Efficient NIZKs and Signatures from Commit-and-Open Protocols in the QROM

Don, J.; Fehr, S.; Majenz, C.; Schaffner, C.

DOI
10.1007/978-3-031-15979-4_25

Publication date
2022

Document Version
Submitted manuscript

Published in
Advances in Cryptology – CRYPTO 2022

Efficient NIZKs and Signatures from Commit-and-Open Protocols in the QROM

Jelle Don[1], Serge Fehr[1,2], Christian Majenz[3], and Christian Schaffner[4,5]

[1] Centrum Wiskunde & Informatica (CWI), Amsterdam, Netherlands
[2] Mathematical Institute, Leiden University, Netherlands
[3] Cyber Security Section, Department of Applied Mathematics and Computer Science, Technical University of Denmark, Kgs. Lyngby, Denmark
[4] Informatics Institute, University of Amsterdam, Amsterdam, Netherlands
[5] QuSoft, Amsterdam, Netherlands
jelle.don@cwi.nl, serge.fehr@cwi.nl, chmaj@dtu.dk, c.schaffner@uva.nl

Abstract. Commit-and-open Σ-protocols are a popular class of protocols for constructing non-interactive zero-knowledge arguments and digital-signature schemes via the Fiat-Shamir transformation. Instantiated with hash-based commitments, the resulting non-interactive schemes enjoy tight online-extractability in the random oracle model. Online extractability improves the tightness of security proofs for the resulting digital-signature schemes by avoiding lossy rewinding or forking-lemma based extraction.

In this work, we prove tight online extractability in the quantum random oracle model (QROM), showing that the construction supports post-quantum security. First, we consider the default case where committing is done by element-wise hashing. In a second part, we extend our result to Merkle-tree based commitments. Our results yield a significant improvement of the provable post-quantum security of the digital-signature scheme Picnic.

Our analysis makes use of a recent framework by Chung et al. [CFHL21] for analysing quantum algorithms in the QROM using purely classical reasoning. Therefore, our results can to a large extent be understood and verified without prior knowledge of quantum information science.

1 Introduction

Some interactive proofs come with amazing properties like *zero-knowledge* which intuitively allows a prover to convince a verifier that she knows the witness to an NP-statement without giving away any information about this witness. Such zero-knowledge proofs of knowledge are some of the most fascinating objects in cryptography, and possibly in all of theoretical computer science. One might suspect that their "magic" is rooted in the fact that the prover and verifier run an *interactive* protocol with each other, and that this interaction causes the verifier to be convinced. Surprisingly, if the interactive proof is of suitable form, e.g. a Σ-protocol (i.e. a 3-round public-coin protocol), the Fiat-Shamir transformation [FS87] provides a natural way to remove the interaction from such protocols while preserving (most of) the security properties, resulting in *non-interactive zero-knowledge* proofs (NIZKs). The idea is to compute the challenge c as a hash $c = H(a)$ of the first message, rather than letting the verifier choose c. If the original Σ-protocol has additional soundness properties, the resulting NIZK after the Fiat-Shamir transformation is ideally suited to be turned into a *digital-signature scheme*, simply by hashing the message m to be signed together with the first message a in order to obtain the challenge c. The candidates Picnic [CDG+17] and Dilithium [DKL+18] in the ongoing NIST post-quantum cryptography competition follow this design paradigm.

This intuitive preservation of security properties under the Fiat-Shamir transformation can be formalized in the random-oracle model (ROM), where the hash function H is treated as a uniformly random function, and the security reduction gets *enhanced access* to anybody who queries the random oracle, by seeing which values are queried, and by possibly returning (random-looking) outputs. While this situation is conveniently easy to handle in a non-quantum world, complications arise in the context of post-quantum security. When studying the security of these non-quantum protocols against attackers equipped with large-enough quantum computers, it is natural to assume that such attackers have access to the public description of the employed hash function, and can therefore compute it in superposition on their quantum computers. Therefore, the proper notion of post-quantum security for random oracles

is the *quantum-accessible random-oracle model (QROM)* as introduced in [BDF+11]. Due to the difficulty of recording adversarial random-oracle queries in superposition (also referred to as the *recording barrier*), establishing post-quantum security in the QROM has turned out to be quite a bit more difficult compared to the regular ROM.

Previous results in [DFMS19] (and concurrently in [LZ19b]) establish that for any interactive Σ-protocol Π that is a proof of knowledge, the non-interactive $\mathsf{FS}[\Pi]$ is a proof of knowledge in the QROM. [DFM20] simplified the technical proof and extended these results to multi-round interactive proofs. However, the most desirable property from such a proof of knowledge is *online extractability*. Indeed, online extractability avoids *rewinding*, which typically causes a significant loss in the security reduction (see later for a comparison) and has other disadvantages. Thus, online extractability allows for the tightest security reductions.

Chailloux was the first to aim for showing online extractability of the Fiat-Shamir transformation in the QROM when considering the relevant class of *commit-and-open* (C&O) Σ-protocols and modelling the hash function used for the commitments (and for computing the challenge) as a random oracle. Indeed, the Fiat-Shamir transformation of such C&O Σ-protocols are known to be online extractable in the classical ROM (see e.g. discussion in [Fis05]). In a first attempt [Cha19], Chailloux tried to lift the argument to the quantum setting by means of Zhandry's compressed-oracle technique [Zha19], which offers a powerful approach for re-establishing ROM results in the QROM, that has been successful in many instances. Unfortunately, this first attempt contained a subtle flaw, which turned out to be unfixable, and despite changing the technical approach, the latest version [Cha21] of this work still contains an open gap in the proof, which is put as an assumption.[1]

In a recent article [DFMS21], online extractability of *interactive* C&O Σ-protocols Π in the QROM is established; the result applies as soon as Π satisfies some liberal notion of *special soundness*, which is typically satisfied. As pointed out in Appendix E of [DFMS21], one can use previous results from [DFMS19, LZ19b, DFM20] to reduce the extractability of the resulting non-interactive protocol $\mathsf{FS}[\Pi]$ to the extractability of the interactive protocol Π. However, the resulting extraction error still scales as $O(\varepsilon/q^2)$, which results in a prohibitive loss for digital-signature schemes (see Table 1), leaving open the main question originally posed by Chailloux:

> *How to establish tight security reductions of the Fiat-Shamir transformation for commit-and-open Σ-protocols in the QROM?*

As the technical quantum details of Zhandry's compressed-oracle technique are rather complicated and only accessible for experts, a recent article by Chung, Fehr, Huang and Liao [CFHL21] attempts to give a comprehensive exposition of Zhandry's technique. In addition, they establish a framework that allows researchers without extensive quantum knowledge to still deploy the compressed-oracle technique (in certain cases), basically by reasoning about classical quantities only. In short, the punchline of [CFHL21] is that, if applicable, one can prove *quantum* query complexity lower bounds (think of collision finding, for instance) by means of the following recipe, which is an abstraction of the technique developed in a line of works started by Zhandry [Zha19, LZ19a, CGLQ20, HM21]. First, one considers the corresponding *classical* query complexity problem, analyzing it by simulating the random oracle using lazy sampling and showing that the database, which keeps track of the oracle queries and the responses, is unlikely to satisfy a certain property (e.g. to contain a collision) after a bounded number of queries. Then, one lifts the analysis to the quantum setting by plugging certain key observations from the classical analysis into generic theorems provided by the [CFHL21] framework.

1.1 Our Contributions

In this work, we slightly extend the framework from [CFHL21], and use it in a conceptually new (and arguably roundabout) way to establish strong and tight security statements for a large, popular class of

[1] Informally, quoting from [Cha21], the considered Assumption 2 is that the random oracle can be replaced with a random function of a particular form *"without harming too much the studied scheme"*. More formally, the security loss caused by the considered replacement is assumed to remain bounded by a given function of the number of oracle queries. This assumption is rather ad-hoc and non-standard in that it is very much tailored to the scheme and its proof. Furthermore, even though Assumption 2 is an assumption that could potentially be proven in future work, it is hard to judge whether proving the assumption is actually any easier than proving the security of the considered scheme *directly*, avoiding Assumption 2 — as a matter of fact, in this work we show that the latter is feasible, while Assumption 2 remains open.

non-interactive zero-knowledge proofs and digital signature schemes. In broad strokes, our contributions are threefold.

