denotes the event that the input register of one of A's queries to $\mathsf{F}' \setminus \{\mathsf{r}^*\}$ collapsed to r^*. In order to verify this claim, note that we can replace A with an adversary A_1 that executes A, but before returning its one-way guess m', it first queries G on message m' to receive $r' = \mathsf{G}(m')$. Since A_1 wins with the same probablity as A,

$$\Pr[\mathsf{A} \text{ wins }] = \Pr[\mathsf{A}_1 \text{ wins }] \ .$$

But if A_1 wins, then A_1 also triggered FIND_{m^*}, hence

$$\Pr[\mathsf{A}_1 \text{ wins } \wedge \neg\mathrm{FIND}_{r^*} \wedge \neg\mathrm{FIND}_{m^*}] = 0 \ .$$

Applying Equation (1.3) and then Equation (1.1) of Theorem 1.3.3, we obtain

$$\begin{aligned}\sqrt{\Pr[\mathsf{A}_1 \text{ wins }]} &= |\sqrt{\Pr[\mathsf{A}_1 \text{ wins }\ ]} - \sqrt{\Pr[\mathsf{A}_1 \text{ wins } \wedge \neg\mathrm{FIND}_{m^*} \wedge \neg\mathrm{FIND}_{r^*}]}| \\ &\leq \sqrt{(q_\mathsf{G} + q_\mathsf{F} + 2) \cdot \Pr[\mathrm{FIND}_{m^*} \vee \mathrm{FIND}_{r^*} : {\mathsf{A}_1}^{|\mathsf{G}\rangle,|\mathsf{F}\rangle}]} \\ &\leq \sqrt{(q_\mathsf{G} + q_\mathsf{F} + 2) \cdot \Pr[\mathrm{FIND}_{m^*} \vee \mathrm{FIND}_{r^*} : {\mathsf{A}_1}^{|\mathsf{G}'\rangle,|\mathsf{F}'\rangle}]} \ .\end{aligned}$$

(Note that we can define a wrapper oracle distinguisher that simulates the game to A_1 and issues exactly as many oracle queries as A_1, i.e., as many queries as A plus one additional query to G.) Squaring the inequality hence yields the bound claimed in Equation (2.17).

To further upper bound the right-hand side, notice that

$$\begin{aligned}\Pr[\mathrm{FIND}_{m^*} \vee \mathrm{FIND}_{r^*} : {\mathsf{A}_1}^{|\mathsf{G}'\rangle,|\mathsf{F}'\rangle}] &= \Pr[\mathrm{FIND}_{m^*} \vee \mathrm{FIND}_{r^*} \text{ in game } G_0] \\ &\leq \Pr[\mathrm{FIND}_{m^*} \text{ in game } G_0] + \Pr[\mathrm{FIND}_{r^*} \text{ in game } G_0] \ ,\end{aligned}$$

where game G_0 is given in Figure 2.45.

In game G_0, A has access to G' and F' instead of G and F, we can hence replace r^* and $\tilde{m}^*$ with uniformly random values, yielding game G_1. Now that m^* is masked by a uniformly random message $\tilde{m}^*$ which is completely independent of A's view, we can replace $M^* := m^* \oplus \tilde{m}^*$ with uniformly random, yielding game G_2. Both changes do not cause any change in A's view.

In game G_2, A's input is independent of m^*, we can hence apply Equation (1.5) of

```
GAMES $G_0$ - $G_2$
$(pk, sk) \leftarrow \mathsf{KG}_0$
$m^* \leftarrow_\$ \mathcal{M}$
$r^* := \mathsf{G}(m^*)$                                  //$G_0$
$\tilde{m}^* := \mathsf{F}(r^*)$                                //$G_0$
$(r^*, \tilde{m}^*) \leftarrow_\$ \mathcal{R} \times \mathcal{M}$                //$G_1$-$G_2$
$M^* := m^* \oplus \tilde{m}^*$                       //$G_0$ - $G_1$
$M^* \leftarrow_\$ \mathcal{M}$                                  //$G_2$
$c^* := \mathsf{Enc}_0(pk, M^*; r^*)$
$m' \leftarrow \mathsf{A}^{|\mathsf{G}' \setminus \{\mathsf{m}^*\}\rangle, |\mathsf{F}' \setminus \{\mathsf{r}^*\}\rangle}(pk, c^*)$
Query $\mathsf{G}' \setminus \{\mathsf{m}^*\}$ on $m'$
return $[\![m' = m^*]\!]$
```

```
$\mathsf{B}(pk, c^*)$
$m^* \leftarrow_\$ \mathcal{M}$
$i \leftarrow_\$ \{1, \cdots, q_\mathsf{F}\}$
Run $\mathsf{A}^{|\mathsf{G}' \setminus \{\mathsf{m}^*\}\rangle, |\mathsf{F}'\rangle}(pk, c^*)$
    until its $i$-th query to $\mathsf{F}'$
$r' \leftarrow$ Measure query input register
$M' := \mathsf{Inv}(pk, c^*, r')$
return $M'$
```

Fig. 2.45: Games G_0 - G_2 and adversary B for the proof of Theorem 2.4.5.

Theorem 1.3.4 to obtain

$$\Pr[\mathrm{FIND}_{m^*} \text{ in game } G_2] \leq \frac{4(q_\mathsf{G}+1)}{|\mathcal{M}|} \ .$$

Furthermore, we can apply Equation (1.4) of Theorem 1.3.4 (with respect to oracle F) to obtain that there exists a one-way adversary B such that

$$\Pr[\mathrm{FIND}_{r^*} \text{ in game } G_2] \leq 4q_\mathsf{F} \cdot \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}) \ ,$$

as measuring a random query to F and finding r^* directly translates to finding M^*. Since m^* is completely independent of A's input, A's view can be perfectly simulated by the OW adversary B given in Figure 2.45: B can simulate G and F with a $2q_\mathsf{G}$-wise independent function, and puncture G with respect to a message m^* of its own choosing. If B's measurement outcome is r^*, then B wins.

Combining the bounds into one yields

$$\Pr[\mathrm{FIND}_{m^*} \vee \mathrm{FIND}_{r^*} : \mathsf{A_1}^{|\mathsf{G}'\rangle, |\mathsf{F}'\rangle}] \leq 4q_\mathsf{F} \cdot \mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}}(\mathsf{B}) + \frac{4(q_\mathsf{G}+1)}{|\mathcal{M}|} \ .$$

□

The same strategy carries over to proving that $\mathsf{TPunc}[\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}], \mathsf{G}]$ is simulatable.

Theorem 2.4.6 (DS of $\mathsf{TPunc} \circ \mathsf{ACWC}$)**.** Assume PKE_0 to come with invertible encryption. For any adversary A issuing at most q_G (quantum) queries to G and q_F (quantum) queries to F, there exists an adversary B such that

$$\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{TPunc}[\mathsf{ACWC}[\mathsf{PKE}_0,\mathsf{F}],\hat{m},\mathsf{G}]}(\mathsf{A}) \leq 4 \cdot (q_{\mathsf{RO}} + 1) \cdot (\sqrt{\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}_0}(\mathsf{B})} + \frac{1}{\sqrt{|\mathcal{M}_0| - 1}} + \frac{1}{\sqrt{|\mathcal{R}_0|}}) \ ,$$

where q_{RO} counts the number of all queries to G and F, and the running time of B is about that of A.

Proof. In the DS game, the adversary either obtains a challenge $c_0^* := \mathsf{Enc}_0(pk, m_0 \oplus \mathsf{F}(\mathsf{G}(m_0)); \mathsf{G}(m_0))$ for a random message from $\mathcal{M}_0 \backslash \{\hat{m}\}$, or a challenge $c_1^* := \mathsf{Enc}_0(pk, \hat{m} \oplus \mathsf{F}(\hat{r}); \hat{r})$ for uniform randomness $\hat{r}$.

Using Theorem 1.3.3, we can decouple c_0^* from $\mathsf{G}(m_0)$ and $\mathsf{F}(\mathsf{G}(m_0))$, and c_1^* from $\mathsf{F}(\hat{r})$. After this change, both ciphertexts are random encryptions of random plaintexts and hence perfectly indistinguishable, and it hence suffices to upper bound the probability that FIND occurs for the randomness that was used to encrypt either of the ciphertexts, or for message m_0:

$$\mathrm{Adv}^{\mathsf{DS}}_{\mathsf{TPunc}[\mathsf{ACWC}[\mathsf{PKE}_0,\mathsf{F}],\hat{m},\mathsf{G}]}(\mathsf{A}) \leq 2 \cdot \sqrt{(q_{\mathsf{RO}} + 1) \cdot p_{\mathrm{FIND}}} \ .$$

With the redefined oracles, m_0 is now independent of A's view, and so is the randomness that was not used in A's challenge ciphertext. Applying both equations from Theorem 1.3.4, we obtain that there exists an adversary B such that

$$p_{\mathrm{FIND}} \leq 4 q_{\mathsf{RO}} \cdot (\mathrm{Adv}^{\mathsf{OW}}_{\mathsf{PKE}_0}(\mathsf{B}) + \frac{1}{|\mathcal{M}_0| - 1} + \frac{1}{|\mathcal{R}_0|}) \ .$$

□

2.4.4 $\mathsf{FO}^{\not\perp}_m \circ \mathsf{ACWC}$: *From* OW *to* $\mathsf{IND}\text{-}\mathsf{CCA}$*, in the QROM*

As a corollary, we obtain that transformation $\mathsf{FO}^{\not\perp}_m \circ \mathsf{Punc} \circ \mathsf{ACWC}$ achieves $\mathsf{IND}\text{-}\mathsf{CCA}$ security from one-wayness.

Corollary 2.4.7 ($\mathsf{IND}\text{-}\mathsf{CCA}$ security of $\mathsf{FO}^{\not\perp}_m \circ \mathsf{Punc}$.)**.** Assume PKE_0 to be δ_0-average-case correct, γ-spread, and to come with invertible encryption. Let $\hat{m} \in \mathcal{M}_0$. Let $\mathsf{KEM} := \mathsf{FO}^{\not\perp}_m[\mathsf{Punc}[\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}], \hat{m}], \mathsf{G}, \mathsf{H}]$, where $\mathsf{H} : \{0,1\}^* \to \{0,1\}^\ell$. The algorithms of KEM are made explicit in Figure 2.46. Then, for any (quantum) $\mathsf{IND}\text{-}\mathsf{CCA}$ adversary A issuing at most q_D (classical) queries to the decapsulation oracle $\textsc{Dec}^{\not\perp}_m$ and at most q_{RO} quantum queries to H and G, there exists an adversary B against the

```
$\mathsf{KG}^{\not\perp}$
    $(pk, sk) \leftarrow \mathsf{KG}$
    $s \leftarrow_\$ \mathcal{M}$
    $sk' := (sk, s)$
    return $(pk, sk')$

$\mathsf{Encaps}(pk)$
    $m \leftarrow_\$ \mathcal{M}_0 \setminus \{\hat{m}\}$
    $c := \mathsf{Enc}(pk, m \oplus \mathsf{F}(\mathsf{G}(m)); \mathsf{G}(m))$
    $K := \mathsf{H}(m)$
    return $(K, c)$

$\mathsf{Decaps}_m^{\not\perp}(sk', c)$
    Parse $(sk, s) := sk'$
    $M' := \mathsf{Dec}_0(sk, c)$
    $r' := \mathsf{Rec}(sk, M', c)$
    $m' := M' \oplus \mathsf{F}(r')$
    if $M' = \perp$
    or $c \neq \mathsf{Enc}(pk, m' \oplus \mathsf{F}(\mathsf{G}(m')); \mathsf{G}(m'))$
        return $K := \mathsf{H}(s, c)$
    else
        return $K := \mathsf{H}(m')$
```

Fig. 2.46: Key encapsulation mechanism $\mathsf{KEM} := \mathsf{FO}_m^{\not\perp}[\mathsf{Punc}[\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}], \hat{m}], \mathsf{G}, \mathsf{H}]$.

OW security of PKE_0 such that

$$\begin{aligned}
\mathrm{Adv}_{\mathsf{KEM}}^{\mathsf{IND\text{-}CCA}}(\mathsf{A}) \leq\ & 4 \cdot (q_{\mathsf{RO}} + 1) \cdot \sqrt{\mathrm{Adv}_{\mathsf{PKE}_0}^{\mathsf{OW}}(\mathsf{B})} + 24 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \delta_0 \\
& + \frac{4}{\sqrt{|\mathcal{R}_0|}} \cdot (q_{\mathsf{RO}} + 1 + 6 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \sqrt{\log(\mathcal{M}_0)}) \\
& + 24 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot |\mathcal{M}_0|^{-1.88} \\
& + \frac{4 \cdot ((q_{\mathsf{RO}} + 1)}{\sqrt{|\mathcal{M}_0| - 1}} + 2^{-\gamma} + q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}} \quad ,
\end{aligned}$$

and the running time of B is about that of A.

Proof. In order to verify the claimed upper bound, we revisit the proof of Theorem 2.3.6 (page 118). The only difference between theorem Theorem 2.3.6 and Corollary 2.4.7 is that we now plugged in transformation ACWC. In the proof of Theorem 2.3.6, it was implicitly shown that there exists an adversary B' against the simulatability of $\mathsf{PKE} := \mathsf{TPunc}[\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}], \hat{m}, \mathsf{G}]$ such that

$$\begin{aligned}
\mathrm{Adv}_{\mathsf{KEM}}^{\mathsf{IND\text{-}CCA}}(\mathsf{A}) \leq\ & \mathrm{Adv}_{\mathsf{PKE}}^{\mathsf{DS}}(\mathsf{B}') + 24 \cdot (q_{\mathsf{RO}} + q_D + 2)^2 \cdot \delta \\
& + \frac{4 q_{\mathsf{RO}}}{\sqrt{|\mathcal{M}| - 1}} + 2^{-\gamma} + q_{\mathsf{H}} \cdot 2^{\frac{-\ell+1}{2}} \quad ,
\end{aligned}$$

where δ is the worst-case correctness term for $\mathsf{ACWC}[\mathsf{PKE}_0, \mathsf{F}]$. Applying Theorem 2.4.6 and Theorem 2.4.1 yields the claimed bound. □

2.4.5 Achieving Worst-Case Correctness without Randomness Recovery

Below we give a transformation that achieves worst-case correctness, and that does not require the underlying scheme to come with randomness recovery.

Construction without randomness recovery. To a public-key encryption scheme $\mathsf{PKE}_0 = (\mathsf{KG}_0, \mathsf{Enc}_0, \mathsf{Dec}_0)$ with message space $\mathcal{M}_0 = \{0,1\}^{\ell_1+\ell_2}$, randomness space $\mathcal{R}_0$, and random oracle $\mathsf{F} : \{0,1\}^{\ell_1} \to \{0,1\}^{\ell_2}$, we associate $\mathsf{PKE} := \mathsf{ACWC}'[\mathsf{PKE}_0, \mathsf{F}]$ with message space $\mathcal{M} := \{0,1\}^{\ell_2}$. The algorithms of $\mathsf{PKE} = (\mathsf{KG}_0, \mathsf{Enc}, \mathsf{Dec})$ are defined in Figure 2.47.

```
Enc(pk, m ∈ M)
    $M_1 \leftarrow_{\$} \mathcal{M}$
    $c \leftarrow \mathsf{Enc}(pk, M_1 \| m \oplus \mathsf{F}(M_1))$
    return $c$

Dec(sk, c)
    Parse $M_1' \| M_2' := \mathsf{Dec}(sk, c)$
    return $m' := M_2' \oplus \mathsf{F}(M_1')$
```

Fig. 2.47: Worst-case correct encryption scheme $\mathsf{PKE} = \mathsf{ACWC}'[\mathsf{PKE}_0, \mathsf{F}]$.

With techniques similar to the ones in Section 2.4.1, it can be shown that if the underlying scheme is δ_0-average-case-correct, then PKE is δ-worst-case-correct, where

$$\delta = \delta_0 + \sqrt{\frac{\ell_2}{|\mathcal{R}_0| \cdot 2^{\ell_1}}} + 2^{-1.88 \cdot \ell_2} \quad .$$

A further formal treatment is excluded from this thesis, however, as in order to achieve $\mathsf{IND\text{-}CPA}$ security, the underlying scheme itself is required to satisfy either $\mathsf{IND\text{-}CPA}$ security or partial one-wayness, where the latter means that given the public key and an encryption, it is unfeasible even to recover the first ℓ_1 bits of the engendering plaintext. Since we do not know of any practical schemes that achieve $\mathsf{IND\text{-}CPA}$ security or partial one-wayness, while only achieving average-case correctness, we do not know whether the construction might prove to be of interest.

Chapter 3

Adaptive Reprogramming and its Applications in the QROM

Since its introduction, the ROM has allowed cryptographers to prove practical cryptosystems secure for which proofs in the standard model have been elusive. In general, the ROM allows for proofs that are conceptually simpler and often tighter than standard model security proofs. Unfortunately, the QROM does not generally come with the advantages of its classical counterpart:

- *Lack of conceptual simplicity.* QROM proofs can turn out extremely complex for various reasons, with one reason being that they require at least some understanding of quantum information theory. More important, however, is the fact that many of the useful properties of the ROM (like preimage awareness and adaptive programmability) are not known to translate directly to the QROM.
- *Tightness.* Many primitives that come with tight security proofs in the ROM are not known to be supported by tight proofs in the QROM. As discussed in chapter 2, there has been an ongoing effort to give tighter QROM proofs for FO-like transformations.

In many cases, we expect that certain generic attacks only differ from their ROM counterparts by a square-root factor in the required number of queries if the attack involves a search problem, or no significant factor in the case of guessing. Hence, it was conjectured that it might be sufficient to prove security in the ROM, and then to simply to add a square-root factor for search problems. However, recent results [YZ20] demonstrate a separation of ROM and QROM, showing that this conjecture does not hold true in general, as there exist schemes which are provably secure in the ROM and insecure in the QROM. As a consequence, giving a proof in the QROM is crucial to establish confidence in a post-quantum cryptosystem.[1]

Adaptive programmability. A desirable property of the (classical) ROM is that any oracle value $\mathsf{O}(x)$ can be chosen when O is queried on x for the first time (lazy-sampling). This fact is often exploited by a reduction simulating a security game without knowledge of some secret information. As an example in which we used this property to simulate a decryption oracle, recall sections 2.1.2 and 2.1.3 (pages 68 to 80). Another important example is the security proof for the Fiat-Shamir transform: To achieve $\mathsf{UF\text{-}CMA}$ from $\mathsf{UF\text{-}CMA}_0$ and HVZK, a reduction can simulate the signing oracle by generating a HVZK transcript (w, c, z), and returning (w, z) as the signature. In order to maintain consistency, however, the reduction has to reprogram $\mathsf{H}(m, w) := c$. In the classical ROM, A will not recognise the reprogramming of $\mathsf{O}(x)$ as long as the new value is uniformly distributed and consistent with the rest of A's view. This property is called *adaptive programmability.*

The ability to query an oracle in superposition renders this formerly simple approach more involved, similar to the difficulties arising from the question how to extract classical preimages from a quantum query (preimage awareness) [Unr14b, AHU19, BHH$^+$19, KSS$^+$20, Zha19, DFMS19, LZ19, BL20, CMP20]. Intuitively, a query in superposition can be viewed as a query that might contain all input values at once. Already the first answer of O might hence contain information about some value $\mathsf{O}(x)$ that needs to be reprogrammed during the proceedings of the game. It hence was not clear whether it is possible to adaptively reprogram a quantum random oracle without causing a change in the adversary's view.

Until recently, both properties only had extremely non-tight variants in the QROM. For preimage awareness, it was essentially necessary to randomly guess the right query and measure it (with an unavoidable loss of at least $1/q$ for q queries, and the additional disadvantage of potentially rendering the adversary's output unusable due to measurement disturbance). In a recent breakthrough result, Zhandry developed the compressed oracle technique that provides preimage awareness [Zha19] in many settings. For adaptive reprogramming, variants of Unruh's one-way-to-hiding lemma allowed to prove bounds but only with a square-root loss in the entropy of the reprogramming position [Unr14a, ES15, HRS16].

In some cases (e.g., [BDF$^+$11, KLS18, SXY18]), reprogramming was avoided altogether by giving a proof that rendered the oracle "a-priori consistent", which is also called a "history-free" proof: In this approach, the oracle is completely redefined in a

[1] Unless, of course, a standard model proof is available.

way such that it is enforced to be *a priori* consistent with the rest of an adversary's view, meaning that it is redefined before execution of the adversary, and on *all* possible input values. Unfortunately, it is not always clear whether it is possible to lift a classical proof to the QROM with this strategy. Even if it is, the "a-priori" approach usually leads to conceptually more complicated proofs. More importantly, it can even lead to reductions that are non-tight with respect to runtime, and may necessitate stronger or additional requirements like, e.g., the statistical counterpart of a property that was only used in its computational variant in the ROM. An example is the proof of UF-CMA security for Fiat-Shamir signatures that was given in [KLS18].

Hence, in this chapter we are interested in the question:

> **Can we *tightly* prove that adaptive reprogramming can also be done in the quantum random oracle model?**

For common use cases in the context of post-quantum cryptography, we answer the question above in the affirmative. In more detail, in 3.1 we present a tool for adaptive reprogramming that comes with a tight bound, supposing that the reprogramming positions hold sufficiently large entropy, and reprogramming is triggered by classical queries to an oracle that is provided by the security game (e.g., a signing oracle). These preconditions are usually met in (Q)ROM reductions: The reprogramming is usually triggered by adversarial signature or decryption queries, which remain classical in the post-quantum setting, as the oracles represent honest users.

Using the simplest variant of the superposition oracle technique [Zha19], we prove a very general theorem which we call the "adaptive reprogramming" (AR) theorem. From our AR theorem, we also derive a corollary that is tailored to cases like Fiat-Shamir signatures (or hash-and-sign with randomised hashing). In this case, reprogramming occurs at a position of which one part is an adversarially chosen string. The other part is a commitment w chosen from a distribution with sufficient min-entropy. We manage to bound the distinguishing advantage of any adversary that makes q_S signing and q random oracle queries by

$$1.5 \cdot q_S \sqrt{q \cdot 2^{-r}} \ ,$$

where r is the min-entropy of w.

We then demonstrate the applicability of our tool, by giving

- a runtime-tight reduction of UF-CMA to plain unforgeability (UF-CMA$_0$) for Fiat Shamir signatures, and
- the first proof of fault resistance for the hedged Fiat-Shamir transform, recently proposed in [AOTZ20], in the post-quantum setting.

The Fiat-Shamir transform. In Section 3.2, we show that if an identification scheme ID is Honest-Verifier Zero-Knowledge (HVZK), and if the resulting Fiat-Shamir signature scheme $\mathsf{SIG} := \mathsf{FS}[\mathsf{ID}, \mathsf{H}]$ furthermore possesses $\mathsf{UF\text{-}CMA}_0$ security, then SIG is also $\mathsf{UF\text{-}CMA}$ secure, in the quantum random oracle model. Here, $\mathsf{UF\text{-}CMA}_0$ denotes the security notion in which the adversary only obtains the public key and has to forge a valid signature without access to a signing oracle. While this statement was already proven in [KLS18], we want to point out several advantages of our proof strategy and the resulting bounds.

Conceptual simplicity. A well-known proof strategy for $\mathsf{HVZK}, \mathsf{UF\text{-}CMA}_0 \Rightarrow \mathsf{UF\text{-}CMA}$ in the random oracle model (implicitly contained in [AFLT12]) is to replace honest transcripts with simulated ones, and to render H *a-posteriori* consistent with the signing oracle during the proceedings of the game. I.e., $\mathsf{H}(w, m)$ is patched *after* oracle SIGN was queried on m. Applying our lemma, we observe that this approach actually works in the quantum setting as well. We obtain a very simple QROM proof that is congruent with its ROM counterpart.

In [KLS18], the issue of reprogramming quantum random oracle H was circumvented by giving a history-free proof: In the proof, messages are tied to potential transcripts by generating the latter with message-dependent randomness, *a priori*, and H is patched accordingly, right from the beginning of the game. During each computation of $\mathsf{H}(w, m)$, the reduction therefore has to keep H a-priori consistent by going over all transcript candidates (w_i, c_i, z_i) belonging to m, and returning c_i if $w = w_i$.

Tightness with regards to running time. Our reduction B has about the running time of the adversary A, as it can simply sample simulated transcripts and reprogram H, accordingly. The reduction in [KLS18] suffers from a quadratic blow-up in its running time: They have running time $\mathrm{Time}(\mathsf{B}) \approx \mathrm{Time}(\mathsf{A}) + q_{\mathsf{H}} q_S$, as the reduction has to execute q_S computations upon each query to H in order to keep it a-priori consistent. As they observe, this quadratic blow-up renders the reduction non-tight in all practical aspects. On the other hand, our upper bound of the advantage comes with a bigger disruption in terms of commitment entropy (the min-entropy of the first message (the *commitment*) in the identification scheme). While the source of non-tightness in [KLS18] can not be balanced out, however, we offer a trade-off: If needed, the commitment entropy can be increased by appending a random string to the commitment.[2]

Generality. To achieve a-priori consistency, [KLS18] crucially relies on *statistical*

[2] While this increases the signature size, the increase is mild in typical post-quantum Fiat-Shamir based digital signature schemes. As an example, suppose Dilithium-1024x768, which has a signature size of 2044 bytes, had zero commitment entropy (it actually has quite some, see remarks in [KLS18]). To ensure that about 2^{128} hash queries are necessary to make the term in our security bound that depends on the commitment entropy equal 1, about 32 bytes would need to be added, an increase of about 1.6% (assuming 2^{64} signing queries).

HVZK. Furthermore, they require that the HVZK simulator outputs transcripts such that the challenge c is uniformly distributed. We are able to drop the requirement on c altogether, and to only require *computational* HVZK. (As a practical example, alternate NIST candidate Picnic [ZCD+19] satisfies only *computational* HVZK.)

Robustness of the hedged Fiat-Shamir transform against fault attacks. When it comes to real-world implementations, the assessment of a signature scheme will not solely take into consideration whether an adversary could forge a fresh signature as formalised by the UF-CMA game, as the UF-CMA definition does not capture all avenues of real-world attacks. For instance, an adversary interacting with hardware that realises a cryptosystem can try to induce a hardware malfunction, also called fault injection, in order to derail the key generation or signing process. Although it might not always be straightforward to predict where exactly a triggered malfunction will affect the execution, it is well understood that even a low-precision malfunction can seriously injure a schemes' security. In the context of the ongoing effort to standardise post-quantum secure primitives [NIS17], it hence made sense to affirm [NIS20] that desirable additional security features include, amongst others, resistance against fault attacks and randomness generation that has some bias.

Very recently [AOTZ20], the hedged Fiat-Shamir construction was proven secure against biased nonces and several types of fault injections, in the ROM. This result can for example be used to argue that alternate NIST candidate Picnic [ZCD+19] is robust against many types of fault injections. We revisit the hedged Fiat-Shamir construction in Section 3.3 and lift the result of [AOTZ20] to the QROM. In particular, we thereby obtain that Picnic is resistant against many fault types, even when attacked by an adversary with quantum capabilities.

We considered to generalise the result further by replacing the standard Fiat-Shamir transform with the Fiat–Shamir with aborts transform that was introduced by Lyubashevsky [Lyu09, KLS18]. Recall that Fiat–Shamir with aborts was established due to the fact that for some underlying lattice-based ID schemes (e.g., NIST finalist Dilithium [DKL+18]), the prover sometimes cannot create a correct response to the challenge, and the protocol therefore allows for up to κ many retries during the signing process. While our security statements can be extended in a straightforward manner, we decided not to further complicate our proof with the required modifications. For Dilithium, the implications are limited anyway, as several types of faults are only proven ineffective if the underlying scheme is subset-revealing, which Dilithium is not.[3]

[3] Intuitively, an identification scheme is called subset-revealing if its responses do not depend on the secret key. Dilithium computes its responses as $z := y + c \cdot s_1$, where s_1 is part of the secret key.

3.1 Adaptive Reprogramming: The Theorem

During security proofs, we often need to reprogram some random oracle O several times, adaptively: O needs to be reprogrammed after it has been queried already. Often, the positions x at which O is reprogrammed are partially fixed by the adversary, and partially sampled according to some distribution: As a motivating example, we look ahead to Section 3.2, in which we give a reduction that simulates the $\mathsf{UF\text{-}CMA}$ game for a Fiat-Shamir transformed scheme. We will see that a signing query on a message m results in the need to reprogram an oracle H on some tuple (m, w), where $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$ are freshly sampled by the reduction and then are used to generate the signature.

We will now formalise how to distinguish a random oracle $\mathsf{O} : X \to Y$ from its reprogrammed version.

As a warm-up, we will first present our reprogramming lemma in the simplest setting. Say we reprogram an oracle R many times, where the position is partially controlled by the adversary, and partially picked at random. More formally, let X_1 and X_2 be two finite sets, where X_1 specifies the domain from which the random portions are picked, and X_2 specifies the domain of the adversarially controlled portions. We will now formalise what it means to distinguish a random oracle $\mathsf{O}_0 : X_1 \times X_2 \to Y$ from its reprogrammed version O_1. Consider the two Repro games, given in Figure 3.1: In games Repro$_b$, the distinguisher has quantum access to oracle $\mathsf{O_b}$ (see line 03) that is either the original random oracle O_0 (if $b = 0$), or the oracle O_1 which gets reprogrammed adaptively ($b = 1$). To model the actual reprogramming, we endow the distinguisher with (classical) access to a reprogramming oracle Reprogram. Given a value $x_2 \in X_2$, oracle Reprogram samples random values x_1 and y, and programs the random oracle to map $x_1 \| x_2$ to y (see line 06). Note that apart from already knowing x_2, the adversary even learns the part x_1 of the position at which O_1 was reprogrammed.

```
GAME Repro_b
01 $\mathsf{O}_0 \leftarrow_\$ Y^{X_1 \times X_2}$
02 $\mathsf{O}_1 := \mathsf{O}_0$
03 $b' \leftarrow \mathsf{A}^{|\mathsf{O_b}\rangle, \textsc{Reprogram}}$
04 return $b'$

Reprogram($x_2$)
05 $(x_1, y) \leftarrow_\$ X_1 \times Y$
06 $\mathsf{O}_1 := \mathsf{O}_1^{(x_1 \| x_2) \mapsto y}$
07 return $x_1$
```

Fig. 3.1: Adaptive reprogramming games Repro$_b$ for bit $b \in \{0, 1\}$ in the most basic setting.

Theorem 3.1.1. Let X_1, X_2 and Y be finite sets, and let A be any algorithm issuing R many calls to Reprogram and q many (quantum) queries to $\mathsf{O_b}$ as defined in Figure 3.1.

Then the distinguishing advantage of A is bounded by

$$|\Pr[\textsc{Repro}_1^{\mathsf{A}} \Rightarrow 1] - \Pr[\textsc{Repro}_0^{\mathsf{A}} \Rightarrow 1]| \leq \frac{3R}{2}\sqrt{\frac{q}{|X_1|}}. \tag{3.1}$$

The above theorem constitutes a significant improvement over previous bounds. In [Unr14a] and [ES15], a bound proportional to $q|X_1|^{-1/2}$ was given for the distinguishing advantage in similar settings. This bound, however, only considered the case that $R = 1$. In [HRS16], a bound proportional to $q^2|X_1|^{-1}$ is claimed, but that seems to have resulted from a "translation mistake" from [ES15], and should be similar to the bounds from [Unr14a, ES15].

In fact, we prove something more general than Theorem 3.1.1: We prove that an adversary will not behave significantly different, even if

- the adversary does not only control a portion x_2, but instead it even controls the distributions according to which the whole positions $x := (x_1, x_2)$ are sampled at which O_1 is reprogrammed,
- it can additionally pick different distributions, adaptively, and
- the distributions produce some additional side information x' which the adversary also obtains,

as long as the reprogramming positions x hold enough entropy.

Overloading notation, we formalise this generalisation by games Repro, given in Figure 3.2: Reprogramming oracle Reprogram now takes as input the description of a distribution p that generates a whole reprogramming position x, together with side information x'. Reprogram samples x and x' according to p, programs the random oracle to map x to a random value y, and returns (x, x').

```
GAME Repro_b
    $\mathsf{O}_0 \leftarrow_\$ Y^X$
    $\mathsf{O}_1 := \mathsf{O}_0$
    $b' \leftarrow \mathsf{D}^{|\mathsf{O_b}\rangle, \textsc{Reprogram}}$
    return $b'$

Reprogram(p)
    $(x, x') \leftarrow p$
    $y \leftarrow_\$ Y$
    $\mathsf{O}_1 := \mathsf{O}_1^{x \mapsto y}$
    return $(x, x')$
```

Fig. 3.2: Adaptive reprogramming games $\textsc{Repro}_b$ for bit $b \in \{0, 1\}$.

We are now ready to present our main Theorem 3.1.2. On a high level, the only difference between the statement of Theorem 3.1.1 and Theorem 3.1.2 is that we now have to consider R many (possibly different) joint distributions on $X \times X'$, and to replace $\frac{1}{|X_1|}$ (the probability of the uncontrolled reprogramming portion) with the highest likelihood of any of those distributions generating a position x.

Theorem 3.1.2 ("Adaptive reprogramming" (AR))**.** Let X, X', Y be some finite sets, and let D be any distinguisher, issuing R many reprogramming instructions and q many (quantum) queries to O. Let q_r denote the number of queries to O that are issued inbetween the $(r-1)$-th and the r-th query to Reprogram. Furthermore, let $p^{(r)}$ denote the rth distribution that Reprogram is queried on. By $p_X^{(r)}$ we will denote the marginal distribution of X, according to $p^{(r)}$, and define

$$p_{\max}^{(r)} := \mathbb{E} \max_x p_X^{(r)}(x),$$

where the expectation is taken over D's behaviour until its rth query to Reprogram. Then

$$|\Pr[\textsc{Repro}_1^{\mathsf{D}} \Rightarrow 1] - \Pr[\textsc{Repro}_0^{\mathsf{D}} \Rightarrow 1]| \leq \sum_{r=1}^{R} \left(\sqrt{\hat{q}_r p_{\max}^{(r)}} + \frac{1}{2} \hat{q}_r p_{\max}^{(r)} \right) \quad , \tag{3.2}$$

where $\hat{q}_r := \sum_{i=0}^{r-1} q_i$.

Before we prove Theorem 3.1.2, we will now quickly discuss how to simplify the bound given in Equation (3.2) for our applications in Sections 3.2 and 3.3, and in particular, how we can derive Equation (3.1) from Theorem 3.1.2: Throughout sections 3.2 and 3.3, we will only have to consider reprogramming instructions that occur on positions $x = (x_1, x_2)$ such that

- x_1 is drawn according to the same distribution p for each reprogramming instruction, and

- x_2 represents a message that is already fixed by the adversary.

To be more precise, (x_1, x') will represent a tuple (w, st) that is drawn according to $\mathsf{Commit}(sk)$.

In the language of Theorem 3.1.2, the marginal distribution $p_X^{(r)}$ will always be the same distribution p, apart from the already fixed part x_2. We can hence upper bound $p_{\max}^{(r)}$ by $p_{\max} := \max_{x_1} p(x_1)$, and $\hat{q}_r$ by q, to obtain that $\hat{q}_r p_{\max}^{(r)} < q p_{\max}$ for all $1 \leq r \leq R$.

In our applications, we will always require that p holds sufficiently large entropy. To be more precise, we will assume that $p_{\max} < \frac{1}{q}$. In this case, we have that $q p_{\max} < 1$, and that we can upper bound $q p_{\max}$ by $\sqrt{q p_{\max}}$ to obtain

Corollary 3.1.3. Let X_1, X_2, X' and Y be some finite sets, and let p be a distribution on $X_1 \times X'$. Let D be any distinguisher, issuing q many (quantum) queries to O and R many reprogramming instructions such that each instruction consists of a value x_2, together with the fixed distribution p. Then

$$|\Pr[\textsc{Repro}_1^{\mathsf{D}} \Rightarrow 1] - \Pr[\textsc{Repro}_0^{\mathsf{D}} \Rightarrow 1]| \leq \frac{3R}{2}\sqrt{qp_{\max}} \ ,$$

where $p_{\max} := \max_{x_1} p(x_1)$.

From this we obtain Theorem 3.1.1 setting $p_{max} = |X_1|$.

3.1.1 Proof of Theorem 3.1.2

We now proceed to the proof of Theorem 3.1.2, which we break down into three steps: In Theorem 3.1.4, we first consider the simple special case in which only a single reprogramming instance occurs, and where no additional input x' is provided to the adversary. We then use a standard hybrid argument to generalise Theorem 3.1.4 for multiple reprogramming instances (see Corollary 3.1.5). Finally, we to how to generalise Corollary 3.1.5 for distributions that generate additional input x'. The generalisation is also straightforward, as the achieved bounds are information-theoretical and a reduction can hence compute marginal and conditioned distributions on its own.

Theorem 3.1.4. Let $\mathsf{O_0}$ be a random oracle. Consider a two-stage distinguisher $\mathsf{D} = (\mathsf{D_0}, \mathsf{D_1})$ such that

- the first stage $\mathsf{D_0}$ has trivial input, makes q quantum queries to $\mathsf{O_0}$, and outputs a quantum state $|\psi_{int}\rangle$, together with a sampling algorithm for a probability distribution p on $\{0,1\}^n$,
- the second stage $\mathsf{D_1}$ gets as input $x^* \leftarrow p$ and $|\psi_{int}\rangle$, has quantum access to $\mathsf{O_b}$, where $\mathsf{O_1} := \mathsf{O}_0^{x^* \mapsto y^*}$ for $y^* \leftarrow_\$ \{0,1\}^m$, and outputs a guessing bit b' with the goal that $b' = b$.