Online extractability for a class of NIZKs in the QROM. We prove online extractability of the Fiat-Shamir transformation in the QROM for (a large class of) C&O Σ-protocols. This solves the problem considered and attacked by Chailloux. In more detail, we prove that if the considered C&O Σ-protocol satisfies some very liberal notion of special soundness, then the resulting NIZK is a proof of knowledge with online extractability in the QROM, i.e., when the hash function used for the commitments and the Fiat-Shamir transformation is modeled as a quantum-accessible random oracle. Our security reduction is tight: Whenever a prover outputs a valid proof, the online-extractor succeeds, except with a small probability accounting for collision and preimage attacks on the involved hash functions. For previous reductions, the guaranteed extraction success probability was at least by a factor of q^2 smaller than the succes probability of the prover subjected to extraction (see Table 1). This is our main technical contribution, see Theorem 4.2. Our result also applies to a variant of the Fiat-Shamir transformation where a digital signature scheme (DSS) is constructed. It thereby, for the first time, enables a multiplicatively tight security reduction for, e.g., DSS based on the MPC-in-the-head paradigm [IKOS07], like Picnic [CDG+17], Banquet [BdSGK+21] and Rainier [DKR+21], in the QROM.

A more efficient Unruh transformation. When a Σ-protocol does not have the mentioned C&O structure, a non-interactive proof of knowledge with online extractability in the QROM can be obtained using the Unruh transformation [Unr15]. For technical reasons, the Unruh transformation requires the hash function to be *length preserving*, which may result in large commitments, and thus large NIZKs and digital signature schemes. We revisit this transformation and show, by a rather direct application of our main result above, that the online extractability of the Unruh transform still holds when using a *compressing* hash function. The crucial observation is that the Unruh transformation can be viewed as the composition of a pre-Unruh transformation, which makes use of hash-based commitments and results in a C&O protocol, and the Fiat-Shamir transformation. By applying our security reduction, we obtain the tight online extractability without requiring the hash function to be length preserving.

More efficient NIZKs via Merkle tree based commitments. In real-world constructions based on C&O protocols, like e.g., the Picnic digital signature scheme, commitments and their openings are responsible for a significant fraction of the signature/proof size. For certain parameters, this cost can be reduced by using a collective commitment mechanism based on Merkle trees. This was observed in passing, e.g. in [Fis05], and is exploited in the most recent versions of Picnic. We formalize Merkle-tree-based C&O protocols and extend our main result to NIZKs constructed from them (see Theorem 5.2). Applications of this result include a security reduction of Picnic 3, the newest version of the Picnic digital signature scheme, that is significantly tighter than existing ones: An adversary against the Picnic 3 signature scheme in the QROM with success probability ε can now be used to break the underlying hard problem with probability ε, up to some additive error terms, while previous reductions yielded at most ε^5/q^{10}, where q is the number of random oracle queries. We outline this reduction in Section 5.3.

We compare our reductions in detail to existing techniques in Table 1.

1.2 Technical Overview

Our starting point is the fact that the compressed-oracle technique can be appreciated as a variant of the classical lazy-sampling technique that is applicable in the QROM. Namely, to some extent and informally described here, the compressed-oracle technique gives access to a database that contains the hash values that the adversary $\mathcal{A}$, who has interacted with the random oracle (RO), may know. In particular, up to a small error, for any claimed-to-be hash value y output by $\mathcal{A}$, one can find its preimage x by inspecting the database (and one can safely conclude that $\mathcal{A}$ does not know a preimage of y if there is none in the database). Recalling that a C&O Σ-protocol Π is an interactive proof where the first message consists of hash-based commitments, and exploiting that typically some sort of special soundness property ensures that knowing sufficiently many preimages of these commitments/hashes allows one to efficiently compute a witness, constructing an online extractor for the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$

	2-s⇒PoK	PoK$\overset{\text{FS}}{\Rightarrow}$NIZK-PoK, PoK$\overset{\text{FS}}{\Rightarrow}$UF-NMA DSS	2-s$\overset{\text{FS}}{\Rightarrow}$NIZK-PoK, 2-s$\overset{\text{FS}}{\Rightarrow}$UF-NMA DSS
Unruh rewinding [Unr12] + generic FS [DFMS19]	$O(\varepsilon^3)$	$O(\varepsilon/q^2)$	$O(\varepsilon^3/q^6)$
Σ-protocol OE [DFMS21] + generic FS [DFMS19]	$\varepsilon - g(q,r,n)$	$O(\varepsilon/q^2)$	$O(\varepsilon/q^2) - g(q,r,n)$
this work: NIZK OE	-	-	$\boldsymbol{\varepsilon - h(q,r,n)}$

Table 1. Comparison of the losses of different reductions for the construction of a NIZK proof of knowledge (NIZK-PoK) from a special-sound (Merkle tree based) C&O protocol with constant challenge space size C using r-fold parallel repetition and the Fiat-Shamir transformation. "OE" stands for online extraction, 2-s for special soundness, UF-NMA for plain unforgeability and DSS for digital signature scheme. If the content of a cell in row "security property A ⇒ security property B" is $f(\varepsilon)$, this means that an adversary breaking property B with probability ε yields an adversary breaking property A with probabilty $f(\varepsilon)$. Grey text indicates results that do not apply to Merkle-tree-based C&O protocols like the one used to construct the digital signature schemes Picnic 2 [KZ20] and Picnic 3 [CDG+19b]. The additive error terms are $g(q,r,n) = C^{-r} + O(rq2^{-n/2}) + O(q^3 2^{-n})$ and $h(q,r,n) = O(q^3 2^{-n}) + O(q^2 C^{-r})$, where n is the output length of the random oracles, and q is the number of adversarial (quantum) queries to the random oracle. Finally, we note that the constants hidden by the big-O in $h(q,r,n)$ are reasonable, see Theorems 4.2 and 5.2.

then appears straightforward: The extractor $\mathcal{E}$ simply runs the (possibly dishonest) prover P^*, answering RO queries using the compressed oracle. Once P^* has finished and outputs a proof, $\mathcal{E}$ measures the compressed-oracle database and classically reads off any preimages of the commitments in the proof. Finally, $\mathcal{E}$ run the special soundness extractor that computes a witness from the obtained preimages. It is, however, not obvious that the database contains the preimages of the commitments that are *not* opened in the proof, or that these preimages are correctly formed. Intuitively this should be the case: the RO used for the Fiat-Shamir transformation replaces interaction in that it forces the prover to chose a full set of commitments *before* knowing which ones need to be opened. The crux lies in replacing this intuition by a rigorous proof.

The main insight leading to our proof is that the event that needs to be controlled, namely that *the prover succeeds yet the extractor fails*, can be translated into a property SUC (as in "adversarial SUCcess") of the compressed-oracle database, which needs to be satisfied for the event to hold. It is somewhat of a peculiar property though. The database properties that have led to query complexity lower bounds in prior work, e.g. for (multi-)collision finding [LZ19a, HM21, CFHL21] and similar problems [Zha19, CGLQ20, BLZ21], require the database to contain some particular input-output pairs (e.g. pairs that collide), while the database property SUC additionally *forbids* certain input-output pairs to be contained.

Indeed, the framework from [CFHL21] is almost expressive enough to treat our problem. So, after a mild extension, we can apply it to prove that it is hard for any query algorithm to cause the compressed-oracle database to have property SUC. Analyzing the relevant classical statistical properties of SUC is somewhat tedious but can be done (see the proof of Lemma 5.1). The resulting bound on the probability for the database to satisfy SUC then gives us a bound on the probability of the event that the prover succeeds in producing a valid proof while at the same time fooling the extractor.

Whenever it is advantageous for communication complexity, a Merkle tree can be used to collectively commit to all required messages in a C&O protocol. This collective commitment is one of the optimizations that improve the performance of, e.g. Picnic 2 [KZ20] over Picnic [CDG+17]. As the above-described argument for the extractability of C&O protocols already analyses iterated hashing (the hash-based commitments are hashed to compute the challenge), it generalizes to Merkle-tree-based C&O protocols without too much effort. We present this generalization in Section 5, and obtain similar bounds (see Theorem 5.2).

1.3 Additional Related Work

Besides the already mentioned work above, we note that Chiesa, Manohar and Spooner [CMS19] consider and prove security of various SNARG constructions, while we consider the Fiat-Shamir transformation

of C&O protocols with a form of special soundness. Similar in to [CFHL21], they also provide some tools for deducing security of certain oracle games against quantum attacks by bounding a natural classical variant of the game.

2 Preliminaries

Our main technical proofs reliy on the recently introduced framework by Chung, Fehr, Huang, and Liao [CFHL21] for proving query complexity bounds in the QROM. This framework exploits Zhandry's compressed-oracle technique but abstracts away all the quantum aspects, so that the reasoning becomes purely classical. We give here an introduction to a simplified, and slightly adjusted version that does not consider parallel queries. We start with recalling (a particular view on) the compressed oracle. Along the way, we also give an improved version of Zhandry's central lemma for the compressed oracle.

Before getting into this, we fix the following standard notation. For any positive integer $\ell > 0$, we set $[\ell] := \{1, 2, \ldots, \ell\}$, and we let $2^{[\ell]}$ denote the power set of $[\ell]$, i.e., the set of all subsets of $[\ell]$. We write $\{0,1\}^{\leq \ell}$ for the set of bit strings of size at most ℓ, including the empty string denoted $\emptyset$; similarly for $\{0,1\}^{<\ell}$. Concatenation of two bit strings $v \in \{0,1\}^m$ and $w \in \{0,1\}^n$ is denoted by $v\|w \in \{0,1\}^{m+n}$.

Finally, for any finite non-empty set $\mathcal{Z}$, $\mathbb{C}[\mathcal{Z}]$ denotes the Hilbert space $\mathbb{C}^{|\mathcal{Z}|}$ together with a basis $\{|z\rangle\}$ labeled by the elements $z \in \mathcal{Z}$.

2.1 The Compressed Oracle — Seen as Quantum Lazy Sampling

With the goal to analyze oracle algorithms that interact with a RO $H : \mathcal{X} \to \mathcal{Y}$, consider the set $\mathfrak{D}$ of all functions $D : \mathcal{X} \to \mathcal{Y} \cup \{\bot\}$, where $\bot$ is a special symbol. Such a function is referred to as a *database*. Later, we will fix $\mathcal{X} = \{0,1\}^{\leq B}$ and $\mathcal{Y} = \{0,1\}^n$. For $D \in \mathfrak{D}$, $x \in \mathcal{X}$ and $y \in \mathcal{Y} \cup \{\bot\}$, $D[x \mapsto y]$ denotes the database that maps x to y and otherwise coincides with D, i.e., $D[x \mapsto y](x) = y$ and $D[x \mapsto y](\bar{x}) = D(\bar{x})$ for all $\bar{x} \in \mathcal{X} \setminus \{x\}$.

Following the exposition of [CFHL21], the compressed-oracle technique is a quantum analogue of the classical lazy-sampling technique, commonly used to analyze algorithms in the classical ROM. In the classical lazy-sampling technique, the (simulated) RO starts off with the empty database, i.e., with $D_0 = \bot$, which maps any $x \in \mathcal{X}$ to $\bot$. Then, recursively, upon a query x, the current database D_i is updated to $D_{i+1} := D_i$ if $D_i(x) \neq \bot$, and to $D_{i+1} := D_i[x \mapsto y]$ for a randomly chosen $y \in \mathcal{Y}$ otherwise. This construction ensures that $|\{x \,|\, D_i(x) \neq \bot\}| \leq i$; after i queries thus, using standard sparse-encoding techniques, the database D_i can be efficiently represented and updated.

In the compressed-oracle quantum analogue of this lazy-sampling technique, the (simulated) RO also starts off with the empty database, but now considered as a quantum state $|\bot\rangle$ in the $|\mathfrak{D}|$-dimensional state space $\mathbb{C}[\mathfrak{D}]$, and after i queries the state of the compressed oracle is then supported by databases $|D_i\rangle$ for which $|\{x \,|\, D_i(x) = \bot\}| \leq i$.[2] Here, the update is given by a unitary operator cO acting on $\mathbb{C}[\mathcal{X}] \otimes \mathbb{C}[\mathcal{Y}] \otimes \mathbb{C}[\mathfrak{D}]$, i.e., on the query register, the response register, and the state of the compressed oracle. With respect to the computational basis $\{|x\rangle\}$ of $\mathbb{C}[\mathcal{X}]$ and the Fourier basis $\{|\hat{y}\rangle\}$ of $\mathbb{C}[\mathcal{Y}]$, cO is a *control* unitary, i.e., of the form $\mathsf{cO} = \sum_{x,\hat{y}} |x\rangle\langle x| \otimes |\hat{y}\rangle\langle\hat{y}| \otimes \mathsf{cO}_{x,\hat{y}}$, where $\mathsf{cO}_{x,\hat{y}}$ is a unitary on $\mathbb{C}[\mathcal{Y} \cup \{\bot\}]$, which in the above expression is understood to act on the register that carries the value of the database at the point x. More formally, $\mathsf{cO}_{x,\hat{y}}$ acts on register R_x when identifying $\mathbb{C}[\mathfrak{D}]$ with $\bigotimes_{x \in \mathcal{X}} \mathbb{C}[\mathcal{Y} \cup \{\bot\}]$ by means of the isomorphism $|D\rangle \mapsto \bigotimes_{x \in \mathcal{X}} |D(x)\rangle_{R_x}$. We refer to Lemma 4.3 in the full version of [CFHL21] for the full specification of $\mathsf{cO}_{x,\hat{y}}$; it is not really relevant here.

The compressed oracle is tightly related to the *purified* oracle, which initiates its internal state with a uniform superposition $\sum_h |H\rangle \in \mathbb{C}[\mathfrak{D}]$ of all functions $H : \mathcal{X} \to \mathcal{Y}$, and then answers queries "in superposition". Indeed, at any point in time during the interaction with an oracle quantum algorithm $\mathcal{A}$, the joint state of $\mathcal{A}$ and the compressed oracle coincides with the joint state of $\mathcal{A}$ and the purified oracle after "compressing" the latter.[3] Formally, identifying $\mathbb{C}[\mathfrak{D}]$ with $\bigotimes_{x \in \mathcal{X}} \mathbb{C}[\mathcal{Y} \cup \{\bot\}]$ again, the compression

[2] This means that the density operator that describes the state of the compressed oracle has its support contained in the span of these $|D_i\rangle$.

[3] The terminology is somewhat misleading here; the actual compression takes place when invoking the sparse encoding (see below).

of the state of the purified oracle works by applying the unitary Comp to each register R_x, where

$$\mathsf{Comp} : |y\rangle \mapsto (|y\rangle + \frac{1}{\sqrt{|\mathcal{Y}|}}(|\perp\rangle - |\hat{0}\rangle)$$

for any $y \in \mathcal{Y}$, and $\mathsf{Comp} : |\perp\rangle \mapsto |\hat{0}\rangle$. Here, $|\hat{0}\rangle$ is the $\hat{0}$-vector from the Fourier basis $\{|\hat{y}\rangle\}$ of $\mathbb{C}[\mathcal{Y}]$.

Similarly to the classical case, by exploiting a quantum version of the sparse-encoding technique, both the internal state of the compressed oracle and the evolution cO can be efficiently computed. Furthermore, for any classical function $f : \mathfrak{D} \to \mathcal{T}$ that can be efficiently computed when given the sparse representation of $D \in \mathfrak{D}$, the corresponding quantum measurement given by the projections $P_t = \sum_{D:f(D)=t} |D\rangle\langle D|$ can be efficiently performed when given the sparse representation of the internal state of the compressed oracle. In particular, in Lemma 2.1 below, the condition $\mathbf{y} = D(\mathbf{x})$ for given $\mathbf{x}$ and $\mathbf{y}$ can be efficiently checked by a measurement. See Appendix A in (the full version of) [CFHL21], or Appendix B in [DFMS21] for more details on this technique.