Let $p_{\max} := \mathbb{E}[\max_x p(x)]$, where the expectation is taken over $(|\psi_{int}\rangle, p) \leftarrow \mathsf{D}_0^{|\mathsf{O_0}\rangle}$. Then the success probability of any such distinguisher D is bounded by

$$\Pr[b = b'] - \frac{1}{2} \leq \frac{1}{2}\sqrt{qp_{\max}} + \frac{1}{4}qp_{\max}, \tag{3.3}$$

where the probability is taken over $b \leftarrow_\$ \{0,1\}$, $(|\psi_{int}\rangle, p) \leftarrow \mathsf{D}_0^{|\mathsf{O_0}\rangle}$, $x^* \leftarrow p$, $y^* \leftarrow_\$ \{0,1\}^m$ and $b' \leftarrow \mathsf{D}_1^{|\mathsf{O_b}\rangle}(x^*, |\psi_{int}\rangle)$.

In order to prove Theorem 3.1.4, we will use the superposition oracle formalism (see Section 1.3.4, page 47), i.e., we implement $\mathsf{O_b}$ as superposition oracles.

Proof. Let $|\Psi\rangle_{AF}$ be the joint algorithm-oracle-state, right before D_0 finishes by returning a probability distribution p and an internal state $|\psi_{int}\rangle$ that will later be passed on to D_1. Without loss of generality (see Theorem 1.3.1), we can assume that D_0 until now has proceeded by performing a unitary quantum computation to arrive at $|\Psi\rangle_{AF}$, which will only be followed by a measurement to produce the classical output of the distribution p, and the discarding of D_0's internal working registers. We can therefore identify $|\Psi\rangle_{AF}$ with a state $|\gamma\rangle_{RGF}$ such that

- R is the register that contains D_0's output $|\psi_{int}\rangle$, and G is the garbage register that D_0 will discard after finishing, and

- the joint algorithm-oracle-state, *after* D_0 has finished, can be identified with the result of discarding the G-register of $|\gamma\rangle_{RGF}$.

Since the optimal distinguishing advantage between two states can be upper bounded in terms of the trace distance of their density matrices (see Theorem 1.3.7), we want to relate the density matrix of the algorithm-oracle state in the reprogrammed case to that of the state in the non-reprogrammed case, and since the reprogramming happens on a random x^*, we will first fix any x^* and define the density matrices, conditioned on this x^*. (For $|\gamma\rangle_{RGF}$, the density matrix of course is independent of x^*, but we will index it anyways for notational convenience.) Let $\rho_{RGF}^{(0,x^*)} := |\gamma\rangle\langle\gamma|_{RGF}$ be the density matrix of $|\gamma\rangle_{RGF}$, and let $\rho_{RGF}^{(1,x^*)}$ denote the density matrix of the algorithm-oracle-state, after D_0 has finished and the oracle has been reprogrammed at x^*, had D_0 not discarded the garbage register.

The distinguisher's second stage, D_1, has arbitrary query access to the oracle O_b. In the superposition oracle framework, that means in other words that D_1 can apply arbitrary unitary operations on its internal registers, including its input register R, interspersed with applications of the oracle unitary O_{XYF} from Section 1.3.4 on some internal registers XY and the oracle register F. We bound the success probability by allowing arbitrary operations on F, thus reducing the oracle distinguishing task to the task of distinguishing the quantum states $\rho_{RF}^{(b,x^*)} := \mathrm{Tr}_G \rho_{RGF}^{(b,x^*)}$ for $b = 0, 1$. (Recall that G is the garbage register that was discarded by D_0, and hence is "traced out".)

For any fixed x^*, we can now use Theorem 1.3.7 to bound

$$\Pr[b = b'] - \frac{1}{2} \leq \frac{1}{4} \left\| \rho_{RF}^{(0,x^*)} - \rho_{RF}^{(1,x^*)} \right\|_1 \leq \frac{1}{4} \left\| \rho_{RGF}^{(0,x^*)} - \rho_{RGF}^{(1,x^*)} \right\|_1 \quad ,$$

where the second inequality uses that the trace distance is non-increasing under partial trace. Taking the expectation over $(|\psi_{int}\rangle, p) \leftarrow \mathsf{D}_0^{|\mathsf{O}_0\rangle}$and $x^* \leftarrow p$, and applying

Jensen's inequality, we obtain

$$\Pr[b=b'] - \frac{1}{2} \le \frac{1}{4} \cdot \mathbb{E} \left\| \rho^{(0,x^*)}_{RGF} - \rho^{(1,x^*)}_{RGF} \right\|_1 \ . \tag{3.4}$$

In order to upper bound the trace distance, we will now examine how the two density matrices are related to each other, conditioned on a fixed x^*. We can use basic linear algebra to dissect the density matrix $\rho^{(0,x^*)}_{RGF} := |\gamma\rangle\langle\gamma|_{RGF}$ into

$$\begin{aligned}
\rho^{(0,x^*)}_{RGF} =& \rho^{(0,x^*)}_{RGF} \left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}} + |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right) \\
=& \rho^{(0,x)}_{RGF} \left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right) + \langle\phi_0|_{F_{x^*}} \rho^{(0,x^*)}_{RGF} |\phi_0\rangle_{F_{x^*}} \otimes |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \\
&+ \left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right) \rho^{(0,x)}_{RGF} |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \ .
\end{aligned} \tag{3.5}$$

In the superposition oracle framework, we can formalise the reprogramming of O_0 at x^* with an independent uniformly random output by replacing the contents of the register F_{x^*} with a fresh uniform superposition $|\phi_0\rangle_{F_{x^*}}$. We can hence write the density matrix $\rho^{(1,x^*)}_{RGF}$ as

$$\begin{aligned}
\rho^{(1,x^*)}_{RGF} =& \mathrm{Tr}_{F_{x^*}}[\rho^{(1,x^*)}_{RGF}] \otimes |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \\
=& \langle\phi_0|_{F_{x^*}} \rho^{(0,x^*)}_{RGF} |\phi_0\rangle_{F_{x^*}} \otimes |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \\
&+ \mathrm{Tr}_{F_{x^*}}[(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}) \rho^{(0,x^*)}_{RGF}] \otimes |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \ ,
\end{aligned} \tag{3.6}$$

where the first equality again stems from basic linear algebra, and the second equality can be verified by computing the partial trace in an orthonormal basis containing $|\phi_0\rangle$.

Using Equations (3.5) and (3.6) and the triangle inequality, we bound

$$\begin{aligned}
&\left\| \rho^{(0,x^*)}_{RGF} - \rho^{(1,x^*)}_{RGF} \right\|_1 \\
&= \left\| \rho^{(0,x^*)}_{RGF} \left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right) + \left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right) \rho^{(0,x^*)}_{RGF} |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \right. \\
&\qquad \left. - \mathrm{Tr}_{F_{x^*}}[(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}) \rho^{(0,x^*)}_{RGF}] \otimes |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \right\|_1
\end{aligned} \tag{3.7}$$

$$\begin{aligned}
&\le \left\| \rho^{(0,x^*)}_{RGF} \left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right) \right\|_1 + \left\| \left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right) \rho^{(0,x^*)}_{RGF} |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \right\|_1 \\
&\qquad + \left\| \mathrm{Tr}_{F_{x^*}}[(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}) \rho^{(0,x^*)}_{RGF}] \otimes |\phi_0\rangle\langle\phi_0|_{F_{x^*}} \right\|_1 \ .
\end{aligned} \tag{3.8}$$

The trace norm of a positive semidefinite matrix is equal to its trace, so the last term of Equation (3.8) can be simplified as

$$\left\|\mathrm{Tr}_{F_{x^*}}[(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}})\rho_{RGF}^{(0,x^*)}] \otimes |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right\|_1 = \mathrm{Tr}[(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}})\,|\gamma\rangle\langle\gamma|_{RGF}] \\ = 1 - \delta_{x^*} \ , \tag{3.9}$$

where

$$\delta_{x^*} := \left\|\langle\phi_0|_{F_{x^*}}\,|\gamma\rangle_{RGF}\right\|^2 \ .$$

By Hölder's inequality, the second term of Equation (3.8) can be upper bounded by the first, which we can simplify as

$$\begin{aligned}\|\rho_{RGF}^{(0,x^*)}\left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right)\|_1 &= \|\,|\gamma\rangle\langle\gamma|_{RGF}\left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right)\|_1 \\ &= \|\left(\mathbb{1} - |\phi_0\rangle\langle\phi_0|_{F_{x^*}}\right)|\gamma\rangle_{RGF}\|_2 \\ &= \sqrt{1-\delta_{x^*}}.\end{aligned} \tag{3.10}$$

Inserting Equations (3.9) and (3.10) into Equation (3.4), we obtain

$$\Pr[b = b'] - \frac{1}{2} \leq \frac{1}{4} \cdot \mathbb{E}\left[2\sqrt{1-\delta_{x^*}} + 1 - \delta_{x^*}\right] \ . \tag{3.11}$$

We claim that for any fixed probability p,

$$\mathbb{E}_{x^* \leftarrow p}\left[\delta_{x^*}\right] \geq 1 - q \cdot \max_x p(x) \ . \tag{3.12}$$

Using Jensen's inequality and inserting the claimed Equation (3.12) into Equation (3.11), we obtain the desired bound

$$\frac{1}{4} \cdot \mathbb{E}\left[2\sqrt{1-\delta_{x^*}} + 1 - \delta_{x^*}\right] \leq \frac{1}{2} \cdot \sqrt{qp_{\max}} + \frac{1}{4} \cdot qp_{\max} \ ,$$

it hence remains to prove Equation (3.12). Inserting the definition of δ_{x^*}, we observe that

$$\mathbb{E}_{x^* \leftarrow p}\left[\delta_{x^*}\right] = \sum_{x^* \in \{0,1\}^n} p(x^*)\left[\delta_{x^*}\right] = \sum_{x^* \in \{0,1\}^n} p(x^*)\left\|\langle\phi_0|_{F_{x^*}}\,|\gamma\rangle_{RGF}\right\|^2 \ .$$

We can now use Lemma 1.3.8 (see page 50) and its notation $|\psi_q^{(S)}\rangle$ to rewrite $|\gamma\rangle_{RGF}$, and thereby identify

$$\sum_{x^*\in\{0,1\}^n} p(x^*) \left\| \langle\phi_0|_{F_{x^*}} |\gamma\rangle_{RGF} \right\|^2$$
$$= \sum_{x^*\in\{0,1\}^n} p(x^*) \left\| \sum_{\substack{S\subset\{0,1\}^n\\ |S|\leq q}} \langle\phi_0|_{F_{x^*}} |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n-|S|)}\right)_{F_{S^c}} \right\|^2 .$$

Our goal will now be to rewrite this sum such that it can be lower bounded by the product of $(1-q\max_x p(x))$ and the squared norm of (the normalised) $|\gamma\rangle_{RGF}$ to obtain the desired bound. Using for the following second line the fact that $\langle\phi_0|_{F_{x^*}} |\psi_q^{(S)}\rangle_{RGF_S} = 0$ for all sets S that contain x^*, and that $|\phi_0\rangle_{F_{x^*}}$ is normalised, we can rewrite

$$\sum_{x^*\in\{0,1\}^n} p(x^*) \left\| \sum_{\substack{S\subset\{0,1\}^n\\ |S|\leq q}} \langle\phi_0|_{F_{x^*}} |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n-|S|)}\right)_{F_{S^c}} \right\|^2$$
$$= \sum_{x^*\in\{0,1\}^n} p(x^*) \left\| \sum_{\substack{S\subset\{0,1\}^n\\ |S|\leq q\\ S\not\ni x^*}} |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n-|S|-1)}\right)_{F_{S^c\setminus\{x^*\}}} \right\|^2$$
$$= \sum_{x^*\in\{0,1\}^n} p(x^*) \sum_{\substack{S\subset\{0,1\}^n\\ |S|\leq q\\ S\not\ni x^*}} \left\| |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n-|S|-1)}\right)_{F_{S^c\setminus\{x^*\}}} \right\|^2$$
$$= \sum_{\substack{S\subset\{0,1\}^n\\ |S|\leq q}} \sum_{x^*\in S^c} p(x^*) \left\| |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n-|S|-1)}\right)_{F_{S^c\setminus\{x^*\}}} \right\|^2$$
$$= \sum_{\substack{S\subset\{0,1\}^n\\ |S|\leq q}} \sum_{x^*\in S^c} p(x^*) \left\| |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n-|S|)}\right)_{F_{S^c}} \right\|^2 ,$$

where we get the third line because the summands in the second sum are pairwise orthogonal, we get the fourth line by a reordering of the summands, and we have used again the fact that the state $|\phi_0\rangle_{F_{x^*}}$ is normalised in the last line to reinclude register F_{x^*} into the computation of the norm. Note that in the last line, the norm in the summands is independent of x^* and can therefore be moved outside of the second sum.

For any $S \subset \{0,1\}^n$, we have

$$\sum_{x^* \in S^c} p(x^*) = 1 - \sum_{x^* \in S} p(x^*) \geq 1 - |S| \cdot \max_x p(x) \ ,$$

we hence obtain

$$\begin{aligned}
&\sum_{\substack{S \subset \{0,1\}^n \\ |S| \leq q}} \sum_{x^* \in S^c} p(x^*) \left\| |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n - |S|)} \right)_{F_{S^c}} \right\|^2 \\
&\geq \sum_{\substack{S \subset \{0,1\}^n \\ |S| \leq q}} (1 - |S| \cdot p'_{\max}) \left\| |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n - |S|)} \right)_{F_{S^c}} \right\|^2 \\
&\geq (1 - q \max_x p(x)) \sum_{\substack{S \subset \{0,1\}^n \\ |S| \leq q}} \left\| |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n - |S|)} \right)_{F_{S^c}} \right\|^2 \\
&\geq (1 - q \max_x p(x)) \, \| |\gamma\rangle_{RGF} \|^2 \\
&= 1 - q \max_x p(x),
\end{aligned}$$

where we have reidentified $\sum |\psi_q^{(S)}\rangle_{RGF_S} \otimes \left(|\phi_0\rangle^{\otimes(2^n - |S|)} \right)_{F_{S^c}}$ with $|\gamma\rangle_{RGF}$, and used the normalisation of $|\gamma\rangle_{RGF}$ in the last line. Combining the above equations proves Equation (3.12).

□

We now extend Theorem 3.1.4 to multiple reprogramming instances. To this end, we introduce helper games G_b in Figure 3.3, in which the adversary has access to oracle Reprogram′. (These are already almost the same as the Repro games used in our main Theorem 3.1.2. The only difference is that they do not sample and return the additional side information x'.) We get the following

Corollary 3.1.5. Let D be any distinguisher, issuing R many reprogramming instructions. Let $\hat{q}^{(r)}$ denote the total number of D's queries to O until the r-th query to Reprogram′. Furthermore, let $p^{(r)}$ denote the r-th distribution on X on which Reprogram′ is queried, and let

$$p_{\max}^{(r)} := \mathbb{E}\left[\max_x p^{(r)}(x) \right] \ ,$$

where the expectation is taken over D's behaviour until its r-th query to Reprogram′.

The success probability for any distinguisher D is bounded by

$$|\Pr[G_0^{\mathsf{D}} \Rightarrow 1] - \Pr[G_1^{\mathsf{D}} \Rightarrow 1]| \leq \sum_{r=1}^{R} \left(\sqrt{\hat{q}^{(r)} p_{\max}^{(r)}} + \frac{1}{2} \hat{q}^{(r)} p_{\max}^{(r)} \right) \quad .$$

```
GAMES $G_b$
    $\mathsf{O}_0 \leftarrow_\$ Y^X$
    $\mathsf{O}_1 := \mathsf{O}_0$
    $b' \leftarrow \mathsf{D}^{|\mathsf{O_b}\rangle, \textsc{Reprogram}'}$
    return $b'$

$\textsc{Reprogram}'(p)$
    $x \leftarrow p$
    $y \leftarrow_\$ Y$
    $\mathsf{O}_1 := \mathsf{O}_1^{x \mapsto y}$
    return $x$
```

Fig. 3.3: Games G_b of Corollary 3.1.5.

Proof. We define hybrid settings H_r for $r = 0, ..., R$, in which D has access to oracle O which is not reprogrammed at the first r many positions, but is reprogrammed from the $(r+1)$-th position on. Hence, H_0 is the distinguishing game G_1, and H_R is G_0. Any distinguisher D succeeds with advantage

$$\begin{aligned} |\Pr[G_0^{\mathsf{D}} \Rightarrow 1] - \Pr[G_1^{\mathsf{D}} \Rightarrow 1]| &= \Pr[H_0^{\mathsf{D}} \Rightarrow 1] - \Pr[H_R^{\mathsf{D}} \Rightarrow 1]| \\ &= \left| \sum_{r=1}^{R} \left(\Pr[H_{r-1}^{\mathsf{D}} \Rightarrow 1] - \Pr[H_r^{\mathsf{D}} \Rightarrow 1] \right) \right| \\ &\leq \sum_{r=1}^{R} \left| \Pr[H_{r-1}^{\mathsf{D}} \Rightarrow 1] - \Pr[H_r^{\mathsf{D}} \Rightarrow 1] \right| \quad , \end{aligned}$$

where we have used the triangle inequality in the last line.

To upper bound $|\Pr[H_{r-1}^{\mathsf{D}} \Rightarrow 1] - \Pr[H_r^{\mathsf{D}} \Rightarrow 1]|$, we will now define distinguishers $\hat{\mathsf{D}}_r = (\hat{\mathsf{D}}_{r,0}, \hat{\mathsf{D}}_{r,1})$ that are run in the single-instance distinguishing games G'_b of Theorem 3.1.4: Let O' denote the oracle that is provided by G'_b. Until right before the r-th query to $\textsc{Reprogram}'$, the first stage $\hat{\mathsf{D}}_{r,0}$ uses O' to simulate the hybrid setting H_{r-1} to D. (Until this query, H_{r-1} and H_r do not differ.) $\hat{\mathsf{D}}_{r,0}$ then uses as its output to game G'_b the r-th distribution on which $\textsc{Reprogram}'$ was queried. The second stage $\hat{\mathsf{D}}_{r,1}$ uses its input x^* to simulate the r-th response of $\textsc{Reprogram}'$. As from (and including) the $(r+1)$-th query, $\hat{\mathsf{D}}_{r,1}$ can simulate the reprogramming by using fresh uniformly random values to overwrite O'. To be more precise, during each call to $\textsc{Reprogram}'$ on some distribution p, $\hat{\mathsf{D}}_{r,1}$ samples $x \leftarrow p$ and $y \leftarrow_\$ Y$, and adds (x, y) to a list $\mathfrak{L}_{\mathsf{O}}$. (If x has been sampled before, $\hat{\mathsf{D}}_{r,1}$ replaces the former oracle value in the list.) $\hat{\mathsf{D}}_{r,1}$ defines O by

$$\mathsf{O}(x) := \begin{cases} y & \exists y \text{ s.th. } (x,y) \in \mathfrak{L}_{\mathsf{O}} \\ \mathsf{O}'(x) & \text{o.w.} \end{cases}$$

In the case that $\hat{\mathsf{D}}_r$ is run in game G_0', the reprogramming starts with the $(r+1)$-th query and $\hat{\mathsf{D}}_r$ perfectly simulates game H_r. In the case that $\hat{\mathsf{D}}_r$ is run in game G_1', the reprogramming already starts with the r-th query and $\hat{\mathsf{D}}_r$ perfectly simulates game H_{r-1}.

$$|\Pr[H_{r-1}^{\mathsf{D}} \Rightarrow 1] - \Pr[H_r^{\mathsf{D}} \Rightarrow 1]| = |\Pr[{G_1'}^{\hat{\mathsf{D}}_r} \Rightarrow 1] - \Pr[{G_0'}^{\hat{\mathsf{D}}_r} \Rightarrow 1]| \ .$$

Since the first stage $\hat{\mathsf{D}}_{r,0}$ issues $\hat{q}_r$ many queries to O', we can apply Theorem 3.1.4 to obtain

$$|\Pr[{G_1'}^{\hat{\mathsf{D}}_r} \Rightarrow 1] - \Pr[{G_0'}^{\hat{\mathsf{D}}_r} \Rightarrow 1]| \leq \sqrt{\hat{q}_r \cdot p_{\max}^{(r)}} + \frac{1}{2}\hat{q}_r \cdot p_{\max}^{(r)} \ .$$

□

Finally, we will now prove that Corollary 3.1.5 implies our main Theorem 3.1.2. I.e., we generalise Corollary 3.1.5 such that it considers the case where x shares a distribution with some value x', which the adversary also obtains.