In the classical lazy-sampling technique, if at the end of the execution of an oracle algorithm $\mathcal{A}$, having made q queries to the (lazy-sampled) RO, the database D_q is such that, say, $D_q(x) \neq 0$ for any $x \in \mathcal{X}$, then $\mathcal{A}$'s output is unlikely to be a 0-preimage, i.e., an x that is hashed to 0 upon one more query. $\mathcal{A}$'s best chance is to output an x that he has not queried yet, and thus $D_q(x) = \perp$, and then he has a $1/|\mathcal{Y}|$-chance that $D_{q+1}(x) := D_q[x \mapsto y](x) = 0$, given that y is randomly chosen. Something similar holds in the quantum setting, with some adjustments. The general statement is given by the following result by Zhandry.

Lemma 2.1 (Lemma 5 in [Zha19]). *Let $R \subseteq \mathcal{X}^\ell \times \mathcal{Y}^\ell \times \mathcal{Z}$ be a relation, and let $\mathcal{A}$ be an oracle quantum algorithm that outputs $\mathbf{x} \in \mathcal{X}^\ell$, $\mathbf{y} \in \mathcal{Y}^\ell$ and $z \in \mathcal{Z}$. Furthermore, let*

$$p = p(\mathcal{A}) := \Pr[\mathbf{y} = H(\mathbf{x}) \wedge (\mathbf{x}, \mathbf{y}, z) \in R]$$

be the considered probability when $\mathcal{A}$ has interacted with the standard RO, initialized with a uniformly random function H, and let

$$p' = p'(\mathcal{A}) := \Pr[\mathbf{y} = D(\mathbf{x}) \wedge (\mathbf{x}, \mathbf{y}, z) \in R]$$

be the considered probability when $\mathcal{A}$ has interacted with the compressed oracle instead and D is obtained by measuring its internal state (in the basis $\{|D\rangle\}_{D\in\mathfrak{D}}$). Then

$$\sqrt{p} \leq \sqrt{p'} + \sqrt{\frac{\ell}{|\mathcal{Y}|}}\,.$$

Remark 2.2. This bound is particular useful in case $\mathcal{Z} = \emptyset$ (or R does not depend on its third input z), since then p' is bounded by $\Pr[\exists\, \tilde{\mathbf{x}} : (\tilde{\mathbf{x}}, D(\tilde{\mathbf{x}})) \in R]$ and the latter is determined solely by the evolution of the compressed oracle (when interacting with $\mathcal{A}$) and does not depend on the actual output of $\mathcal{A}$.

In Section 2.3, Corollary 2.8, we will give an alternative such relation between the success probability of an algorithm interacting with the actual RO, and probabilities obtained by inspecting the compressed oracle instead. Strictly speaking, the results of Lemma 2.1 and Corollary 2.8 are incomparable, but in typical applications the latter gives a significantly better bound.

2.2 The Quantum Transition Capacity and Its Relevance

The above discussion shows that, in order to bound the success probability p of an oracle algorithm $\mathcal{A}$, it is sufficient to bound the probability of the database D, obtained by measuring the internal state of the compressed oracle after the interaction with $\mathcal{A}$, satisfying a certain property (e.g., the property of there existing an x such that $D(x) = 0$).

To facilitate that latter, Chung et al. [CFHL21] introduced a framework that, in certain cases, allows to bound this alternative figure of merit by means of purely classical reasoning. We briefly recall here some of the core elements of this framework, which are relevant to us. Note that [CFHL21] considers the parallel-query model, where in each of the q (sequential) interactions with the RO, an oracle algorithm

$\mathcal{A}$ can make k queries simultaneously in parallel with each interaction. Here, we consider the (more) standard model of one query per interaction, i.e., setting $k = 1$. On the other hand, we state and prove a slight generalization of Theorem 5.16 in [CFHL21] (when restricted to $k = 1$).

A subset $\mathsf{P} \subseteq \mathfrak{D}$ is called a *database property*. We say that $D \in \mathfrak{D}$ *satisfies* P if $D \in \mathsf{P}$, and the complement of P is denoted $\neg\mathsf{P} = \mathfrak{D} \setminus \mathsf{P}$. For such a database property P, [CFHL21] defines $\llbracket \bot \overset{q}{\Longrightarrow} \mathsf{P} \rrbracket$ as the square-root of the maximal probability of D satisfying P when D is obtained by measuring the internal state of the compressed oracle after the interaction with $\mathcal{A}$, maximized over all oracle quantum algorithms $\mathcal{A}$ with query complexity q, i.e., in short

$$\llbracket \bot \overset{q}{\Longrightarrow} \mathsf{P} \rrbracket := \max_{\mathcal{A}} \sqrt{\Pr[D \in \mathsf{P}]}\,. \tag{1}$$

In the context of Lemma 2.1 for the case $\mathcal{Z} = \emptyset$ (see Remark 2.2), we can define the database property $\mathsf{P}^R := \{D \in \mathfrak{D} \mid \exists\, \mathbf{x} \in \mathcal{X}^\ell \colon (\mathbf{x}, D(\mathbf{x})) \in R\}$ induced by R, and thus bound

$$p'(\mathcal{A}) \le \Pr[(\mathbf{x}, D(\mathbf{x})) \in R] \le \Pr[D \in \mathsf{P}^R] \le \llbracket \bot \overset{q}{\Longrightarrow} \mathsf{P}^R \rrbracket^2 \tag{2}$$

for any oracle quantum algorithm $\mathcal{A}$ with query complexity q.

Furthermore, Lemma 5.6 in [CFHL21] shows that for any target database property P and for any sequence $\mathsf{P}_0, \mathsf{P}_1, \ldots, \mathsf{P}_q$ with $\neg\mathsf{P}_0 = \{\bot\}$ and $\mathsf{P}_q = \mathsf{P}$,

$$\llbracket \bot \overset{q}{\Longrightarrow} \mathsf{P} \rrbracket \le \sum_{s=1}^{q} \llbracket \neg\mathsf{P}_{s-1} \to \mathsf{P}_s \rrbracket\,, \tag{3}$$

where, for any database properties P and P', the definition of the *quantum transition capacity* $\llbracket \mathsf{P} \to \mathsf{P}' \rrbracket$ is recalled in Definition 2.3.

The nice aspect of the framework of [CFHL21] is that it provides means to manipulate and bound quantum transition capacities using purely classical reasoning, i.e., without the need to understand and work with the definition. Indeed, for instance Theorem 2.4 below, which is a variant of Theorem 5.17 in (the full version of) [CFHL21], shows how to bound $\llbracket \mathsf{P} \to \mathsf{P}' \rrbracket$ by means of a certain classical probability; furthermore, to facilitate the application of such theorems, [CFHL21] showed that the quantum transition capacity satisfies several natural manipulation rules, like $\llbracket \mathsf{P} \to \mathsf{P}' \rrbracket = \llbracket \mathsf{P}' \to \mathsf{P} \rrbracket$ (i.e., it is symmetric), and

$$\begin{gathered} \llbracket \mathsf{P} \cap \mathsf{Q} \to \mathsf{P}' \rrbracket \le \min\{ \llbracket \mathsf{P} \to \mathsf{P}' \rrbracket, \llbracket \mathsf{Q} \to \mathsf{P}' \rrbracket \} \qquad \text{and} \\ \min\{ \llbracket \mathsf{P} \to \mathsf{P}' \rrbracket, \llbracket \mathsf{P} \to \mathsf{Q}' \rrbracket \} \le \llbracket \mathsf{P} \to \mathsf{P}' \cup \mathsf{Q}' \rrbracket \le \llbracket \mathsf{P} \to \mathsf{P}' \rrbracket + \llbracket \mathsf{P} \to \mathsf{Q}' \rrbracket\,, \end{gathered} \tag{4}$$

which allow to decompose complicated capacities into simpler ones. Therefore, by means of the above series of inequalities with p from Lemma 2.1 on the left hand side, it is possible (in certain cases) to bound the success probability of any oracle quantum algorithm $\mathcal{A}$ in the QROM by means of the following recipe: (1) Choose suitable transitions $\mathsf{P}_{s-1} \to \mathsf{P}_s$, (2) decompose the capacities $\llbracket \neg\mathsf{P}_{s-1} \to \mathsf{P}_s \rrbracket$ into simpler ones using manipulation rules as above, and (3) bound the simplified capacities by certain classical probabilities, exploiting results like Theorem 2.4. We will closely follow this recipe.