Proof. Consider a distinguisher D run in games Repro_b. To upper bound D's advantage, we now define a distinguisher $\hat{\mathsf{D}}$ against the helper games G_b from Figure 3.3.

When queried on a distribution p on $X \times X'$, $\hat{\mathsf{D}}$ will simulate Reprogram as follows: $\hat{\mathsf{D}}$ will forward the marginal distribution p_X of x to its own oracle Reprogram$'$, and obtain some x that was sampled accordingly. It will then sample x' according to $p_{X'|x}$, where $p_{X'|x}$ is the probability distribution on X', conditioned on x, i.e.,

$$p_{X'|x}(x') := \frac{\Pr[x,x']}{\Pr[x]} \ .$$

where the probability in the numerator is taken over $(x,x') \leftarrow p$, and the probability in the denominator is taken over $x \leftarrow p_X$. Note that $\hat{\mathsf{D}}$ can be unbounded with regards to its running time, as the statement of Corollary 3.1.5 is information-theoretical, and hence can sample $p_{X'|x}$. Since the distribution of (x,x') is identical to p, and since the reprogramming only happens on x, $\hat{\mathsf{D}}$ perfectly simulates game Repro_b to D if run in game G_b and

$$|\Pr[\textsc{Repro}_1^{\mathsf{D}} \Rightarrow 1] - \Pr[\textsc{Repro}_0^{\mathsf{D}} \Rightarrow 1]| = |\Pr[G_1^{\hat{\mathsf{D}}} \Rightarrow 1] - \Pr[G_0^{\hat{\mathsf{D}}} \Rightarrow 1]| \ .$$

Since $\hat{\mathsf{D}}$ can answer any random oracle query issued by D by simply forwarding it, $\hat{\mathsf{D}}$ issues exactly as many queries to O (until the r-th reprogramming instruction) as D. We can now apply Corollary 3.1.5 to obtain

$$|\Pr[G_1^{\hat{\mathsf{D}}} \Rightarrow 1] - \Pr[G_0^{\hat{\mathsf{D}}} \Rightarrow 1]| \leq \sum_{r=1}^{R} \left(\sqrt{\hat{q}_r p_{\max}^{(r)}} + \frac{1}{2} \hat{q}_r p_{\max}^{(r)} \right) \ ,$$

where $p_{\max}^{(r)} = \mathbb{E} \max_x p_X^r(x)$.

□

3.2 Revisiting the Fiat-Shamir Transform

The following statement established that if an identification scheme ID is HVZK, and if $\mathsf{SIG} := \mathsf{FS}[\mathsf{ID}, \mathsf{H}]$ possesses $\mathsf{UF\text{-}CMA_0}$ security, then SIG is also $\mathsf{UF\text{-}CMA}$ secure, in the quantum random oracle model.

Theorem 3.2.1. For any (quantum) $\mathsf{UF\text{-}CMA}$ adversary A issuing at most q_S (classical) queries to the signing oracle SIGN and at most q_{H} quantum queries to H, there exists a $\mathsf{UF\text{-}CMA_0}$ adversary B and a multi-HVZK adversary C such that

$$\begin{aligned} \mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}CMA}}(\mathsf{A}) &\leq \mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}CMA_0}}(\mathsf{B}) + \mathrm{Adv}_{\mathsf{ID}}^{q_S-\mathsf{HVZK}}(\mathsf{C}) && (3.13) \\ &+ \frac{3q_S}{2} \sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})} \ , && (3.14) \end{aligned}$$

and the running time of B and C is about that of A. The bound given in Equation (3.13) also holds for the modified Fiat-Shamir transform that defines challenges by letting $c := \mathsf{H}(w, m, pk)$ instead of letting $c := \mathsf{H}(w, m)$.

Note that if ID is statistically HVZK, we can replace $\mathrm{Adv}_{\mathsf{ID}}^{q_S-\mathsf{HVZK}}(\mathsf{C})$ with $q_S \cdot \Delta_{\mathsf{HVZK}}$.

Proof. Consider the sequence of games given in Figure 3.4.

Game G_0. Since game G_0 is the original $\mathsf{UF\text{-}CMA}$ game,

$$\mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}CMA}}(\mathsf{A}) = \Pr[G_0^{\mathsf{A}} \Rightarrow 1] \ .$$

```
GAMES $G_0$ - $G_2$
01 $(pk, sk) \leftarrow \mathsf{IG}$
02 $(m^*, \sigma^*) \leftarrow \mathsf{A}^{\mathrm{SIGN}, |\mathsf{H}\rangle}(pk)$
03 if $m^* \in \mathfrak{L}_\mathcal{M}$ return 0
04 Parse $(w^*, z^*) := \sigma^*$
05 $c^* := \mathsf{H}(w^*, m^*)$
06 return $\mathsf{V}(pk, w^*, c^*, z^*)$

SIGN$(m)$
07 $\mathfrak{L}_\mathcal{M} := \mathfrak{L}_\mathcal{M} \cup \{m\}$
08 $(w, c, z) \leftarrow \mathsf{getTrans}(m)$   // $G_0$-$G_1$
09 $(w, c, z) \leftarrow \mathsf{Sim}(pk)$        // $G_2$
10 $\mathsf{H} := \mathsf{H}^{(w,m)\mapsto c}$     // $G_1$ -$G_2$
11 return $\sigma := (w, z)$

getTrans$(m)$                                   // $G_0$-$G_1$
12 $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
13 $c := \mathsf{H}(w, m)$                        // $G_0$
14 $c' \leftarrow_\$ \mathcal{C}$                  // $G_1$
15 $z \leftarrow \mathsf{Respond}(sk, w, c, \mathrm{st})$
16 return $(w, c, z)$
```

Fig. 3.4: Games G_0 - G_2 for the proof of Theorem 3.2.1.

Game G_1. In game G_1, we change the game twofold: First, the transcript is now drawn according to the underlying ID scheme, i.e., it is drawn uniformly at random as opposed to letting $c := \mathsf{H}(w, m)$, see line 14. Second, we reprogram the random oracle H in line 10 such that it is rendered a-posteriori-consistent with this transcript, i.e., we reprogram H such that $\mathsf{H}(w, m) = c$.

To upper bound the game distance, we construct a quantum distinguisher D in Figure 3.5 that is run in the adaptive reprogramming games $\mathrm{Repro}_{R,b}$ with $R := q_S$ many reprogramming instances. We identify reprogramming position x with (w, m), additional input x' with st, and y with c. Hence, the distribution p consists of the constant distribution that always returns m (as m was already chosen by A), together with the distribution $\mathsf{Commit}(sk)$. Since D perfectly simulates game G_b if run in its respective game Repro_b, we have

$$|\Pr[G_0^\mathsf{A} = 1] - \Pr[G_1^\mathsf{A} = 1]| = |\Pr[\mathrm{Repro}_1^\mathsf{D} \Rightarrow 1] - \Pr[\mathrm{Repro}_0^\mathsf{D} \Rightarrow 1]| \ .$$

Since D issues q_S reprogramming instructions and $(q_H + q_S + 1)$ many queries to H, Corollary 3.1.3 yields

$$|\Pr[\mathrm{Repro}_1^\mathsf{D} \Rightarrow 1] - \Pr[\mathrm{Repro}_0^\mathsf{D} \Rightarrow 1]| \leq \frac{3q_S}{2}\sqrt{(q_H + q_S + 1) \cdot p_{\max}} \ , \tag{3.15}$$

where $p_{\max} = \mathbb{E}_\mathsf{IG} \max_w \Pr_{W,\mathrm{ST} \leftarrow \mathsf{Commit}(sk)}[W = w] = \gamma(\mathsf{Commit})$.

```
Distinguisher $\mathsf{D}^{|\mathsf{H}\rangle}$
$(pk, sk) \leftarrow \mathsf{IG}$
$(m^*, \sigma^*) \leftarrow \mathsf{A}^{\mathrm{SIGN}, |\mathsf{H}\rangle}(pk)$
if $m^* \in \mathfrak{L}_\mathcal{M}$ return 0
Parse $(w^*, z^*) := \sigma^*$
$c^* := \mathsf{H}(w^*, m^*)$
return $\mathsf{V}(pk, w^*, c^*, z^*)$

SIGN$(m)$
$\mathfrak{L}_\mathcal{M} := \mathfrak{L}_\mathcal{M} \cup \{m\}$
$(w, \mathrm{st}) \leftarrow \mathrm{Reprogram}(m, \mathsf{Commit}(sk))$
$c := \mathsf{H}(w, m)$
$z \leftarrow \mathsf{Respond}(sk, w, c, \mathrm{st})$
return $\sigma := (w, z)$
```

Fig. 3.5: Reprogramming distinguisher D for the proof of Theorem 3.2.1.

Game G_2. In game G_2, we change the game such that the signing algorithm does not make use of the secret key any more: Instead of being defined relative to the honestly generated transcripts, signatures are now defined relative to the simulator's transcripts. We will now upper bound $|\Pr[G_1^{\mathsf{A}} = 1] - \Pr[G_2^{\mathsf{A}} = 1]|$ via computational multi-HVZK.

Consider multi-HVZK adversary C in Figure 3.6. C takes as input a list of q_s many transcripts, which are either all honest transcripts or simulated ones. Since reprogramming is done a-posteriori in game G_1, C can simulate a reprogrammed oracle H' via access to its own oracle H and an initial table look-up: C keeps track of the (classical) values on which H' has to be reprogrammed (see line 13) and tweaks A's oracle H', accordingly. The latter means that, given the table $\mathfrak{L}_{\mathsf{H}'}$ of pairs (w, m, c) that were already defined in previous signing queries, controlled on the query input being equal to (w, m), output c, and controlled on the input not being equal to any $(w, m) \in \mathfrak{L}_{\mathsf{H}'}$, forward the query to C's own oracle H. If needed, C reprograms already set values (see line 12). Given quantum access to H, C can implement this as a quantum circuit, allowing quantum access to H'.

C perfectly simulates game G_1 if run on honest transcripts, and game G_2 if run on simulated ones, hence

$$|\Pr[G_1^{\mathsf{A}} = 1] - \Pr[G_2^{\mathsf{A}} = 1]| \leq \mathrm{Adv}_{\mathsf{ID}}^{q_S-\mathsf{HVZK}}(\mathsf{C}) \ .$$

```
Adversary $\mathsf{C}^{|\mathsf{H}\rangle}(pk, ((w_i, c_i, z_i)_{i\in\{1,\cdots,q_s\}})$
01 $i := 0$
02 $\mathfrak{L}_{\mathsf{H}'} := \emptyset$
03 $(m^*, \sigma^*) \leftarrow \mathsf{A}^{\mathrm{SIGN}, |\mathsf{H}'\rangle}(pk)$
04 if $m^* \in \mathfrak{L}_{\mathcal{M}}$ return 0
05 Parse $(w^*, z^*) := \sigma^*$
06 $c^* := \mathsf{H}(w^*, m^*)$
07 return $\mathsf{V}(pk, w^*, c^*, z^*)$

SIGN$(m)$
08 $i++$
09 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
10 $(w, c, z) := (w_i, c_i, z_i)$
11 if $\exists c'$ s. th. $(w, m, c') \in \mathfrak{L}_{\mathsf{H}'}$
12     $\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \setminus \{(w, m, c')\}$
13 $\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \cup \{(w, m, c)\}$
14 return $\sigma := (w, z)$

$\mathsf{H}'(w, m)$
15 if $\exists c$ s. th. $(w, m, c) \in \mathfrak{L}_{\mathsf{H}'}$
16     return $c$
17 else return $\mathsf{H}(w, m)$
```

Fig. 3.6: HVZK adversary C for the proof of Theorem 3.2.1.

It remains to upper bound $\Pr[G_2^{\mathsf{A}} \Rightarrow 1]$. Consider adversary B, given in Figure 3.7. B is run in game $\mathsf{UF\text{-}CMA}_0$ and perfectly simulates game G_2 to A. If A wins in game G_2, it cannot have queried SIGN on m^*. Therefore, H' is not reprogrammed on (m^*, w^*) and hence, σ^* is a valid signature in B's $\mathsf{UF\text{-}CMA}_0$ game.

$$\Pr[G_2^{\mathsf{A}} \Rightarrow 1] \leq \mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}CMA}_0}(\mathsf{B}) \ .$$

Collecting the probabilities yields the desired bound.

```
Adversary $\mathsf{B}^{|\mathsf{H}\rangle}(pk)$
01 $\mathfrak{L}_{\mathsf{H}'} := \emptyset$
02 $(m^*, \sigma^*) \leftarrow \mathsf{A}^{\text{SIGN},|\mathsf{H}'\rangle}(pk)$
03 if $m^* \in \mathfrak{L}_{\mathcal{M}}$ ABORT
04 return $(m^*, \sigma^*)$

SIGN$(m)$
05 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
06 $(w, c, z) \leftarrow \mathsf{Sim}(pk)$
07 if $\exists c'$ s. th. $(w, m, c') \in \mathfrak{L}_{\mathsf{H}'}$
08     $\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \setminus \{(w, m, c')\}$
09 $\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \cup \{(w, m, c)\}$
10 return $\sigma := (w, z)$

$\mathsf{H}'(w, m)$
11 if $\exists c$ s. th. $(w, m, c) \in \mathfrak{L}_{\mathsf{H}'}$
12     return $c$
13 else
14     return $\mathsf{H}(w, m)$
```

Fig. 3.7: Adversary B for the proof of Theorem 3.2.1.

It remains to show that the bound also holds if challenges are derived by letting $c := \mathsf{H}(w, m, pk)$. To that end, we revisit the sequence of games given in Figure 3.4: We replace $c := \mathsf{H}(w, m)$ (and $c^* := \mathsf{H}(w^*, m^*)$) with $c := \mathsf{H}(w, m, pk)$ (and $c^* := \mathsf{H}(w^*, m^*, pk)$) in line 13 (line 05), and change the reprogram instruction in line 10, accordingly. Since pk is public, we can easily adapt both distinguisher D and adversaries B and C to account for these changes. In particular, D will simply include pk as a (fixed) part of the probability distribution that is forwarded to its reprogramming oracle. Since the public key holds no entropy once that it is fixed by the game, this change does not affect the upper bound given in Equation (3.15).

□

3.3 Revisiting the Hedged Fiat-Shamir Transform under Fault Attacks

In this section, we show how Corollary 3.1.3 can be used to extend the results of [AOTZ20] to the quantum random oracle model: We show that the Fiat-Shamir transform is robust against several types of one-bit fault injections, even in the quantum random oracle model, and that the hedged Fiat-Shamir transform is as robust, even if an attacker is in control of the nonce that is used to generate the signing randomness. In this section, we follow [AOTZ20] and consider the modified Fiat-Shamir transform that includes the public key into the hash when generating challenges. We consider the following one-bit tampering functions:

$\mathsf{flip\text{-}bit}_i(x)$: Does a logical negation of the i-th bit of x.

$\mathsf{set\text{-}bit}_i(x, b)$: Sets the i-th bit of x to b.

Security of (hedged) Fiat-Shamir against fault injections and nonce attacks. Next, we define UnForgeability in the presence of Faults, under Chosen Message Attacks (UF-F-CMA), for Fiat-Shamir transformed schemes. In game UF-F-CMA, the adversary has access to a faulty signing oracle FAULTSIGN which returns signatures that were created relative to an injected fault. To be more precise, game $\mathsf{UF\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}$ is defined relative to a set $\mathcal{F}$ of indices, and the indices $i \in \mathcal{F}$ specify at which point during the signing procedure exactly the faults are allowed to occur. An overview is given in Figure 3.8.

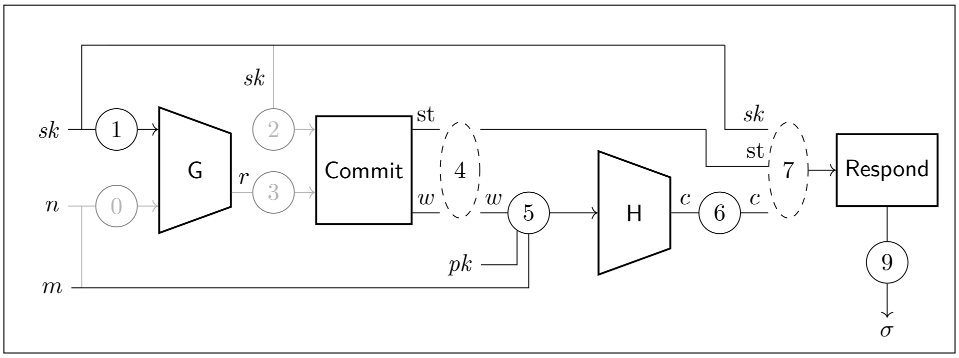


Fig. 3.8: Faulting a (hedged) Fiat-Shamir signature. Circles represent faults, and their numbers are the respective fault indices $i \in \mathcal{F}$ (following [AOTZ20], for the formal definition see Figure 3.9). Greyed out fault wires indicate that the hedged construction can not be proven robust against these faults, in general. Dashed fault nodes indicate that the Fiat-Shamir construction is robust against these faults if the scheme is subset-revealing.

For the hedged Fiat-Shamir construction, we further define UnForgeability, with control over the used Nonces and in the presence of Faults, under Chosen Message Attacks (UF-N-F-CMA). In game UF-N-F-CMA, the adversary is even allowed to control the nonce n that is used to derive the internal randomness of algorithm Commit. We therefore denote the respective oracle by N-FAULTSIGN.

Our definition slightly simplifies the one of [AOTZ20]: While [AOTZ20] also considered fault attacks on the input of algorithm Commit (with corresponding indices 2 and 3), they showed that the hedged construction can not be proven robust against these faults, in general. We therefore omitted them from our games, but adhered to the numbering for comparability. The hedged Fiat-Shamir scheme derandomises the signing procedure by setting the signing randomness to $r := \mathsf{G}(sk, m, n)$, see Figure 1.11. Hence, game UF-N-F-CMA considers two additional faults: An attacker can fault the input of G, i.e., either the secret key (fault index 1), or the tuple (m, n) (fault index 0). As shown in [AOTZ20], the hedged construction can not be proven robust against faults on (m, n), in general, therefore we only consider index 1.

Furthemore, we do not formalise derivation/serialisation and drop the corresponding indices 8 and 10 in order not to overly complicate our application example. A generalisation of our result that also considers derivation/serialisation, however, is straightforward.

Definition 3.3.1. (UF-F-CMA and UF-N-F-CMA) For any subset $\mathcal{F} \subset \{4, \cdots, 9\}$, we define the UF-F$_\mathcal{F}$-CMA game as in Figure 3.9, and the UF-F$_\mathcal{F}$-CMA advantage function of a quantum adversary A against FS[ID, H] as

$$\mathrm{Adv}^{\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}}_{\mathsf{FS[ID,H]}}(\mathsf{A}) := \Pr[\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}^{\mathsf{A}}_{\mathsf{FS[ID,H]}} \Rightarrow 1] \ .$$

Furthermore, we define the UF-N-F$_\mathcal{F}$-CMA game (also in Figure 3.9) for any subset $\mathcal{F} \subset \{1, 4, \cdots, 9\}$, and the UF-N-F$_\mathcal{F}$-CMA advantage function of a quantum adversary A against $\mathsf{SIG}' := \mathsf{R2H}[\mathsf{FS}[\mathsf{ID}, \mathsf{H}], \mathsf{G}]$ as

$$\mathrm{Adv}^{\mathsf{UF\text{-}N\text{-}F_{\mathcal{F}}\text{-}CMA}}_{\mathsf{SIG}'}(\mathsf{A}) := \Pr[\mathsf{UF\text{-}N\text{-}F_{\mathcal{F}}\text{-}CMA}^{\mathsf{A}}_{\mathsf{SIG}'} \Rightarrow 1] \ .$$

```
Game $\boxed{\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}}$ UF-N-F$_\mathcal{F}$-CMA
$(pk, sk) \leftarrow \mathsf{IG}$
$\boxed{(m^*, \sigma^*) \leftarrow \mathsf{A}^{\mathrm{FAULTSIGN}, |\mathsf{H}\rangle}(pk)}$
$(m^*, \sigma^*) \leftarrow \mathsf{A}^{\mathrm{N\text{-}FAULTSIGN}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk)$
if $m^* \in \mathfrak{L}_\mathcal{M}$ return 0
Parse $(w^*, z^*) := \sigma^*$
$c^* := \mathsf{H}(w^*, m^*)$
return $\mathsf{V}(pk, w^*, c^*, z^*)$

FAULTSIGN$(m, i \in \mathcal{F}, \phi)$
$f_i := \phi$ and $f_j := Id \ \forall \ j \neq i$

$(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
$(w, \mathrm{st}) := f_4(w, \mathrm{st})$
$(\hat{w}, \hat{m}, \hat{pk}) := f_5(w, m, pk)$
$c := f_6(\mathsf{H}(\hat{w}, \hat{m}, \hat{pk}))$
$z \leftarrow \mathsf{Respond}(f_7(sk, c, \mathrm{st}))$
$\mathfrak{L}_\mathcal{M} := \mathfrak{L}_\mathcal{M} \cup \{\hat{m}\}$
return $\sigma := f_9(w, z)$

N-FAULTSIGN$(m, n, i \in \mathcal{F}, \phi)$
$f_i := \phi$ and $f_j := Id \ \forall \ j \neq i$
$r := \mathsf{G}(f_1(sk), m, n)$
$(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk; r)$
$(w, \mathrm{st}) := f_4(w, \mathrm{st})$
$(\hat{w}, \hat{m}, \hat{pk}) := f_5(w, m, pk)$
$c := f_6(\mathsf{H}(\hat{w}, \hat{m}, \hat{pk}))$
$z \leftarrow \mathsf{Respond}(f_7(sk, c, \mathrm{st}))$
$\mathfrak{L}_\mathcal{M} := \mathfrak{L}_\mathcal{M} \cup \{\hat{m}\}$
return $\sigma := f_9(w, z)$
```

Fig. 3.9: Game UF-F$_\mathcal{F}$-CMA for SIG = FS[ID, H] and game UF-N-F$_\mathcal{F}$-CMA for the hedged Fiat-Shamir construction $\mathsf{SIG}' := \mathsf{R2H}[\mathsf{FS}[\mathsf{ID}, \mathsf{H}], \mathsf{G}]$, defined relative to a set $\mathcal{F}$ of allowed fault index positions. ϕ denotes the fault function, which either negates one particular bit of its input, sets one particular bit of its input to 0 or 1, or does nothing. We implicitly require fault index i to be contained in $\mathcal{F}$, i.e., we make the convention that both faulty signing oracles return $\perp$ if $i \notin \mathcal{F}$.

From UF-CMA$_0$ to UF-F-CMA. First, we generalise [AOTZ20, Lemma 5] to the quantum random oracle model. The proof is given in Section 3.3.1.

Theorem 3.3.2. Assume ID to be validity aware (see Definition 1.2.3, page 34). If SIG := FS[ID, H] is UF-CMA$_0$ secure, then SIG is also UF-F$_\mathcal{F}$-CMA secure for $\mathcal{F} := \{5, 6, 9\}$, in the quantum random oracle model. Concretely, for any adversary A against the UF-F$_\mathcal{F}$-CMA security of SIG, issuing at most q_S (classical) queries to FAULTSIGN and q_H (quantum) queries to H, there exists an UF-CMA$_0$ adversary B and a multi-HVZK adversary C such that

$$\mathrm{Adv}_{\mathsf{SIG}}^{\mathsf{UF\text{-}F}_{\{5,6,9\}}\mathsf{\text{-}CMA}}(\mathsf{A}) \leq \mathrm{Adv}_{\mathsf{SIG}}^{\mathsf{UF\text{-}CMA}_0}(\mathsf{B}) + \mathrm{Adv}_{\mathsf{ID}}^{q_S\mathsf{-HVZK}}(\mathsf{C}) \\ + \frac{3q_S}{2}\sqrt{2 \cdot (q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})} \ . \tag{3.16}$$

and B and C have about the running time of A.

If we assume that ID is subset-revealing, then SIG is even $\mathsf{UF\text{-}F}_{\mathcal{F}'}\mathsf{\text{-}CMA}$ secure for $\mathcal{F}' := \mathcal{F} \cup \{4,7\}$. Concretely, the bound of Equation (3.16) then holds also for $\mathcal{F}' = \{4,5,6,7,9\}$.

From UF-F-CMA to UF-N-F-CMA. Second, we generalise [AOTZ20, Lemma 4] to the QROM. The proof is given in Section 3.3.9.

Theorem 3.3.3. If $\mathsf{SIG} := \mathsf{FS}[\mathsf{ID},\mathsf{H}]$ is $\mathsf{UF\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}$ secure for a fault index set $\mathcal{F}$, then $\mathsf{SIG}' := \mathsf{R2H}[\mathsf{SIG},\mathsf{G}]$ is $\mathsf{UF\text{-}N\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}$ secure for $\mathcal{F}' := \mathcal{F} \cup \{1\}$, in the quantum random oracle model, against any adversary that issues no query (m,n) to N-FAULTSIGN more than once. Concretely, for any adversary A against the $\mathsf{UF\text{-}N\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}$ security of SIG' for $\mathcal{F}'$, issuing at most q_S queries to N-FAULTSIGN, at most q_{H} queries to H, and at most q_{G} queries to G, there exist $\mathsf{UF\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}$ adversaries B_1 B_2 such that

$$\mathrm{Adv}_{\mathsf{SIG}'}^{\mathsf{UF\text{-}N\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}}(\mathsf{A}) \leq \mathrm{Adv}_{\mathsf{SIG}}^{\mathsf{UF\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}}(\mathsf{B}_1) + 2q_{\mathsf{G}} \cdot \sqrt{\mathrm{Adv}_{\mathsf{SIG}}^{\mathsf{UF\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}}(\mathsf{B}_2)} \ ,$$

and B_1 has about the running time of A, while B_2 has a running time of roughly $\mathrm{Time}(\mathsf{B}_2) \approx \mathrm{Time}(\mathsf{A}) + |sk| \cdot (\mathrm{Time}(\mathsf{Sign}) + \mathrm{Time}(\mathsf{Vrfy}))$, where $|sk|$ denotes the length of sk.

With regards to the reduction's advantage, this proof is not as tight as the one in [AOTZ20]: $\mathsf{R2H}[\mathsf{SIG},\mathsf{G}]$ derives the commitment randomness as $r := \mathsf{G}(sk,m,n)$. During our proof, we need to decouple r from the secret key. In the ROM, it is straightforward how to turn any adversary noticing this change into an extractor that returns the secret key. In the QROM, however, all currently known extraction techniques still come with a quadratic loss in the extraction probability. On the other hand, our reduction is tighter with regards to running time, which we reduce by a factor of q_{G} when compared to [AOTZ20].

If the scheme is hedged with an independent seed s of length ℓ (instead of sk), it can be shown with a multi-instance generalisation of [SXY18, Lem. 2.2] that

$$\mathrm{Adv}_{\mathsf{SIG}'}^{\mathsf{UF\text{-}N\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}}(\mathsf{A}) \leq \mathrm{Adv}_{\mathsf{SIG}}^{\mathsf{UF\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}}(\mathsf{B}) + (\ell+1) \cdot (q_S + q_{\mathsf{G}}) \cdot \sqrt{1/2^{\ell-1}} \ .$$

3.3.1 *From* UF-CMA$_0$ *to* UF-F-CMA *(Proof of Theorem 3.3.2)*

Following the proof structure of [AOTZ20], we will break down the proof into several sequential steps. Consider the sequence of games, given in Figure 3.10. With each game-hop, we take one more index i for which we replace execution of FAULTSIGN with a simulation that can be executed without knowledge of sk, see line Item 14. The workings of these simulations will be made explicit in the proof for the respective game-hop. Similar to [AOTZ20], the order of the indices for which we start simulating is 9, 5, 6, 7, 4.

For a scheme that cannot be assumed to be subset-revealing, we will only proceed until game G_3, and then use game G_3 to argue that we can turn any adversary against the UF-F$_{\{5,6,9\}}$-CMA security of SIG into an UF-CMA$_0$ adversary (see Lemma 3.3.7).

If we can assume the scheme to be subset-revealing, we will proceed until game G_5, and then use game G_5 to argue that we can turn any adversary against the UF-F$_{\{4,5,6,7,9\}}$-CMA security of SIG into an UF-CMA$_0$ adversary (see Lemma 3.3.10).

Note that our sequential proof is given for statistical sHVZK. The reason why we do not give our proof in the computational setting right away is that it would then be required to make all of our changes at once, rendering the proof overly involved, while not providing any new insights. At the end of this section, we show how to generalise the proof to the computational setting.

```
Games $G_0$ - $G_5$
01 $(pk, sk) \leftarrow \mathsf{IG}$
02 $(m^*, \sigma^*) \leftarrow \mathsf{A}^{\text{FAULTSIGN},|\mathsf{H}\rangle}(pk)$
03 if $m^* \in \mathfrak{L}_{\mathcal{M}}$ return 0
04 Parse $(w^*, z^*) := \sigma^*$
05 $c^* := \mathsf{H}(w^*, m^*)$
06 return $\mathsf{V}(pk, w^*, c^*, z^*)$
```

```
FAULTSIGN$(m, i \in \mathcal{F}, \phi)$
07 $S := \emptyset$                  // $G_0$
08 $S := \{9\}$                      // $G_1$-$G_5$
09 $S := S \cup \{5\}$               // $G_2$-$G_5$
10 $S := S \cup \{6\}$               // $G_3$-$G_5$
11 $S := S \cup \{7\}$               // $G_4$-$G_5$
12 $S := S \cup \{4\}$               // $G_5$
13 if $i \in S$
14     $\sigma \leftarrow \mathsf{simSignature}_i(m, \phi)$
15 else $\sigma \leftarrow$ getSignature$(m, i, \phi)$
16 return $\sigma$
```

```
getSignature$(m, i, \phi)$
17 $f_i := \phi$ and $f_j := Id \; \forall \; j \neq i$
18 $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
19 $(w, \mathrm{st}) := f_4(w, \mathrm{st})$
20 $(\hat{w}, \hat{m}, \hat{pk}) := f_5(w, m, pk)$
21 $c := f_6(\mathsf{H}(\hat{m}, \hat{w}, \hat{pk}))$
22 $z \leftarrow \mathsf{Respond}(f_7(sk, c, \mathrm{st}))$
23 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
24 return $\sigma := f_9(w, z)$
```

Fig. 3.10: Games G_0 - G_5 for the proof of Theorem 3.3.2. Helper methods getSignature and $\mathsf{simSignature}_i$ (where $i \in \{4, 5, 6, 7, 9\}$) are internal and cannot be accessed directly by A. Recall that we require queried indices i to be contained in $\mathcal{F}$ (see Figure 3.9).

Game G_0. Since game G_0 is the original $\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}$ game,

$$\mathrm{Adv}^{\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}}_{\mathsf{SIG'}}(\mathsf{A}) = \Pr[G_0^{\mathsf{A}} \Rightarrow 1] \ .$$

Games G_1 - G_3. In games G_1 to G_3, we sequentially start to simulate faulty signatures for fault indices 9, 5 and 6.

Lemma 3.3.4. There exists an algorithm $\mathsf{simSignature}_9$ such that for any adversary A against the $\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}$ security of SIG, issuing at most $q_{S,9}$ queries to FAULTSIGN on index 9, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H,

$$|\Pr[G_0^{\mathsf{A}} = 1] - \Pr[G_1^{\mathsf{A}} = 1]| \leq q_{S,9} \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{2}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})}\right) \ . \tag{3.17}$$

The details on algorithm $\mathsf{simSignature}_9$ and the proof for Equation (3.17) are given in Section 3.3.2.

Lemma 3.3.5. There exists an algorithm $\mathsf{simSignature}_5$ such that for any adversary A against the $\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}$ security of SIG, issuing at most $q_{S,5}$ queries to FAULTSIGN on index 5, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H,

$$|\Pr[G_1^{\mathsf{A}} = 1] - \Pr[G_2^{\mathsf{A}} = 1]| \leq q_{S,5} \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{2}\sqrt{(q_H + q_S + 1) \cdot 2\gamma(\mathsf{Commit})}\right) \ . \tag{3.18}$$

The details on algorithm $\mathsf{simSignature}_5$ and the proof for Equation (3.18) are given in Section 3.3.3.

Lemma 3.3.6. There exists an algorithm $\mathsf{simSignature}_6$ such that for any adversary A against the $\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}$ security of SIG, issuing at most $q_{S,6}$ queries to FAULTSIGN on index 6, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H,

$$|\Pr[G_2^{\mathsf{A}} = 1] - \Pr[G_3^{\mathsf{A}} = 1]| \leq q_{S,6} \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{2}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})}\right) \ . \tag{3.19}$$

The details on algorithm $\mathsf{simSignature}_6$ and the proof for Equation (3.19) are given in Section 3.3.4. What we have shown by now is that

$$|\Pr[G_0^{\mathsf{A}} \Rightarrow 1] - \Pr[G_3^{\mathsf{A}} \Rightarrow 1]| \leq q_{S,\{5,6,9\}} \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{2}\sqrt{(q_H + q_S + 1) \cdot 2\gamma(\mathsf{Commit})}\right) \ , \tag{3.20}$$

where $q_{S,\{5,6,9\}}$ denotes the maximal number of queries to FAULTSIGN on all indices $i \in \{5,6,9\}$. We are now ready to give our first security statement.

Lemma 3.3.7. For any adversary A against the $\mathsf{UF\text{-}F}_{\{5,6,9\}}\text{-}\mathsf{CMA}$ security of SIG, there exists an adversary B such that

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] \leq \mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}CMA}_0}(\mathsf{B}) \ ,$$

and B has the same running time as A.

The proof is given in Section 3.3.5. Collecting the probabilities, we obtain

$$\begin{aligned}\mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}F}_{\{5,6,9\}}\text{-}\mathsf{CMA}}(\mathsf{A}) \leq &\mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}CMA}_0}(\mathsf{B}) \\ &+ q_S \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{2}\sqrt{(q_H + q_S + 1) \cdot 2\gamma(\mathsf{Commit})}\right) \ .\end{aligned}$$

GAMES G_4 - G_5. In games G_4 to G_5, we sequentially start to simulate faulty signatures for fault indices 7 and 4.

Lemma 3.3.8. Suppose that ID is subset-revealing. Then there exists an algorithm $\mathsf{simSignature}_7$ such that for any adversary A against the $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$ security of SIG, issuing at most $q_{S,7}$ queries to FAULTSIGN on index 7, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H,

$$|\Pr[G_3^{\mathsf{A}} = 1] - \Pr[G_4^{\mathsf{A}} = 1]| \leq q_{S,7} \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{2}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})}\right) \ . \tag{3.21}$$

The details on algorithm $\mathsf{simSignature}_7$ and the proof for Equation (3.21) are given in Section 3.3.6.

Lemma 3.3.9. Suppose that ID is subset-revealing. There exists an algorithm $\mathsf{simSignature}_4$ such that for any adversary A against the $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$ security of SIG, issuing at most $q_{S,4}$ queries to FAULTSIGN on index 4, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H,

$$|\Pr[G_4^{\mathsf{A}} = 1] - \Pr[G_5^{\mathsf{A}} = 1]| \leq q_{S,6} \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{2}\sqrt{(q_H + q_S + 1) \cdot 2\gamma(\mathsf{Commit})}\right) \ . \tag{3.22}$$

The details on algorithm $\mathsf{simSignature}_4$ and the proof for Equation (3.22) are given in Section 3.3.7. What we have shown by now is that

$$|\Pr[G_3^{\mathsf{A}} \Rightarrow 1] - \Pr[G_5^{\mathsf{A}} \Rightarrow 1]| \leq q_{S,\{4,7\}} \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{\sqrt{2}}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})}\right) \ ,$$

where $q_{S,\{4,7\}}$ denotes the maximal number of queries to FAULTSIGN on all indices $i \in \{4, 7\}$. We are now ready to give our second security statement.

Lemma 3.3.10. For any adversary A against the $\mathsf{UF\text{-}F}_{\{4,5,6,7,9\}}\mathsf{\text{-}CMA}$ security of SIG, there exists an adversary B such that

$$\Pr[G_5^{\mathsf{A}} \Rightarrow 1] \leq \mathrm{Adv}^{\mathsf{UF\text{-}CMA}_0}_{\mathsf{FS[ID,H]}}(\mathsf{B}) \ ,$$

and B has the same running time as A. The proof is given in Section 3.3.8.

Collecting the probabilities, we obtain

$$\begin{aligned} \mathrm{Adv}^{\mathsf{UF\text{-}F}_{\{4,5,6,7,9\}}\mathsf{\text{-}CMA}}_{\mathsf{FS[ID,H]}}(\mathsf{A}) \leq & \mathrm{Adv}^{\mathsf{UF\text{-}CMA}_0}_{\mathsf{FS[ID,H]}}(\mathsf{B}) \\ & + q_S \cdot \left(\Delta_{\mathsf{sHVZK}} + \frac{3}{\sqrt{2}}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})}\right) \ , \end{aligned}$$

given that ID is subset-revealing.