In order to state and later use Theorem 2.4, we need to introduce the following additional concepts. As explained above, there is no need to actually spell out the definition of the quantum transition capacity in order to use Theorem 2.4; for completeness, and since it is needed for the proof of Theorem 2.4, we do provide it below.

For any database $D \in \mathfrak{D}$ and any $x \in \mathcal{X}$,

$$D|^x := \{D[x \mapsto y] \mid y \in \mathcal{Y} \cup \{\bot\}\}$$

denotes the set of all databases that coincide with D outside of x. Furthermore, for a database property P,

$$\mathsf{P}|_{D|^x} := \{y \in \mathcal{Y} \cup \{\bot\} \mid D[x \mapsto y] \in \mathsf{P}\} \subseteq \mathcal{Y} \cup \{\bot\}$$

denotes the set of values y for which $D[x \mapsto y]$ satisfies P. Following the convention used in [CFHL21], we identify the subset $\mathsf{P}|_{D|^x} \subseteq \mathcal{Y} \cup \{\bot\}$ with the projector $\mathsf{P}|_{D|^x} = \sum_y |y\rangle\langle y|$ acting on $\mathbb{C}[\mathcal{Y} \cup \{\bot\}]$, where the sum is over all $y \in \mathsf{P}|_{D|^x}$.

Definition 2.3 (Def. 5.5 of [CFHL21], case $k = 1$). *Let P, P' be two database properties. Then, the* quantum transition capacity *(of order 1) is defined as*

$$[\![\mathsf{P} \to \mathsf{P}']\!] := \max_{\mathbf{x},\hat{\mathbf{y}},D} \|\mathsf{P}'|_{D|^{\mathbf{x}}}\, \mathsf{cO}_{\mathbf{x},\hat{\mathbf{y}}}\, \mathsf{P}|_{D|^{\mathbf{x}}}\|$$

where the max is over all $\mathbf{x} \in \mathcal{X}^k$, $\hat{\mathbf{y}} \in \hat{\mathcal{Y}}^k$, and $D \in \mathfrak{D}$.

The following is a variation of Theorem 5.17 in (the full version of) [CFHL21], obtained by restricting k to 1. On the other hand, we exploit and include some symmetry that is not explicit in the original statement. The proof is a small adjustment to the original proof.

Theorem 2.4. *Let P and P' be database properties with trivial intersection, i.e., $\mathsf{P} \cap \mathsf{P}' = \emptyset$, and for every $D \in \mathfrak{D}$ and $x \in \mathcal{X}$ let*

$$\mathsf{L}^{x,D} := \begin{cases} \mathsf{P}|_{D|^x} & \text{if } \perp \in \mathsf{P}'|_{D|^x} \\ \mathsf{P}'|_{D|^x} & \text{if } \perp \in \mathsf{P}|_{D|^x} \end{cases},$$

with $\mathsf{L}^{x,D}$ being either of the two if $\perp \notin \mathsf{P}|_{D|^x} \cup \mathsf{P}'|_{D|^x}$.[4] *Then*

$$[\![\mathsf{P} \to \mathsf{P}']\!] \le \max_{x,D} \sqrt{10P\big[U \in \mathsf{L}^{x,D}\big]}\,,$$

where U is uniform over $\mathcal{Y}$, and the maximization can be restricted to $D \in \mathfrak{D}$ and $x \in \mathcal{X}$ for which both $\mathsf{P}|_{D|^x}$ and $\mathsf{P}'|_{D|^x}$ are non-empty.

Remark 2.5. Both, $\mathsf{P}|_{D|^x}$ and $\mathsf{P}'|_{D|^x}$, and thus also $\mathsf{L}^{x,D}$, do not depend on the value of $D(x)$, only on the values of D outside of x.

Proof. For any $D \in \mathfrak{D}$ and $x \in \mathcal{X}$, we observe that

$$\|\mathsf{P}'|_{D|^x}\, \mathsf{cO}_{x,\hat{y}}\, \mathsf{P}|_{D|^x}\| = \|\mathsf{P}|_{D|^x}\, \mathsf{cO}_{x,-\hat{y}}\, \mathsf{P}'|_{D|^x}\|\,,$$

and so it is sufficient to argue for the case when $\mathsf{L}^{x,D}$ is set to $\mathsf{P}'|_{D|^x}$. By the disjointness requirement, as subsets of $\mathcal{Y} \cup \{\perp\}$, the complement of $\mathsf{L}^{x,D} = \mathsf{P}'|_{D|^x}$ is a superset of $\mathsf{P}|_{D|^x}$. Thus, as projections acting on $\mathbb{C}[\mathcal{Y} \cup \{\perp\}]$, $\mathsf{P}|_{D|^x} \le \mathbb{I} - \mathsf{L}^{x,D}$. Therefore, the above norm is upper bounded by $\|\mathsf{L}^{x,D}\, \mathsf{cO}_{x,y}\, (\mathbb{I} - \mathsf{L}^{x,D})\|$. Given that $\perp \notin \mathsf{L}^{x,D}$, the square norm $\|\mathsf{L}^{x,D}\, \mathsf{cO}_{x,\hat{y}}\, (\mathbb{I} - \mathsf{L}^{x,D})\|^2$ can be upper bounded exactly as in the proof of Theorem 5.17 in [CFHL21] by $10P\big[U \in \mathsf{L}^{x,D}\big]$, giving the claimed bound. □

2.3 An Improved Variant of Zhandry's Lemma

We show here an alternative to Zhandry's lemma (Lemma 2.1), which offers a better bound in typical applications. To start with, note that Lemma 2.1 considers an algorithm $\mathcal{A}$ that not only outputs $\mathbf{x} = (x_1, \ldots, x_\ell)$ but also $\mathbf{y} = (y_1, \ldots, y_\ell)$, where the latter is supposed to be the point-wise hash of $\mathbf{x}$; indeed, this is what is being checked in the definition of the probability p, along with $(\mathbf{x}, \mathbf{y}, z) \in R$. This requirement is somewhat unnatural, in that an algorithm $\mathcal{A}$ for, say, finding a collision, i.e., $x_1 \neq x_2$ with $H(x_1) = H(x_2)$, does *not* necessarily output the (supposed to be equal) hashes $y_1 = H(x_1)$ and $y_2 = H(x_2)$. Of course, this is no problem since one can easily transform such an algorithm $\mathcal{A}$ that does not output the hashes into one that does, simply by making a few more (classical) queries to the RO at the end of the execution, and then one can apply Lemma 2.1 to this tweaked algorithm $\tilde{\mathcal{A}}$.

We show below that if we anyway consider this tweaked algorithm $\tilde{\mathcal{A}}$, which is *promised* to query the RO to obtain and then output the hashes of $\mathbf{x} = (x_1, \ldots, x_\ell)$, then we can actually improve the bound and avoid the square-roots in Lemma 2.1. On top, the proof is much simpler than Zhandry's proof for his lemma.

At the core is the following lemma; Corollary 2.8 below then puts it in a form that is comparable to Lemma 2.1 and shows the improvement.

[4] By the disjointness requirement, $\perp$ cannot be contained in both.