Generalising the proof for computational sHVZK. To generalise the proof, we observe that every game-hop consists of two steps: Adaptive reprogramming and, subsequently, replacing honest transcripts with simulated ones. To obtain the result for computational sHVZK, we have to reorder the games: We will first reprogram the random oracle for *all* fault indices *at once*, with oracle FAULTSIGN reprogramming the random oracle for each fault index as specified in the sequential proof (see Sections 3.3.2 to 3.3.7). Combined reprogramming yields an upper bound of $\frac{3q_S}{\sqrt{2}}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})}$. After these changes, the random oracle is a-posteriori reprogrammed such that it is consistent with the transcripts, and hence, the transition to simulated transcripts can be reduced to distinguishing the special computational multi-HVZK games (see Definition 1.2.6). In more detail, the HVZK reduction can simply use its own transcript oracle $\mathsf{getTransChall}$, and simulate the adaptive reprogramming like our $\mathsf{UF\text{-}CMA}_0$

reductions, see, e.g., the reduction given in Section 3.3.5.

3.3.2 Game G_1: Simulating FAULTSIGN *for index 9 (Proof of Lemma 3.3.4)*

As a warm-up, we will first consider simulations with respect to fault index 9. Recall that index 9 denotes the fault type which allows A to fault the resulting (honestly generated) signature (see line 05 in Figure 3.11). To prove Lemma 3.3.4, let A be an adversary against the $\mathsf{UF}\text{-}\mathsf{F}_{\mathcal{F}}\text{-}\mathsf{CMA}$ security of SIG, issuing at most $q_{S,9}$ queries to FAULTSIGN on index 9, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H. We define the signature simulation algorithm $\mathsf{simSignature}_9$ as in Figure 3.11.

```
FAULTSIGN$(m, i = 9, \phi)$
01 $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
02 $c := \mathsf{H}(w, m, pk)$
03 $z \leftarrow \mathsf{Respond}(sk, c, \mathrm{st})$
04 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
05 return $\sigma := \phi(w, z)$

$\mathsf{simSignature}_9(m, \phi)$
06 $c \leftarrow_\$ \mathcal{C}$
07 $(w, z) \leftarrow \mathsf{Sim}(pk, c)$
08 $\mathsf{H} := \mathsf{H}^{(w,m,pk)\mapsto c}$
09 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
10 return $\sigma := \phi(w, z)$
```

Fig. 3.11: Original oracle FAULTSIGN for the case that $i = 9$, and signature simulation algorithm $\mathsf{simSignature}_9$ for the proof of Lemma 3.3.4.

To proceed from game G_0 to G_1, we use an argument similar to the one given in Theorem 3.2.1: During execution of FAULTSIGN$(m, 9, \phi)$, we first derandomise the challenges and reprogram H such that it is rendered a-posteriori-consistent with the resulting transcripts, resulting in an invocation of Corollary 3.1.3, where $R = q_{S,9}$, $q = q_H + q_S + 1$, and $p_{\max} = \gamma(\mathsf{Commit})$. As the second step, we then make use of the fact that we assume ID to be statistically sHVZK, and hence, honestly generated transcripts can be replaced with simulated ones during execution of FAULTSIGN$(m, 9, \phi)$.

After these changes, FAULTSIGN$(m, 9, \phi) = \mathsf{simSignature}_9(m, \phi)$ and

$$|\Pr[G_0^{\mathsf{A}} = 1] - \Pr[G_1^{\mathsf{A}} = 1]| \leq q_{S,9} \cdot \Delta_{\mathsf{sHVZK}} + \frac{3q_{S,9}}{2}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})} \ .$$

3.3.3 Game G_2: Simulating FAULTSIGN *for index 5 (Proof of Lemma 3.3.5)*

Recall that index 5 denotes the fault type which allows A to fault the triplet (w, m, pk), when taken as input to random oracle H to compute the challenge c (see line 02 in Figure 3.12). To prove Lemma 3.3.5, let A be an adversary against the $\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}$ security of SIG, issuing at most $q_{S,5}$ queries to FAULTSIGN on index 5, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H. We define the signature simulation algorithm $\mathsf{simSignature}_5$ as in Figure 3.12.

```
FAULTSIGN$(m, i = 5, \phi)$
01 $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
02 $(\hat{w}, \hat{m}, \hat{pk}) := \phi(w, m, pk)$
03 $c := \mathsf{H}(\hat{w}, \hat{m}, \hat{pk}))$
04 $z \leftarrow \mathsf{Respond}(sk, c, \mathrm{st})$
05 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
06 return $\sigma := (w, z)$

$\mathsf{simSignature}_5(m, \phi)$
07 $c \leftarrow_\$ \mathcal{C}$
08 $(w, z) \leftarrow \mathsf{Sim}(pk, c)$
09 $(\hat{w}, \hat{m}, \hat{pk}) := \phi(w, m, pk)$
10 $\mathsf{H} := \mathsf{H}^{(\hat{w}, \hat{m}, \hat{pk}) \mapsto c}$
11 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
12 return $\sigma := (w, z)$
```

Fig. 3.12: Original oracle FAULTSIGN for the case that $i = 5$, and signature simulation algorithm $\mathsf{simSignature}_5$ for the proof of Lemma 3.3.5.

To proceed from game G_1 to G_2, we adapt the argument of Section 3.3.2: During execution of FAULTSIGN$(m, 5, \phi)$, we first derandomise the challenges and reprogram H such that it is rendered a-posteriori-consistent with with the resulting transcripts, resulting in an invocation of Corollary 3.1.3, where $R = q_{S,5}$ and $q = q_H + q_S + 1$. To make $p_{\max}$ explicit, let ϕ_w (ϕ_m, ϕ_{pk}) denote the share of ϕ acting on w (m, pk). We can now identify reprogramming positions x with $(\phi_m(m), \phi_w(w), \phi_{pk}(pk))$. The distribution p consists hence of the constant distribution that always returns $\phi_m(m)$ and $\phi_{pk}(pk)$, as these parts of the reprogramming position are already fixed, together with the distribution $\phi_w(\mathsf{Commit}(sk))$. Note that ϕ_w is either the identity, a bit flip, or a function that fixes one bit of w, hence $p_{\max} \leq 2\gamma(\mathsf{Commit})$.

As the second step, we can again make use of the fact that we assume ID to be statistically sHVZK, and honestly generated transcripts can be replaced with simulated ones during execution of FAULTSIGN$(m, 5, \phi)$.

After these changes, FAULTSIGN$(m, 5, \phi) = \mathsf{simSignature}_5(m, \phi)$ and

$$|\Pr[G_1^{\mathsf{A}} = 1] - \Pr[G_2^{\mathsf{A}} = 1]| \leq q_{S,5} \cdot \Delta_{\mathsf{sHVZK}} + \frac{3q_{S,5}}{2}\sqrt{(q_H + q_S + 1) \cdot 2\gamma(\mathsf{Commit})} \ .$$

3.3.4 Game G_3: Simulating FAULTSIGN *for index 6 (Proof of Lemma 3.3.6)*

Recall that index 6 denotes the fault type which allows A to fault the output $c = \mathsf{H}(w, m, pk)$ of the challenge hash function H (see line 03 in Figure 3.13). To prove Lemma 3.3.6, let A be an adversary against the $\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}$ security of SIG, issuing at most $q_{S,6}$ queries to FAULTSIGN on index 6, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H. We define the signature simulation algorithm $\mathsf{simSignature}_6$ as in Figure 3.13.

```
FAULTSIGN$(m, i = 6, \phi)$
$(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
$c := \mathsf{H}(w, m, pk)$
$z \leftarrow \mathsf{Respond}(sk, \phi(c), \mathrm{st})$
$\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
return $\sigma := (w, z)$
```

```
$\mathsf{simSignature}_6(m, \phi)$
$c \leftarrow_\$ \mathcal{C}$
$(w, z) \leftarrow \mathsf{Sim}(pk, \phi(c))$
if $\phi(c) \notin \mathcal{C}$
    $z := \bot$
$\mathsf{H} := \mathsf{H}^{(w,m,pk) \mapsto c}$
$\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
return $\sigma := (w, z)$
```

Fig. 3.13: Original oracle FAULTSIGN for the case that $i = 6$, and signature simulation algorithm $\mathsf{simSignature}_6$ for the proof of Lemma 3.3.6.

To proceed from game G_2 to G_3, we again adapt the argument from Section 3.3.2: During execution of FAULTSIGN$(m, 6, \phi)$, we first derandomise the challenges and reprogram H such that it is rendered a-posteriori-consistent with with the resulting transcripts, resulting in an invocation of Corollary 3.1.3, where $R = q_{S,6}$ and $q = q_H + q_S + 1$. Like in Section 3.3.2, $p_{\max} = \gamma(\mathsf{Commit})$.

As the second step, we can again make use of the fact that we assume ID to be statistically sHVZK, and hence, honestly generated transcripts can be replaced with simulated ones during execution of FAULTSIGN$(m, 6, \phi)$. Note that as the challenges are faulty, however, we have to simulate rejection whenever faulting the challenge results in an invalid challenge, i.e., whenever $\phi(c) \notin \mathcal{C}$.

Since ID is validity aware (see Definition 1.2.3), it holds that after these changes, FAULTSIGN$(m, 6, \phi) = \mathsf{simSignature}_6(m, \phi)$ and

$$|\Pr[G_2^{\mathsf{A}} = 1] - \Pr[G_3^{\mathsf{A}} = 1]| \leq q_{S,6} \cdot \Delta_{\mathsf{sHVZK}} + \frac{3q_{S,6}}{2}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})} \ .$$

3.3.5 $\mathsf{UF\text{-}CMA}_0$ *adversary for game* G_3*, for* $\mathcal{F} = \{5, 6, 9\}$ *(Proof of Lemma 3.3.7)*

Recall that in game G_3, faulty signatures are simulated for all indices $i \in \{5, 6, 9\}$. Since adversaries against the $\mathsf{UF\text{-}F}_{\{5,6,9\}}\text{-}\mathsf{CMA}$ security of SIG only have access to FAULTSIGN(m, i, ϕ) for $i \in \{5, 6, 9\}$, the game derives all oracle answers by a call to one of the simulated oracles $\mathsf{simSignature}_i(m, \phi)$, where $i \in \{5, 6, 9\}$. To prove

Lemma 3.3.7, we construct an $\mathsf{UF\text{-}CMA}_0$ adversary B in Figure 3.14.

```
Adversary $\mathsf{B}^{|\mathsf{H}\rangle}(pk)$
$(m^*, \sigma^*) \leftarrow \mathsf{A}^{\text{FAULTSIGN},|\mathsf{H}'\rangle}(pk)$
if $m^* \in \mathfrak{L}_{\mathcal{M}}$ ABORT
return $(m^*, \sigma^*)$

FAULTSIGN$(m, i \in \{5, 6, 9\}, \phi)$
$\sigma \leftarrow \mathsf{simSignature}_i(m, \phi)$
return $\sigma$

$\mathsf{H}'(w, m, pk)$
if $\exists c$ s. th. $(w, m, pk, c) \in \mathfrak{L}_{\mathsf{H}'}$
    return $c$
else return $\mathsf{H}(w, m, pk)$

$\mathsf{simSignature}_9(m, \phi)$
$c \leftarrow_\$ \mathcal{C}$
$(w, z) \leftarrow \mathsf{Sim}(pk, c)$
if $\exists c'$ s. th. $(w, m, pkc') \in \mathfrak{L}_{\mathsf{H}'}$
    $\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \setminus \{(w, m, pkc')\}$
$\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \cup \{(w, m, pk, c)\}$
$\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
return $\sigma := \phi(w, z)$

$\mathsf{simSignature}_5(m, \phi)$
$c \leftarrow_\$ \mathcal{C}$
$(w, z) \leftarrow \mathsf{Sim}(pk, c)$
$(\hat{w}, \hat{m}, \hat{pk}) := \phi(w, m, pk)$
if $\exists c'$ s. th. $(\hat{w}, \hat{m}, \hat{pk}, c') \in \mathfrak{L}_{\mathsf{H}'}$
    $\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \setminus \{(\hat{w}, \hat{m}, \hat{pk}, c')\}$
$\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \cup \{(\hat{w}, \hat{m}, \hat{pk}, c)\}$
$\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
return $\sigma := (\hat{w}, z)$

$\mathsf{simSignature}_6(m, \phi)$
$c \leftarrow_\$ \mathcal{C}$
$(w, z) \leftarrow \mathsf{Sim}(pk, \phi(c))$
if $\phi(c) \notin \mathcal{C}$
    $z := \perp$
if $\exists c'$ s. th. $(w, m, pkc') \in \mathfrak{L}_{\mathsf{H}'}$
    $\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \setminus \{(w, m, pk, c')\}$
$\mathfrak{L}_{\mathsf{H}'} := \mathfrak{L}_{\mathsf{H}'} \cup \{(w, m, pk, c)\}$
$\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
return $\sigma := (w, z)$
```

Fig. 3.14: $\mathsf{UF\text{-}CMA}_0$ Adversary B for the proof of Lemma 3.3.7.

Since in game G_3, all signatures are defined relative to simulated transcripts, and the random oracle is reprogrammed accordingly, B perfectly simulates G_3 and has the same running time as A.

Furthermore, A can not win if m^* was a query to FAULTSIGN. Therefore, it is ensured that no reprogramming did occur on m^* and A's signature is also valid in B's $\mathsf{UF\text{-}CMA}_0$ game.

$$\Pr[G_3^{\mathsf{A}} \Rightarrow 1] \leq \mathrm{Adv}_{\mathsf{FS[ID,H]}}^{\mathsf{UF\text{-}CMA}_0}(\mathsf{B}) \ .$$

3.3.6 Game G_4: Simulating FAULTSIGN for index 7 (Proof of Lemma 3.3.8)

Recall that index 7 denotes the fault type which allows A to fault the input (sk, c, st) to the response function Respond (see line 03 in Figure 3.15), and that we assume that ID is subset-revealing. To prove Lemma 3.3.8, let A be an adversary against the UF-F$_{\mathcal{F}}$-CMA security of SIG, issuing at most $q_{S,7}$ queries to FAULTSIGN on index 7, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H. We define the signature simulation algorithm $\mathsf{simSignature}_7$ as in Figure 3.15.

```
FAULTSIGN($m, i = 7, \phi$)
01 $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
02 $c := \mathsf{H}(w, m, pk)$
03 $z \leftarrow \mathsf{Respond}(\phi(sk, c, \mathrm{st}))$
04 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
05 return $\sigma := (w, z)$
```

```
$\mathsf{simSignature}_7(m, \phi)$
06 $c \leftarrow_{\$} \mathcal{C}$
07 Parse $(\phi_{sk}, \phi_c, \phi_{\mathrm{st}}) := \phi$
08 if $\phi_c \neq Id$                          // $\phi$ targets $c$
09    $(w, z) \leftarrow \mathsf{Sim}(pk, \phi(c))$
10    if $\phi(c) \notin \mathcal{C}$
11       $z := \bot$
12 else
13    $(w, z) \leftarrow \mathsf{Sim}(pk, c)$
14    if $\phi_{\mathrm{st}} \neq Id$              // $\phi$ targets st
15       $I \leftarrow \mathsf{DeriveSet}(c)$
16       Parse $(\mathrm{st}_i)_{i \in I} := z$
17       $z := (\phi_{\mathrm{st},i}(\mathrm{st}_i))_{i \in I}$
18 $\mathsf{H} := \mathsf{H}^{(w,m,pk) \mapsto c}$
19 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
20 return $\sigma := (w, z)$
```

Fig. 3.15: Original oracle FAULTSIGN for the case that $i = 7$, and signature simulation algorithm $\mathsf{simSignature}_7$ for the proof of Lemma 3.3.8.

If fault function ϕ is targeted at c, the situation is essentially the same as for fault index 6, and thus, the simulation strategy is identical to that of $\mathsf{simSignature}_6$ (see Section 3.3.4). If fault function ϕ is targeted at sk, ϕ has no effect whatsoever since we assume ID to be subset-revealing, meaning that the responses returned by Respond do not depend on sk (see Definition 1.2.7). The simulation strategy is hence identical to that of $\mathsf{simSignature}_9$. The simulation algorithm covers both cases by dissecting ϕ into the shares ϕ_{sk} $(\phi_c, \phi_{\mathrm{st}})$ acting on sk (c, st) and treating the cases where $\phi_c \neq Id$ $(\phi_{sk} \neq Id)$ similar to $\mathsf{simSignature}_6$ $(\mathsf{simSignature}_9)$.

It remains to discuss the case where ϕ is targeted at st. Since we assume ID to be subset-revealing (see Definition 1.2.7), we observe that $\mathsf{Respond}(\phi(sk, c, \mathrm{st})) = \mathsf{Respond}(sk, c, \phi_{\mathrm{st}}(\mathrm{st})) = ((\phi_{\mathrm{st}}(\mathrm{st}))_i)_{i \in I}$, where $I = \mathsf{DeriveSet}(c)$. Hence, computing $z \leftarrow \mathsf{Respond}(\phi(sk, c, \mathrm{st}))$ is equivalent to deriving $I \leftarrow \mathsf{DeriveSet}(c)$, only considering the shares $\phi_{\mathrm{st},i}$ of ϕ_{st} that act on st_i, and returning $(\phi_{\mathrm{st},i}(\mathrm{st}_i))_{i \in I}$. With this alternative description of the original $\mathsf{Respond}$ algorithm, it can easily be verified that even for the case where ϕ is targeted at the state, honest transcripts can be replaced with simulated transcripts by letting ϕ act on the response z as described above.

After these changes, $\mathrm{FAULTSIGN}(m, 7, \phi) = \mathsf{simSignature}_7(m, \phi)$ and

$$|\Pr[G_3^{\mathsf{A}} = 1] - \Pr[G_4^{\mathsf{A}} = 1]| \leq q_{S,7} \cdot \Delta_{\mathsf{sHVZK}} + \frac{3q_{S,7}}{2}\sqrt{(q_H + q_S + 1) \cdot \gamma(\mathsf{Commit})} \ .$$

3.3.7 *Game G_5: Simulating* FAULTSIGN *for index 4 (Proof of Lemma 3.3.9)*

Recall that index 4 denotes the fault type which allows A to fault the output of $\mathsf{Commit}(sk)$ (see line 02 in Figure 3.16). To prove Lemma 3.3.9, let A be an adversary against the $\mathsf{UF\text{-}F_{\mathcal{F}}\text{-}CMA}$ security of SIG, issuing at most $q_{S,4}$ queries to FAULTSIGN on index 4, q_S queries to FAULTSIGN in total, and at most q_{H} queries to H. We define the signature simulation algorithm $\mathsf{simSignature}_4$ as in Figure 3.16.

```
FAULTSIGN$(m, i = 4, \phi)$
01 $(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk)$
02 $(w, \mathrm{st}) := \phi(w, \mathrm{st})$
03 $c := \mathsf{H}(w, m, pk)$
04 $z \leftarrow \mathsf{Respond}(sk, c, \mathrm{st})$
05 $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
06 return $\sigma := (w, z)$
```

```
$\mathsf{simSignature}_4(m, \phi)$
07 $c \leftarrow_\$ \mathcal{C}$
08 $(w, z) \leftarrow \mathsf{Sim}(pk, c)$
09 Parse $(\phi_w, \phi_{\mathrm{st}}) := \phi$
10 if $\phi_w \neq Id$                          // $\phi$ targets $w$
11    $(\hat{w}, \hat{m}, \hat{pk}) := \phi(w, m, pk)$
12    $\mathsf{H} := \mathsf{H}^{(\hat{w}, \hat{m}, \hat{pk}) \mapsto c}$
13    $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
14 else
15    if $\phi_{\mathrm{st}} \neq Id$               // $\phi$ targets st
16       $I \leftarrow \mathsf{DeriveSet}(c)$
17       Parse $(\mathrm{st}_i)_{i \in I} := z$
18       $z := (\phi_{\mathrm{st},i}(\mathrm{st}_i))_{i \in I}$
19    $\mathsf{H} := \mathsf{H}^{(w, m, pk) \mapsto c}$
20    $\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{m\}$
21 return $\sigma := (w, z)$
```

Fig. 3.16: Original oracle FAULTSIGN for the case that $i = 4$, and signature simulation algorithm $\mathsf{simSignature}_4$ for the proof of Lemma 3.3.9.

If fault function ϕ is targeted at w, the situation is essentially the same as for fault index 5, and thus, the simulation strategy is identical to that of $\mathsf{simSignature}_5$ (see Section 3.3.3). If fault function ϕ is targeted at st, the situation is essentially the same as for fault index 7, and thus, the simulation strategy is identical to that of

$\mathsf{simSignature}_7$ (see Section 3.3.6). Putting both cases together, we obtain

$$|\Pr[G_4^{\mathsf{A}} = 1] - \Pr[G_5^{\mathsf{A}} = 1]| \leq q_{S,6} \cdot \Delta_{\mathsf{sHVZK}} + \frac{3q_{S,6}}{2} \sqrt{(q_H + q_S + 1) \cdot 2\gamma(\mathsf{Commit})} \ .$$

3.3.8 UF-CMA$_0$ *adversary for game G_5, for $\mathcal{F} = \{4, 5, 6, 7, 9\}$ (Proof of Lemma 3.3.10)*

Recall that in game G_5, faulty signatures are simulated for all indices $i \in \{4, 5, 6, 7, 9\}$. For adversaries against the UF-F$_{\{4,5,6,7,9\}}$-CMA security of SIG, the game derives all oracle answers by a call to one of the simulated oracles $\mathsf{simSignature}_i(m, \phi)$. To prove Lemma 3.3.10, observe that we can now extend adversary B defined in Figure 3.14 such that it is capable to perfectly simulate game G_5 by running the simulations, and simulating the random oracle to A, accordingly. (I.e., B runs A with oracle access to H$'$ that is first set to H, and that gets reprogrammed, with B keeping track of the classical queries to FAULTSIGN.)

Again, A can not win if m^* was a query to FAULTSIGN, hence a valid signature is also valid in B's UF-CMA$_0$ game and

$$\Pr[G_5^{\mathsf{A}} \Rightarrow 1] \leq \mathrm{Adv}^{\mathsf{UF\text{-}CMA}_0}_{\mathsf{FS[ID,H]}}(\mathsf{B}) \ .$$

3.3.9 From UF-F-CMA *to* UF-N-F-CMA *(Proof of Theorem 3.3.3)*

Let A be an adversary against the UF-N-F$_{\mathcal{F}'}$-CMA security of $\mathsf{SIG}' = \mathsf{R2H}[\mathsf{SIG}, \mathsf{G}]$ for $\mathcal{F}' := \mathcal{F} \cup \{1\}$, issuing at most q_S queries to N-FAULTSIGN, at most q_{H} queries to H, and at most q_{G} queries to G. In the random oracle model, the proof would work as follows: Either G is never queried on any faulted version of sk, or it is. In the case that such query does not exist, the UF-N-F$_{\mathcal{F}'}$-CMA experiment is completely simulatable by a reduction against the UF-F$_{\mathcal{F}}$-CMA security of the underlying scheme SIG, as the signing randomness looks uniformly random to the adversary. (Note that we made the assumption that A issues no query (m, n) to N-FAULTSIGN more than once.) In the case that such a query $\phi(sk)$ exists, it can be used to break UF-F$_{\mathcal{F}}$-CMA security by going over all possible secret key candidates, i.e., by going over all bit-flip functions, and checking whether any of those candidates can be used to generate a valid signature.

In principle, our QROM proof does the same. Consider the sequence of games given in Figure 3.17: We decouple the signing randomness from the secret key in game G_1. Again, game G_1 can be simulated by a reduction B_1 against the UF-F$_{\mathcal{F}}$-CMA security of the underlying scheme SIG. To upper bound the distance between games G_0 and G_1, we will use Theorem 1.3.5. (In order to give a more detailed description of how Theorem 1.3.5 can be used, we "zoom in" and give two intermediate helper games $G_{1/3}$ and $G_{2/3}$.) Applying Theorem 1.3.5, we can upper bound the distance between games G_0 and G_1 in terms of the probability that measuring a random query to G yields $\phi(sk)$. We then give a reduction B_2 that wins whenever the latter happens, with the same strategy as in the ROM sketch.

```
Games $G_0$ - $G_1$
$(pk, sk) \leftarrow \mathsf{IG}(par)$
$(m^*, \sigma^*) \leftarrow \mathsf{A}^{\text{N-FAULTSIGN}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk)$
if $m^* \in \mathfrak{L}_{\mathcal{M}}$ return 0
Parse $(w^*, z^*) := \sigma^*$
$c^* := \mathsf{H}(w^*, m^*)$
return $\mathsf{V}(pk, w^*, c^*, z^*)$

N-FAULTSIGN$(m, n, i \in \mathcal{F}', \phi)$
if $i = 1$
    $f_1 := \phi$
    $r := \mathsf{G}(f_1(sk), m, n)$                //$G_0$
    $r \leftarrow_\$ \mathcal{R}_{\mathsf{Sign}}$    //$G_1$
    $\sigma \leftarrow$ getSignature$(m, r, 2, Id)$
else
    $r := \mathsf{G}(sk, m, n)$                     //$G_0$
    $r \leftarrow_\$ \mathcal{R}_{\mathsf{Sign}}$    //$G_1$
    $\sigma \leftarrow$ getSignature$(m, r, i, \phi)$
return $\sigma$

getSignature$(m, r, i, \phi)$
$f_i := \phi$ and $f_j := Id \forall j \neq i$
$(w, \mathrm{st}) \leftarrow \mathsf{Commit}(sk; r)$
$(w, \mathrm{st}) := f_4(w, \mathrm{st})$
$(\hat{w}, \hat{m}, \hat{pk}) := f_5(w, m, pk)$
$c := f_6(\mathsf{H}(\hat{w}, \hat{m}, \hat{pk}))$
$z \leftarrow \mathsf{Respond}(f_7(sk, c, \mathrm{st}))$
$\mathfrak{L}_{\mathcal{M}} := \mathfrak{L}_{\mathcal{M}} \cup \{\hat{m}\}$
return $\sigma := f_9(w, z)$
```

Fig. 3.17: Games G_0 - G_1 for the proof of Theorem 3.3.3. Helper method getSignature is internal and cannot be accessed directly by A.

Game G_0. The (purely conceptual) difference between game G_0 and the original UF-N-F-CMA game is that after computing the signing randomness according to SIG$'$, we outsource the rest of the signature computation to helper method getSignature. In the case that $i = 1$, getSignature is executed with index 2 and Id, as the rest of the signature generation is unfaulted.

$$\mathrm{Adv}^{\mathsf{UF\text{-}N\text{-}F}_{\mathcal{F}'}\text{-}\mathsf{CMA}}_{\mathsf{SIG}'}(\mathsf{A}) = \Pr[G_0^{\mathsf{A}} \Rightarrow 1] \ .$$

GAME G_1. In game G_1, we re-randomise the Commit algorithm by letting $r \leftarrow_{\$} \mathcal{R}_{\mathsf{Sign}}$ instead of $r := \mathsf{G}(f_1(sk), m, n)$, see lines 10 and 14. To upper bound $\Pr[G_1^{\mathsf{A}} \Rightarrow 1]$, consider $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$ Adversary B_1 given in Figure 3.18. Adversary B_1 has access to the faulty signing oracle FAULTSIGN that is provided by game $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$, and that covers all faults except the ones that would have occurred with respect to index 1, i.e., the ones that fault the secret key as input to G. Due to our change described above, however, randomness r is drawn independently of sk in game G_1, hence the Commit algorithm is randomised. The output of FAULTSIGN therefore allows B_1 to perfectly simulate game G_1 to A. Furthermore, any valid forgery game G_1 is also a valid forgery in B_1's $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$ game. Hence,

$$\Pr[G_1^{\mathsf{A}} \Rightarrow 1] \leq \mathrm{Adv}^{\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}}_{\mathsf{SIG}}(\mathsf{B}_1) \ .$$

```
Adversary $\mathsf{B}_1^{|\mathsf{H}\rangle}(pk)$
$(m^*, \sigma^*) \leftarrow \mathsf{A}^{\text{N-FAULTSIGN}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk)$
return $(m^*, \sigma^*)$

N-FAULTSIGN$(m, n, i \in \mathcal{F}', \phi)$
if $i = 1$
    $\sigma \leftarrow$ FAULTSIGN$(m, 2, Id)$
else $\sigma \leftarrow$ FAULTSIGN$(m, i, \phi)$
return $\sigma$
```

Fig. 3.18: $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$ Adversary B_1, with access to its own faulty signing oracle FAULTSIGN, for the proof of Theorem 3.3.3.

It remains to upper bound $|\Pr[G_0^{\mathsf{A}} = 1] - \Pr[G_1^{\mathsf{A}} = 1]|$. To this end, we will make use of the query extraction variant of one-way to hiding (see Theorem 1.3.5). In order to keep our proof as accessible as possible, we introduce intermediate helper games $G_{1/3}$ and $G_{2/3}$ in Figure 3.19.

As a preparation, we first consider intermediate game $G_{1/3}$, in which we completely replace random oracle G with another random oracle G' (see lines 02, 15 and 19), where G' is defined as follows: Let $\mathfrak{L}_{sk}$ denote the set of secret keys that could occur by faulting the secret key with a one-bit fault injection. We let G' concur with G for all inputs such that the input secret key is not in $\mathfrak{L}_{sk}$, i.e., for all $sk' \notin \mathfrak{L}_{sk}$ and all $(m, n) \in \mathcal{M} \times \mathcal{N}$, we let $\mathsf{G}'(sk', m, n) := \mathsf{G}(sk', m, n)$. We can then complete it to a random oracle on $\mathcal{SK} \times \mathcal{M} \times \mathcal{N}$ by picking another random oracle $\mathsf{G}'' : \mathfrak{L}_{sk} \times \mathcal{M} \times \mathcal{N}$, and letting $\mathsf{G}'(sk', m, n) := \mathsf{G}''(sk', m, n)$ for all $sk' \in \mathfrak{L}_{sk}$ and all $(m, n) \in \mathcal{M} \times \mathcal{N}$. Since G'

```
Games $G_{1/3}$ - $G_1$
01 $(pk, sk) \leftarrow \mathsf{IG}(par)$
02 $\mathsf{O} := \mathsf{G}'$                                   // $G_{1/3}$
03 $\mathsf{O} := \mathsf{G}$                                    // $G_{2/3}$-$G_1$
04 $(m^*, \sigma^*) \leftarrow \mathsf{A}^{\text{N-FAULTSIGN},|\mathsf{H}\rangle,|\mathsf{O}\rangle}(pk)$

05 if $m^* \in \mathfrak{L}_{\mathcal{M}}$ return $0$
06 Parse $(w^*, z^*) := \sigma^*$
07 $c^* := \mathsf{H}(w^*, m^*)$
08 return $\mathsf{V}(pk, w^*, c^*, z^*)$

Extractor $\mathsf{E}^{|\mathsf{O}\rangle,|\mathsf{H}\rangle}(pk, sk, \mathfrak{L}_{\mathsf{G}'})$
09 $j \leftarrow_\$ \{1, \cdots, q_{\mathsf{G}}\}$
10 Run $\mathsf{A}^{\text{N-FAULTSIGN},|\mathsf{H}\rangle,|\mathsf{O}\rangle}(pk)$
   until $j$th query to $\mathsf{O}$
11 $(sk', m, n) \leftarrow$ Measure query
   input reg.
12 return $sk'$

N-FAULTSIGN$(m, n, i \in \mathcal{F}', \phi)$
13 if $i = 1$
14    $f_1 := \phi$
15    $r := \mathsf{G}'(f_1(sk), m, n)$              // $G_{1/3}$, $G_{2/3}$, $\mathsf{E}$
16    $r \leftarrow_\$ \mathcal{R}_{\mathsf{Sign}}$                         // $G_1$
17    $\sigma \leftarrow$ GETSIGNATURE$(m, r, 2, Id)$
18 else
19    $r := \mathsf{G}'(sk, m, n)$                   // $G_{1/3}$, $G_{2/3}$, $\mathsf{E}$
20    $r \leftarrow_\$ \mathcal{R}_{\mathsf{Sign}}$                         // $G_1$
21    $\sigma \leftarrow$ GETSIGNATURE$(m, r, i, \phi)$
22 return $\sigma$
```

Fig. 3.19: Intermediate helper games $G_{1/3}$ and $G_{2/3}$, justifying the game-hop from game G_0 to G_1, and query extractor E. Alternative oracle G' (see lines 02, 15 and 19) is constructed by letting $\mathsf{G}'(sk', m, n) := \mathsf{G}(sk', m, n)$ for all input (sk', m, n) such that sk' cannot result from faulting sk, and completing G' randomly. Helper method GETSIGNATURE remains as in Figure 3.17.

still is a random oracle, and since we also use G' to derive the signing randomness, this change is purely conceptual and

$$\Pr[G_0^{\mathsf{A}} \Rightarrow 1] = \Pr[G_{1/3}^{\mathsf{A}} \Rightarrow 1] \enspace .$$

In game $G_{2/3}$, we prepare to rid the randomness generation of the secret key: We switch back to providing A with oracle access to the original random oracle G, but we keep using G' to derive the signing randomness. After this change, oracle G' is not directly accessible by A anymore, but only indirectly via the signing queries. Since we assume that A issues no query (m, n) to N-FAULTSIGN more than once, we can also replace these values with freshly sampled randomness as in game G_1, i.e.,

$$\Pr[G_{2/3}^{\mathsf{A}} \Rightarrow 1] = \Pr[G_1^{\mathsf{A}} \Rightarrow 1] \enspace .$$

So far, we have shown that

$$\mathrm{Adv}_{\mathsf{SIG}'}^{\mathsf{UF\text{-}N\text{-}F}_{\mathcal{F}'}\mathsf{\text{-}CMA}}(\mathsf{A}) \leq \mathrm{Adv}_{\mathsf{SIG}}^{\mathsf{UF\text{-}F}_{\mathcal{F}}\mathsf{\text{-}CMA}}(\mathsf{B}_1) + |\Pr[G_{1/3}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_{2/3}^{\mathsf{A}} \Rightarrow 1]| \enspace .$$

In order to upper bound $|\Pr[G_{1/3}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_{2/3}^{\mathsf{A}} \Rightarrow 1]|$, we invoke Theorem 1.3.5: Distinguishing between the two games can be reduced to extracting one of the faulted secret keys from the queries to G. To make this claim more formal, consider the query extractor E from Theorem 1.3.5, whose explicit description we give in Figure 3.19. Extractor E is run with access to oracle $\mathsf{O} \in \{\mathsf{G}, \mathsf{G}'\}$, which it will forward to A. It runs A until A's ith oracle query to O, measures the query input register, and thereby obtains a triplet (sk', m, n) of classical input values. Since we are only interested in points where G and G' differ, it is sufficient to let E output the secret key candidate sk'. Note that E is able to simulate the signing oracle regardless of which oracle O it has access to: Recall that Theorem 1.3.5 makes no assumption on the runtime of the query extractor, nor on the size of its input. Hence, the alternative oracle G' can simply be encoded as part of the extractor's input, which we denote by adding $\mathfrak{L}_{\mathsf{G}'}$ to E's input. Since E perfectly simulates game $G_{1/3}$ if $\mathsf{O} = \mathsf{G}'$, and game $G_{1/3}$ if $\mathsf{O} = \mathsf{G}$, Theorem 1.3.5 yields

$$|\Pr[G_{1/3}^{\mathsf{A}} \Rightarrow 1] - \Pr[G_{2/3}^{\mathsf{A}} \Rightarrow 1]| \leq 2q_{\mathsf{G}} \cdot \sqrt{\Pr[sk' \in \mathfrak{L}_{sk} : sk' \leftarrow \mathsf{E}^{|\mathsf{G}\rangle,|\mathsf{H}\rangle}(pk, sk, \mathsf{G}')]} \ .$$

It remains to bound the success probability of the extractor E. At this point, the signing randomness is independent of G. We can hence also replace E with an extractor E' that uses freshly sampled randomness to sign, without any change in the extraction probability. (Again, we require that A issues no query (m, n) to N-FAULTSIGN more than once.)

To bound the success probability of E', consider $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$ Adversary B_2, which is given in Figure 3.20. Like B_1, Adversary B_2 has access to the faulty signing oracle FAULTSIGN provided by game $\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}$, and it uses FAULTSIGN to answer signing queries. B_2 perfectly simulates the view of A when A is run by extractor E', and the probability that E' returns some $sk' \in \mathfrak{L}_{sk}$ is hence exactly the probability that B_2 obtains some $sk' \in \mathfrak{L}_{sk}$ by measuring in line 03. After running A until the jth query to G, and extracting a secret key candidate sk' from this query, B_2 computes the list $\mathfrak{L}_{sk'}$ of candidate secret keys that could occur by faulting sk' with a one-bit fault injection (including the identity function). Since bit flips are involutory, and set-bit functions can be reversed by set-bit functions, $sk' \in \mathfrak{L}_{sk}$ iff $sk \in \mathfrak{L}_{sk'}$. Hence, if B_2 obtains some $sk' \in \mathfrak{L}_{sk}$ by measuring, then B_2 will encounter sk during execution of its loop and therefore generate a valid signature.

$$\Pr[sk' \in \mathfrak{L}_{sk} : sk' \leftarrow \mathsf{E}'^{|\mathsf{G}\rangle,|\mathsf{H}\rangle}(pk, sk, \mathsf{G}')] \leq \mathrm{Adv}_{\mathsf{SIG}}^{\mathsf{UF\text{-}F}_{\mathcal{F}}\text{-}\mathsf{CMA}}(\mathsf{B}_2) \ .$$

```
Adversary $\mathsf{B_2}^{|\mathsf{H}\rangle}(pk)$
01 $j \leftarrow_\$ \{1, \cdots, q_\mathsf{G}\}$
02 Run $\mathsf{A}^{\text{N-FAULTSIGN}, |\mathsf{H}\rangle, |\mathsf{G}\rangle}(pk)$
   until $j$th query to $\mathsf{G}$
03 $sk' \leftarrow$ Measure query input register
04 $m^* \leftarrow_\$ \mathcal{M} \setminus \mathfrak{L}'_\mathcal{M}$
05 for $sk'' \in \mathfrak{L}_{sk'}$
06    $\sigma \leftarrow \mathsf{Sign}(sk'', m)$
07    if $\mathsf{Vrfy}(m, \sigma) = 1$
08       return $(m, \sigma)$
09 return $\perp$
```

```
N-FAULTSIGN$(m, n, i \in \mathcal{F}', \phi)$
10 if $i = 1$
11    $\sigma \leftarrow$ FAULTSIGN$(m, 2, Id)$
12 else $\sigma \leftarrow$ FAULTSIGN$(m, i, \phi)$
13 if $i = 5$ and $\phi$ affects $m$
14    $\mathfrak{L}'_\mathcal{M} := \mathfrak{L}'_\mathcal{M} \cup \{\phi_m(m)\}$
15 else $\mathfrak{L}'_\mathcal{M} := \mathfrak{L}'_\mathcal{M} \cup \{m\}$
16 return $\sigma$
```

Fig. 3.20: UF-F$_\mathcal{F}$-CMA Adversary B_2, with access to its own faulty signing oracle FAULTSIGN, for the proof of Theorem 3.3.3. List $\mathfrak{L}_{sk'}$ (see line 05) denotes the list of secret keys that could occur by faulting sk' with a one-bit fault injection.