Lemma 2.6. *Let $\mathcal{A}$ be an oracle quantum algorithm that outputs $\mathbf{x} = (x_1, ..., x_\ell) \in \mathcal{X}^\ell$ and $z \in \mathcal{Z}$. Let $\tilde{\mathcal{A}}$ be the oracle quantum algorithm that runs $\mathcal{A}$, makes ℓ classical queries on the outputs x_i to obtain $\mathbf{y} = H(\mathbf{x})$, and then outputs $(\mathbf{x}, \mathbf{y}, z)$. When $\tilde{\mathcal{A}}$ interacts with the compressed oracle instead, and at the end D is obtained by measuring the internal state of the compressed oracle, then, conditioned on $\tilde{\mathcal{A}}$'s output $(\mathbf{x}, \mathbf{y}, z)$,*

$$\Pr[\mathbf{y} = D(\mathbf{x}) | (\mathbf{x}, \mathbf{y}, z)] \geq 1 - \frac{2\ell}{|\mathcal{Y}|}\,.$$

Proof. Consider first $\tilde{\mathcal{A}}$ interacting with the *purified* (yet uncompressed) oracle. Conditioned on $\tilde{\mathcal{A}}$'s output $(\mathbf{x}, \mathbf{y}, z)$, the state of the oracle is then supported by $|H\rangle$ with $H(x_i) = y_i$ for all $i \in \{1, \ldots, \ell\}$, i.e., the registers labeled by $x_1, ..., x_\ell$ are in state $|y_1\rangle \cdots |y_\ell\rangle$. Given that the compressed oracle is obtained by applying Comp to all the registers, we thus have that

$$\begin{aligned}\Pr[y_i = y_i' | (\mathbf{x}, \mathbf{y}, z)] &= \left|\langle y_i | \mathsf{Comp} | y_i \rangle\right|^2 = \left|\langle y_i | \Big(|y_i\rangle + \tfrac{1}{\sqrt{|\mathcal{Y}|}} (|\perp\rangle - |\hat{0}\rangle) \Big)\right|^2 \\ &= \left|1 - \tfrac{1}{\sqrt{|\mathcal{Y}|}} \langle y_i | \hat{0} \rangle\right|^2 = \left|1 - \tfrac{1}{|\mathcal{Y}|}\right|^2 \geq 1 - \frac{2}{|\mathcal{Y}|}\,.\end{aligned}$$

Applying union bound concludes the claim. □

The following generalization of Lemma 2.6 follows immediately by enhancing $\mathcal{A}$ so that it computes and outputs all the values x that need to be queried in order to compute $\mathcal{F}^H(z)$, and then apply Lemma 2.6 above.

Corollary 2.7. *Let $\mathcal{A}$ be an oracle quantum algorithm that produces an arbitrary output $z \in \mathcal{Z}$, and let $\mathcal{F}$ be an arbitrary* classical *ℓ-query oracle algorithm. Let $\tilde{\mathcal{A}} := \mathcal{F} \circ \mathcal{A}$ be the oracle quantum algorithm that first runs $\mathcal{A}$ to obtain z, then $\mathcal{F}$ to obtain $y := \mathcal{F}^H(z)$, and finally outputs (y, z). When $\tilde{\mathcal{A}}$ interacts with the compressed oracle instead, and at the end D is obtained by measuring the internal state of the compressed oracle, then, conditioned on $\tilde{\mathcal{A}}$'s output (y, z),*

$$\Pr[y = \mathcal{F}^D(z) | (y, z)] \geq 1 - \frac{2\ell}{|\mathcal{Y}|}.$$

The following corollary of Lemma 2.6 is put in a form that can be nicely compared with Lemma 2.1, understanding that typically Lemma 2.1 is applied to $\tilde{\mathcal{A}}$.

Corollary 2.8. *Let $R \subseteq \mathcal{X}^\ell \times \mathcal{Y}^\ell \times \mathcal{Z}$ be a relation. Let $\mathcal{A}$ be an oracle quantum algorithm that outputs $\mathbf{x} \in \mathcal{X}^\ell$ and $z \in \mathcal{Z}$, and let $\tilde{\mathcal{A}}$ be as in Lemma 2.6. Let*

$$p_\circ(\mathcal{A}) := \Pr[(\mathbf{x}, H(\mathbf{x}), z) \in R]$$

be the considered probability when $\mathcal{A}$ has interacted with the RO. Furthermore, let $p(\tilde{\mathcal{A}})$, $p'(\tilde{\mathcal{A}})$ and $p''(\tilde{\mathcal{A}})$ be defined as in Lemma 2.1 (but now for $\tilde{\mathcal{A}}$). Then

$$p_\circ(\mathcal{A}) = p(\tilde{\mathcal{A}}) \leq p'(\tilde{\mathcal{A}}) + \frac{2\ell}{|\mathcal{Y}|}\,.$$

For convenience, we recall that

$$p'(\tilde{\mathcal{A}}) = \Pr[\mathbf{y} = D(\mathbf{x}) \wedge (\mathbf{x}, \mathbf{y}, z) \in R] \leq \Pr[(\mathbf{x}, D(\mathbf{x}), z) \in R]\,.$$

Proof. The equality holds by construction of $\tilde{\mathcal{A}}$. For the first inequality, we observe that

$$\begin{aligned}p'(\tilde{\mathcal{A}}) &= \Pr[\mathbf{y} = D(\mathbf{x}) | (\mathbf{x}, \mathbf{y}, z) \in R] \Pr[(\mathbf{x}, \mathbf{y}, z) \in R] \\ &\geq \left(1 - \tfrac{2\ell}{|\mathcal{Y}|}\right) \Pr[(\mathbf{x}, \mathbf{y}, z) \in R] \geq \left(1 - \tfrac{2\ell}{|\mathcal{Y}|}\right) p(\tilde{\mathcal{A}}) \geq p(\tilde{\mathcal{A}}) - \tfrac{2\ell}{|\mathcal{Y}|}\,,\end{aligned}$$

where the first inequality is by Lemma 2.6. The second and last inequality in the statement holds trivially by definition of p'. □

3 Some Background on (Non-)Interactive Proofs

Throughout this and later sections, we consider a hash function $H : \mathcal{X} \to \mathcal{Y}$, to be modeled as a RO then. For concreteness and simplicity, we assume that all relevant variables are encoded as bit strings, and that we can therefore choose $H : \{0,1\}^{\leq B} \to \{0,1\}^n$ for sufficiently large B and n.[5]

Let $\{\mathcal{I}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\mathcal{W}_\lambda\}_{\lambda\in\mathbb{N}}$ be two families of sets, with the members being labeled by the security parameter $\lambda \in \mathbb{N}$. Let $R_\lambda \subseteq \mathcal{I}_\lambda \times \mathcal{W}_\lambda$ be a relation that is polynomial-time computable in λ. $w \in \mathcal{W}_\lambda$ is called a *witness* for $\mathsf{inst} \in \mathcal{I}_\lambda$ if $R_\lambda(\mathsf{inst}, w)$, and $L_\lambda := \{\mathsf{inst} \in \mathcal{I}_\lambda \mid \exists\, w \in \mathcal{W}_\lambda : R_\lambda(\mathsf{inst}, w)\}$.

Below, we recall some concepts in the context of interactive and non-interactive proofs for such families $\{R_\lambda\}_{\lambda\in\mathbb{N}}$ of relations. We start by discussing the aspired security definition for non-interactive proofs.

3.1 Non-interactive Proofs and Online Extractability

An *non-interactive proof in the random-oracle model* for a family $\{R_\lambda\}_{\lambda\in\mathbb{N}}$ of relations consists of a pair $(\mathcal{P}, \mathcal{V})$ of oracle algorithms, referred to as *prover* and *verifier*, both making queries to the RO $H : \mathcal{X} \to \mathcal{Y}$. The prover $\mathcal{P}$ takes as input $\lambda \in \mathbb{N}$ and an instance $\mathsf{inst} \in L_\lambda$ and outputs a *proof* $\pi \in \Pi_\lambda$, and $\mathcal{V}$ takes as input $\lambda \in \mathbb{N}$ and a pair $(\mathsf{inst}, \pi) \in \mathcal{I}_\lambda \times \Pi_\lambda$ and outputs a Boolean value, 0 or 1, or `accept` or `reject`. The verifier $\mathcal{V}$ is required to run in time polynomial in λ, while, *per-se*, $\mathcal{P}$ may have unbounded running time.[6]

By default, we require correctness and soundness, i.e., that for any $\lambda \in \mathbb{N}$ and any $\mathsf{inst} \in L_\lambda$

$$\Pr\big[\mathcal{V}^H(\lambda, \mathsf{inst}, \pi) : \pi \leftarrow \mathcal{P}^H(\lambda, \mathsf{inst})\big] \geq 1 - \varepsilon_{\mathrm{cor}}(\lambda),$$

while for any $\lambda \in \mathbb{N}$ and any oracle quantum algorithm $\mathcal{P}^*$ (a *dishonest prover*) with query complexity q

$$\Pr\big[\mathsf{inst} \notin L_\lambda \wedge \mathcal{V}^H(\lambda, \mathsf{inst}, \pi) : (\mathsf{inst}, \pi) \leftarrow \mathcal{P}^{*H}(\lambda)\big] \leq \varepsilon_{\mathrm{snd}}(\lambda, q, n)$$

for certain $\varepsilon_{\mathrm{cor}}$ and $\varepsilon_{\mathrm{snd}}$, respectively referred to as *correctness error* and *soundness error*. The fact that the instance inst, for which $\mathcal{P}^*$ tries to forge a proof, is not given as input to $\mathcal{P}^*$ but is instead chosen by $\mathcal{P}^*$ is referred to as $\mathcal{P}^*$ being *adaptive*.