Bibliography

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 418–433, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Heidelberg, Germany. 33

[ABB$^+$20] Erdem Alkim, Paulo S. L. M. Barreto, Nina Bindel, Juliane Krämer, Patrick Longa, and Jefferson E. Ricardini. The lattice-based digital signature scheme qTESLA. In Mauro Conti, Jianying Zhou, Emiliano Casalicchio, and Angelo Spognardi, editors, *ACNS 20, Part I*, volume 12146 of *LNCS*, pages 441–460, Rome, Italy, October 19–22, 2020. Springer, Heidelberg, Germany. 18

[ABD$^+$18] C. Aguilar-Melchor, O. Blazy, J. Deneuville, P. Gaborit, and G. Zémor. Efficient encryption from random quasi-cyclic codes. *IEEE Transactions on Information Theory*, 64(5):3927–3943, 2018. 18

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 143–158, San Francisco, CA, USA, April 8–12, 2001. Springer, Heidelberg, Germany. 13, 53, 60

[ACFK17] Benedikt Auerbach, David Cash, Manuel Fersch, and Eike Kiltz. Memory-tight reductions. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 101–132, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 12

[ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In *25th USENIX Security Symposium,* *USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pages 327–343, 2016. 54

[AFLT12] Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-secure signatures from lossy identification schemes. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 572–590, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany. 138

[AHU19] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 269–295, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 42, 43, 44, 45, 136

[AMRS20] Gorjan Alagic, Christian Majenz, Alexander Russell, and Fang Song. Quantum-access-secure message authentication via blind-unforgeability. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 788–817, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany. 50

[AOP$^+$17] Martin R. Albrecht, Emmanuela Orsini, Kenneth G. Paterson, Guy Peer, and Nigel P. Smart. Tightly secure ring-LWE based key encapsulation with short ciphertexts. In Simon N. Foley, Dieter Gollmann, and Einar Snekkenes, editors, *ESORICS 2017, Part I*, volume 10492 of *LNCS*, pages 29–46, Oslo, Norway, September 11–15, 2017. Springer, Heidelberg, Germany. 60, 61

[AOTZ20] Diego F. Aranha, Claudio Orlandi, Akira Takahashi, and Greg Zaverucha. Security of hedged Fiat-Shamir signatures under fault attacks. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 644–674, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany. 34, 35, 137, 139, 154, 155, 156, 157, 158

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *55th FOCS*, pages 474–483, Philadelphia, PA, USA, October 18–21, 2014. IEEE Computer Society Press. 46

[BBC$^+$98] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. In *39th FOCS*, pages 352–361, Palo Alto, CA, USA, November 8–11, 1998. IEEE Computer Society Press. 40

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274, Bruges, Belgium, May 14–18, 2000. Springer, Heidelberg, Germany. 12

[BBO07] Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Heidelberg, Germany. 57

[BCD$^+$16] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! Practical, quantum-secure key exchange from LWE. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 1006–1018, Vienna, Austria, October 24–28, 2016. ACM Press. 54

[BCL$^+$19] Daniel J. Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, and Wen Wang. Classic McEliece. Technical report, National Institute of Standards and Technology, 2019. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`. 18

[BCLv16] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU prime. Cryptology ePrint Archive, Report 2016/461, 2016. `http://eprint.iacr.org/2016/461`. 18

[BCNS15] Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In *2015 IEEE Symposium on Security and Privacy*, pages 553–570, San Jose, CA, USA, May 17–21, 2015. IEEE Computer Society Press. 54

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69, Seoul, South Korea, December 4–8, 2011. Springer, Heidelberg, Germany. 17, 40, 136

[BDK$^+$17] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, and Damien Stehlé. CRYSTALS – Kyber: a CCA-secure module-lattice-based KEM. Cryptology ePrint Archive, Report 2017/634, 2017. http://eprint.iacr.org/2017/634. 54, 106

[BFK$^+$12] Romain Bardou, Riccardo Focardi, Yusuke Kawamoto, Lorenzo Simionato, Graham Steel, and Joe-Kai Tsay. Efficient padding oracle attacks on cryptographic hardware. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 608–625, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany. 13

[BFM15] Christina Brzuska, Pooya Farshim, and Arno Mittelbach. Random-oracle uninstantiability from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 428–455, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany. 14

[BHH$^+$19] Nina Bindel, Mike Hamburg, Kathrin Hövelmanns, Andreas Hülsing, and Edoardo Persichetti. Tighter proofs of CCA security in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 61–90, Nuremberg, Germany, December 1–5, 2019. Springer, Heidelberg, Germany. 21, 30, 42, 43, 56, 106, 121, 122, 128, 129, 136

[BHSV98] Mihir Bellare, Shai Halevi, Amit Sahai, and Salil P. Vadhan. Many-to-one trapdoor functions and their relation to public-key cryptosystems. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 283–298, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Heidelberg, Germany. 57

[BI17] Subhadeep Banik and Takanori Isobe. Some cryptanalytic results on lizard. Cryptology ePrint Archive, Report 2017/346, 2017. http://eprint.iacr.org/2017/346. 106

[BJL17] Mihir Bellare, Joseph Jaeger, and Julia Len. Better than advertised: Improved collision-resistance guarantees for MD-based hash functions. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 891–906, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press. 12

[BL20] Anne Broadbent and Sébastien Lord. Uncloneable Quantum Encryption via Oracles. In Steven T. Flammia, editor, *TQC 2020*, LIPIcs, pages 4:1–4:22, Dagstuhl, Germany, 2020. 136

[Ble98] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 1–12, Santa Barbara, CA, USA, August 23–27, 1998. Springer, Heidelberg, Germany. 11, 13

[BLK00] Joonsang Baek, Byoungcheon Lee, and Kwangjo Kim. Secure length-saving ElGamal encryption under the computational Diffie-Hellman assumption. In Ed Dawson, Andrew Clark, and Colin Boyd, editors, *ACISP 00*, volume 1841 of *LNCS*, pages 49–58, Brisbane, Queensland, Australia, July 10–12, 2000. Springer, Heidelberg, Germany. 60

[BP18] Daniel J. Bernstein and Edoardo Persichetti. Towards KEM unification. Cryptology ePrint Archive, Report 2018/526, 2018. https://eprint.iacr.org/2018/526. 32, 121

[BPS16] Mihir Bellare, Bertram Poettering, and Douglas Stebila. From identification to signatures, tightly: A framework and generic transforms. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 435–464, Hanoi, Vietnam, December 4–8, 2016. Springer, Heidelberg, Germany. 37

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press. 13

[BR95] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111, Perugia, Italy, May 9–12, 1995. Springer, Heidelberg, Germany. 13

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 399–416, Saragossa, Spain, May 12–16, 1996. Springer, Heidelberg, Germany. 12, 13

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany. 23

[BR09] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 407–424, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany. 12

[BS20] Nina Bindel and John M. Schanck. Decryption failure is more likely after success. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 206–225, Paris, France, April 15–17 2020. Springer, Heidelberg, Germany. 30, 54, 55

[BT16] Mihir Bellare and Björn Tackmann. Nonce-based cryptography: Retaining security when randomness fails. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 729–757, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany. 37

[BV17] Nir Bitansky and Vinod Vaikuntanathan. A note on perfect correctness by derandomization. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 592–606, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany. 30

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218, Dallas, TX, USA, May 23–26, 1998. ACM Press. 14

[CHJ$^+$02] Jean-Sébastien Coron, Helena Handschuh, Marc Joye, Pascal Paillier, David Pointcheval, and Christophe Tymen. GEM: A generic chosen-ciphertext secure encryption method. In Bart Preneel, editor, *CT-RSA 2002*, volume 2271 of *LNCS*, pages 263–276, San Jose, CA, USA, February 18–22, 2002. Springer, Heidelberg, Germany. 13, 53, 60

[CHR$^+$16] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. From 5-pass MQ-based identification to MQ-based signatures. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 135–165, Hanoi, Vietnam, December 4–8, 2016. Springer, Heidelberg, Germany. 18

[CKLS16] Jung Hee Cheon, Duhyeong Kim, Joohee Lee, and Yongsoo Song. Lizard: Cut off the tail! practical post-quantum public-key encryption from lwe and lwr. Cryptology ePrint Archive, Report 2016/1126, 2016. `http://eprint.iacr.org/2016/1126`. 54

[CKS08] David Cash, Eike Kiltz, and Victor Shoup. The twin Diffie-Hellman problem and applications. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 127–145, Istanbul, Turkey, April 13–17, 2008. Springer, Heidelberg, Germany. 60

[CKS09] David Cash, Eike Kiltz, and Victor Shoup. The twin Diffie-Hellman problem and applications. *Journal of Cryptology*, 22(4):470–504, October 2009. 60

[CMP20] Andrea Coladangelo, Christian Majenz, and Alexander Poremba. Quantum copy-protection of compute-and-compare programs in the quantum random oracle model. arXiv 2009.13865, 2020. 136

[CS03] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003. 29

[Den03] Alexander W. Dent. A designer's guide to KEMs. In Kenneth G. Paterson, editor, *9th IMA International Conference on Cryptography and Coding*, volume 2898 of *LNCS*, pages 133–151, Cirencester, UK, December 16–18, 2003. Springer, Heidelberg, Germany. 27, 53, 59, 60, 121

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 356–383, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 136

[DKL$^+$18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR TCHES*, 2018(1):238–268, 2018. `https://tches.iacr.org/index.php/TCHES/article/view/839`. 18, 139

[DNR04] Cynthia Dwork, Moni Naor, and Omer Reingold. Immunizing encryption schemes from decryption errors. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 342–360, Interlaken, Switzerland, May 2–6, 2004. Springer, Heidelberg, Germany. 30

[DVV18] Jan-Pieter D'Anvers, Frederik Vercauteren, and Ingrid Verbauwhede. On the impact of decryption failures on the security of LWE/LWR based schemes. Cryptology ePrint Archive, Report 2018/1089, 2018. `https://eprint.iacr.org/2018/1089`. 54, 55

[DXL12] Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. Cryptology ePrint Archive, Report 2012/688, 2012. `http://eprint.iacr.org/2012/688`. 54

[ES15] Edward Eaton and Fang Song. Making Existential-unforgeable Signatures Strongly Unforgeable in the Quantum Random-oracle Model. In *TQC 2015*, LIPIcs, 2015. 136, 141

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany. 13, 25, 53, 59

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013. 13, 53

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer, Heidelberg, Germany. 35, 37

[FSXY12] Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 467–484, Darmstadt, Germany, May 21–23, 2012. Springer, Heidelberg, Germany. 20

[FSXY13] Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama. Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In Kefei Chen, Qi Xie, Weidong

Qiu, Ninghui Li, and Wen-Guey Tzeng, editors, *ASIACCS 13*, pages 83–94, Hangzhou, China, May 8–10, 2013. ACM Press. 20

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th ACM STOC*, pages 365–377, San Francisco, CA, USA, May 5–7, 1982. ACM Press. 12

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270 – 299, 1984. 12

[GMMV05] David Galindo, Sebastià Martín, Paz Morillo, and Jorge L. Villar. Fujisaki-okamoto hybrid encryption revisited. *Int. J. Inf. Sec.*, 4(4):228–241, 2005. 60

[GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *17th ACM STOC*, pages 291–304, Providence, RI, USA, May 6–8, 1985. ACM Press. 35

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, April 1988. 36

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 212–219, New York, NY, USA, 1996. Association for Computing Machinery. 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, Part I*, volume 10677 of *LNCS*, pages 341–371, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany. 19, 21, 42, 55, 56

[HKSU20] Kathrin Hövelmanns, Eike Kiltz, Sven Schäge, and Dominique Unruh. Generic authenticated key exchange in the quantum random oracle model. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 389–422, Edinburgh, UK, May 4–7, 2020. Springer, Heidelberg, Germany. 19, 20, 46, 56

[HRS16] Andreas Hülsing, Joost Rijneveld, and Fang Song. Mitigating multi-target attacks in hash-based signatures. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 387–416, Taipei, Taiwan, March 6–9, 2016. Springer, Heidelberg, Germany. 45, 46, 136, 141

[JAC+19] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, and Geovandro Pereira. SIKE. Technical report, National Institute of Standards and Technology, 2019. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`. 18

[JZC+18] Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisited. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 96–125, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. 56

[JZM19a] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. Key encapsulation mechanism with explicit rejection in the quantum random oracle model. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 618–645, Beijing, China, April 14–17, 2019. Springer, Heidelberg, Germany. 56

[JZM19b] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. Cryptology ePrint Archive, Report 2019/134, 2019. `https://eprint.iacr.org/2019/134`. 30, 56, 128

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. Chapman and Hall/CRC, 2nd edition, 2014. 12, 24, 26

[KLS17] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. Manuscript, 2017. 46

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018,* *Part III*, volume 10822 of *LNCS*, pages 552–586, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. 40, 136, 137, 138, 139

[KM03] Eike Kiltz and John Malone-Lee. A general construction of IND-CCA2 secure public key encryption. In Kenneth G. Paterson, editor, *9th IMA International Conference on Cryptography and Coding*, volume 2898 of *LNCS*, pages 152–166, Cirencester, UK, December 16–18, 2003. Springer, Heidelberg, Germany. 60

[KSS$^+$20] Veronika Kuchta, Amin Sakzad, Damien Stehlé, Ron Steinfeld, and Shifeng Sun. Measure-rewind-measure: Tighter quantum random oracle model proofs for one-way to hiding and CCA security. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 703–728, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany. 30, 42, 43, 56, 57, 128, 136

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany. 60

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-LWE cryptography. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 35–54, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany. 60

[LS19] Vadim Lyubashevsky and Gregor Seiler. NTTRU: Truly fast NTRU using NTT. *IACR TCHES*, 2019(3):180–201, 2019. `https://tches.iacr.org/index.php/TCHES/article/view/8293`. 25, 30, 120

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616, Tokyo, Japan, December 6–10, 2009. Springer, Heidelberg, Germany. 139

[LZ19] Qipeng Liu and Mark Zhandry. Revisiting post-quantum Fiat-Shamir. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 326–355, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 136

[NAB$^+$17] Michael Naehrig, Erdem Alkim, Joppe Bos, Léo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions. 106

[NC11] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, USA, 10th edition, 2011. 38, 39, 47

[NIS17] NIST. National institute for standards and technology. postquantum crypto project, 2017. http://csrc.nist.gov/groups/ST/post-quantum-crypto/. 15, 139

[NIS20] NIST. Status report on the second round of the nist post-quantum cryptography standardization process. NISTIR 8309, 2020. https://doi.org/10.6028/NIST.IR.8309. 139

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437, Baltimore, MD, USA, May 14–16, 1990. ACM Press. 13

[OP01] Tatsuaki Okamoto and David Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 159–175, San Francisco, CA, USA, April 8–12, 2001. Springer, Heidelberg, Germany. 13, 53, 60

[Pei14] Chris Peikert. Lattice cryptography for the internet. Cryptology ePrint Archive, Report 2014/070, 2014. http://eprint.iacr.org/2014/070. 54

[Per12] Edoardo Persichetti. *Improving the efficiency of code-based cryptography*. PhD thesis, 2012. 58

[PS96a] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 252–265, Kyongju, Korea, November 3–7, 1996. Springer, Heidelberg, Germany. 14

[PS96b] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398, Saragossa, Spain, May 12–16, 1996. Springer, Heidelberg, Germany. 13

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93, Baltimore, MA, USA, May 22–24, 2005. ACM Press. 60

[Riv] Ronald L. Rivest. *Cryptography*, volume 1, chapter 13, pages 717–755. Elsevier. 11

[RS92] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 433–444, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Heidelberg, Germany. 13, 29, 53

[Sho94] P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994. 15

[Sho04a] Victor Shoup. ISO 18033-2: An emerging standard for public-key encryption. `http://shoup.net/iso/std6.pdf`, December 2004. Final Committee Draft. 60

[Sho04b] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. `http://eprint.iacr.org/2004/332`. 23, 24

[SXY18] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 520–551, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. 28, 40, 56, 104, 105, 106, 107, 110, 111, 115, 121, 136, 157

[TU16] Ehsan Ebrahimi Targhi and Dominique Unruh. Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 192–216, Beijing, China, October 31 – November 3, 2016. Springer, Heidelberg, Germany. 54, 88, 93, 97

[Unr14a] Dominique Unruh. Quantum position verification in the random oracle model. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 1–18, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany. 136, 141

[Unr14b] Dominique Unruh. Revocable quantum timed-release encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 129–146, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. 41, 43, 89, 90, 136

[Wat01] John Watrous. Quantum algorithms for solvable groups. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, STOC '01, page 60–67, New York, NY, USA, 2001. Association for Computing Machinery. 15

[WMHT18] Yuyu Wang, Takahiro Matsuda, Goichiro Hanaoka, and Keisuke Tanaka. Memory lower bounds of reductions revisited. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 61–90, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. 12

[YZ20] Takashi Yamakawa and Mark Zhandry. A note on separating classical and quantum random oracles. Cryptology ePrint Archive, Report 2020/787, 2020. https://eprint.iacr.org/2020/787. 135

[ZCD$^+$19] Greg Zaverucha, Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, Jonathan Katz, Xiao Wang, and Vladmir Kolesnikov. Picnic. technical report. National Institute of Standards and Technology, 2019. https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions. 18, 139

[Zha12a] Mark Zhandry. How to construct quantum random functions. In *53rd FOCS*, pages 679–687, New Brunswick, NJ, USA, October 20–23, 2012. IEEE Computer Society Press. 45

[Zha12b] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany. 40

[Zha19] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 239–268, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany. 47, 136, 137

Publications

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341—371, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany.

[BHH+19] Nina Bindel, Mike Hamburg, Kathrin Hövelmanns, Andreas Hülsing, and Edoardo Persichetti. Tighter proofs of CCA security in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part II*, volume 11892 of *LNCS*, pages 61—90, Nuremberg, Germany, December 1-–5, 2019. Springer, Heidelberg, Germany.

[HKSU20] Kathrin Hövelmanns, Eike Kiltz, Sven Schäge, and Dominique Unruh. Generic Authenticated Key Exchange in the quantum random oracle model. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 389—422, Edinburgh, UK, May 4-–7, 2020. Springer, Heidelberg, Germany.

[GHHM20] Alex B. Grilo, Kathrin Hövelmanns, Andreas Hülsing, Christian Majenz. Tight adaptive reprogramming in the QROM. To be published.