We now move towards defining *online extractability* (for adaptive $\mathcal{P}^*$). For that purpose, let $\mathcal{P}^*$ be a dishonest prover as above, except that it potentially outputs some additional auxiliary (possibly quantum) output Z next to (inst, π). We then consider an interactive algorithm $\mathcal{E}$, called *online extractor*, which takes $\lambda \in \mathbb{N}$ as input and simulates the answers to the oracle queries in the execution of $\mathcal{V}^H \circ \mathcal{P}^{*H}(\lambda)$, which we define to run $(\mathsf{inst}, \pi, Z) \leftarrow \mathcal{P}^{*H}(\lambda)$ followed by $v \leftarrow \mathcal{V}^H(\lambda, \mathsf{inst}, \pi)$; furthermore, at the end, $\mathcal{E}$ outputs $w \in \mathcal{W}_\lambda$. We denote the execution of $\mathcal{V}^H \circ \mathcal{P}^{*H}(\lambda)$ with the calls to H simulated by $\mathcal{E}$, and considering $\mathcal{E}$'s final output w as well, as $(\mathsf{inst}, \pi, Z; v; w) \leftarrow \mathcal{V}^{\mathcal{E}} \circ \mathcal{P}^{*\mathcal{E}}(\lambda)$.

Definition 3.1. *A non-interactive proof in the (quantum-accessible) RO model (QROM) for $\{R_\lambda\}_{\lambda\in\mathbb{N}}$ is a* proof of knowledge with online extractability *(PoK-OE) against adaptive adversaries if there exists an online extractor $\mathcal{E}$, and functions $\varepsilon_{\mathrm{sim}}$ (the* simulation error*) and $\varepsilon_{\mathrm{ex}}$ (the* extraction error*), with the following properties. For any $\lambda \in \mathbb{N}$ and for any dishonest prover $\mathcal{P}^*$ with query complexity q,*

$$\delta\big([(\mathsf{inst}, \pi, Z, v)]_{\mathcal{V}^H \circ \mathcal{P}^{*H}(\lambda)}, [(\mathsf{inst}, \pi, Z, v)]_{\mathcal{V}^{\mathcal{E}} \circ \mathcal{P}^{*\mathcal{E}}(\lambda)}\big) \leq \varepsilon_{\mathrm{sim}}(\lambda, q, n)$$

and

$$\Pr\big[v = \mathtt{accept} \wedge (\mathsf{inst}, w) \notin R : (\mathsf{inst}, \pi, Z; v; w) \leftarrow \mathcal{V}^{\mathcal{E}} \circ \mathcal{P}^{*\mathcal{E}}(\lambda)\big] \leq \varepsilon_{\mathrm{ex}}(\lambda, q, n)\,.$$

Furthermore, the runtime of $\mathcal{E}$ is polynomial in $\lambda + q + n$, and $\varepsilon_{\mathrm{sim}}(\lambda, q, n)$ and $\varepsilon_{\mathrm{ex}}(\lambda, q, n)$ are negligible in λ whenever q and n are polynomial in λ.

Remark 3.2. In the classical definition of a proof of knowledge, the extractor $\mathcal{E}$ interacts with $\mathcal{P}^*$ only, and the verifier $\mathcal{V}$ is not explicitly involved, but would typically be run by $\mathcal{E}$. Here, in the context of online extractability, it is necessary to explicitly go through the verification procedure, which also makes oracle queries, to determine whether a proof is valid, i.e., for the event $v = \mathtt{accept}$ to be well defined.

[5] B and n may depend on the security parameter $\lambda \in \mathbb{N}$. We will then assume that B and n can be computed from λ in polynomial time (in λ).

[6] Alternatively, one may consider a witness w for inst to be given as additional input to $\mathcal{P}$, and then ask $\mathcal{P}$ to be polynomial-time as well.

3.2 (Commit-and-Open) Σ-Protocols

A *Σ-protocol* is a 3-round public-coin interactive proof $(\mathcal{P}, \mathcal{V})$ for a relation $R_\lambda \subseteq \mathcal{I}_\lambda \times \mathcal{W}_\lambda$, indexed by the security parameter. From now on, we leave any dependencies on the security parameter implicit. We therefore simply write R etc. By definition, a Σ-protocol has the following communication pattern. In the first round, $\mathcal{P}$ sends a *first message* a; in the second round, $\mathcal{V}$ sends a random *challenge* $c \in \mathcal{C}$; and in the third round, $\mathcal{P}$ sends a *response* z (see Fig. 1). By a slight abuse of notation, we sometimes write $\mathcal{V}(\mathsf{inst}, a, c, z)$ for the predicate that determines whether $\mathcal{V}$ accepts the transcript (a, c, z) on input inst.

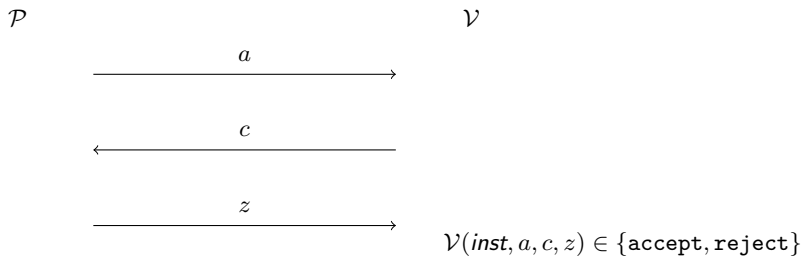

Fig. 1. A plain Σ-protocol, formally introduced in Section 3.2.

For the purpose of this work, a *commit-and-open Σ-protocol*, or *C&O Σ-protocol* or *C&O protocol* for short, is a Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ of a special form, involving a hash function H that is modeled as a RO.[7] Concretely, in a C&O protocol, the transcript (a, c, z) is of the following form (see Fig. 2). The first message a consists of *commitments* $y_1, \ldots, y_\ell$, computed as $y_i = H(m_i)$ for *messages* $m_1, \ldots, m_\ell \in \mathcal{M}$, and possibly an additional string $a_\circ$[8]. The challenge c is picked uniformly at random from the challenge space $\mathcal{C} \subseteq 2^{[\ell]}$, which is set to be a subset of $2^{[\ell]}$. Finally, the response z is given by $\mathbf{m}_c = (m_i)_{i \in c}$. Eventually, $\mathcal{V}$ accepts if and only if $H(m_i) = y_i$ for all $i \in c$ and some given predicate $V(\mathsf{inst}, c, \mathbf{m}_c, a_\circ)$ is satisfied.

For the above to be meaningful, we obviously need that $\mathcal{M} \subseteq \mathcal{X}$, i.e., the bit size of the possible m_i's are upper bounded by B. Furthermore, the parameter n determines the hardness of finding a collision in H (in the random oracle model), and thus the level of binding the commitments provide.

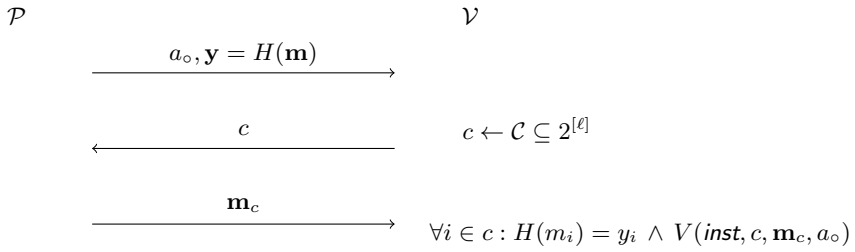

Fig. 2. An (ordinary) C&O Σ-protocol, formally introduced in Section 3.2.

Remark 3.3. Looking ahead, we may also consider a generalization of the above notion of a C&O protocol, where the first message is parsed as a *single* commitment y of the ℓ messages $m_1, \ldots, m_\ell$ and where this commitment is computed by means of an arbitrary "multi-message" commitment scheme involving H, which has the property that any subset of $m_1, \ldots, m_\ell$ can be opened without revealing the remaining m_i's. The above component-wise hashing is then one particular instantiation, but alternatively one can

[7] One could also refer to Σ-protocols that use non-hash-based commitments, and/or are analyzed in the standard model, as *C&O protocols*, but this is not the scope here.

[8] Note that $m_i \in \mathcal{M}$ may consist of the actual "message" (computed by the prover using the witness w), possibly concatenated with randomness.

for instance also compute y by means of a Merkle tree (see Section 5.1), and then open individual m_i's by revealing the corresponding authentication paths. We stress that the concepts discussed below: the notions of $\mathfrak{S}$-soundness and $\mathfrak{S}$-soundness* and the probability $p^{\mathfrak{S}}_{triv}$, do not depend on the choice of commitment scheme, and thus remain unaffected when considering such a *Merkle-tree-based C&O* protocol. To emphasize the default choice of the commitment scheme, which is element-wise hashing, we sometimes also speak of an *ordinary* C&O protocol.

3.3 $\mathfrak{S}$-soundness of C&O Σ-Protocols

We briefly recall the notion of $\mathfrak{S}$-soundness and $\mathfrak{S}$-soundness* for C&O protocols, as considered in [DFMS21], which offers a convenient general notion of special soundness, or more generally k-soundness for C&O protocols. A similar notion of $\mathfrak{S}$-soundness naturally exists for plain Σ-protocols, i.e., Σ-protocols in the plain model. For completeness, we formalize the latter in Appendix A.

Here and below, given a C&O protocol Π with challenge space $\mathcal{C} \subseteq 2^{[\ell]}$, we let $\mathfrak{S} \subseteq 2^{\mathcal{C}}$ be an arbitrary non-empty, monotone increasing set of subsets $S \subseteq \mathcal{C}$, where the monotonicity means that $S \in \mathfrak{S} \wedge S \subseteq S' \Rightarrow S' \in \mathfrak{S}$. We then also set $\mathfrak{S}_{\min} := \{S \in \mathfrak{S} \mid S_\circ \subsetneq S \Rightarrow S_\circ \notin \mathfrak{S}\}$ to be the minimal sets in $\mathfrak{S}$.

For simplicity, the reader can consider $\mathfrak{S} = \mathfrak{T}_k := \{S \subseteq \mathcal{C} \mid |S| \geq k\}$ for some threshold k, and thus $\mathfrak{S}_{\min} = \{S \subseteq \mathcal{C} \mid |S| = k\}$. This then corresponds to the notion of k-soundness for C&O protocols, which in turn means that the witness can be computed from valid responses to k (or more) distinct challenges for a given first message $y_1, \ldots, y_\ell$, assuming the messages $m_1, \ldots, m_\ell$ to be uniquely determined by their commitments.

Definition 3.4 ([DFMS21] Def. 5.1). *A C&O protocol Π is* $\mathfrak{S}$-sound *if there exists an efficient deterministic algorithm $\mathcal{E}_{\mathfrak{S}}(\mathsf{inst}, m_1, \ldots, m_\ell, a_\circ, S)$ that takes as input an instance $\mathsf{inst} \in \mathcal{I}$, messages $m_1, \ldots, m_\ell \in \mathcal{M} \cup \{\bot\}$, a string $a_\circ$, and a set $S \in \mathfrak{S}_{\min}$, and outputs a witness for inst if $V(\mathsf{inst}, c, \mathbf{m}_c, a_\circ)$ for all $c \in S$.*[9]

We note that Wikström [Wik18] also considers a general notion of special soundness (but then for multi-round protocols); however, the notion in [Wik18] is more restrictive in that it requires some matroid structure on top. For instance, the r-fold parallel repetition of a k-sound protocol does not fit into the formalism by Wikström.

A slightly stronger condition than $\mathfrak{S}$-*soundness* is the following variant, which differs in that the extractor needs to work as soon as there *exists* a set S as specified, without the extractor being given S as input. We refer to [DFMS21] for a more detailed discussion of this aspect. As explained there, whether S is given or not often makes no (big) difference.

For instance, when $\mathfrak{S}_{\min}$ consists of a polynomial number of sets S then the extractor can do a brute-force search to find S, and so $\mathfrak{S}$-soundness* is then implied by $\mathfrak{S}$-soundness. Also, the r-fold parallel repetition of a $\mathfrak{S}$-sound protocol, which by default is a $\mathfrak{S}^{\vee r}$-sound protocol (see [DFMS21]), is automatically $\mathfrak{S}^{\vee}$-sound* if $\mathfrak{S}_{\min}$ is polynomial in size: the extractor can then do a brute-force search in every repeated instance.

Definition 3.5 ([DFMS21] Def. 5.2). *A C&O protocol Π is* $\mathfrak{S}$-sound* *if there exists an efficient deterministic algorithm $\mathcal{E}^*_{\mathfrak{S}}(\mathsf{inst}, m_1, \ldots, m_\ell, a_\circ)$ that takes as input an instance $\mathsf{inst} \in \mathcal{I}$ and strings $m_1, \ldots, m_\ell \in \mathcal{M} \cup \{\bot\}$ and $a_\circ$, and it outputs a witness for inst if there exists $S \in \mathfrak{S}$ such that $V(\mathsf{inst}, c, \mathbf{m}_c, a_\circ)$ for all $c \in S$.*

As in [DFMS21], we define

$$p^{\mathfrak{S}}_{triv} := \frac{1}{|\mathcal{C}|} \max_{\hat{S} \notin \mathfrak{S}} |\hat{S}|\,, \tag{5}$$

capturing the "trivial" attack of picking a set $\hat{S} = \{\hat{c}_1, \ldots, \hat{c}_m\} \notin \mathfrak{S}$ of challenges $\hat{c}_i \in \mathcal{C}$ and then prepare $\hat{\mathbf{m}} = (\hat{m}_1, \ldots, \hat{m}_\ell)$ and $a_\circ$ in such a way that $V(\mathsf{inst}, c, \hat{\mathbf{m}}_c, a_\circ)$ holds if $c \in \hat{S}$. After committing to $\hat{m}_1, \ldots, \hat{m}_\ell$, the prover can successfully answer to challenges $c \in \hat{S}$.

[9] The restriction for S to be in $\mathfrak{S}_{\min}$, rather than in $\mathfrak{S}$, is to avoid an exponentially sized input while asking $\mathcal{E}_{\mathfrak{S}}$ to be efficient.

3.4 The Fiat-Shamir Transformation of (C&O) Σ-Protocols

The Fiat-Shamir (FS) transformation [FS87] turns arbitrary Σ-protocols into non-interactive proofs in the random oracle model by setting the challenge $c \in \mathcal{C}$ to be the hash of the instance and the first message a. For this transformation to work smoothly, it is typically assumed that $|\mathcal{C}|$ is a power of 2 and its elements are represented as bit strings of size $\log|\mathcal{C}|$, so that one can indeed set c to be (the first $\log|\mathcal{C}|$ bits of) the hash $H(\mathsf{inst}, a)$. The assumption on $|\mathcal{C}|$ is essentially without loss of generality (WLOG), since one can always reduce the size of $|\mathcal{C}|$ to the next lower power of 2, at the cost of losing at most 1 bit of security. However, for a C&O Σ-protocol, where a challenge space $\mathcal{C}$ is a (typically strict) subset of $2^{[\ell]}$, there is not necessarily a natural way to represent $c \in \mathcal{C}$ as a bitstring of size $\log|\mathcal{C}|$. Therefore, we will make it explicit that the challenge-set $c \in \mathcal{C} \subset 2^{[\ell]}$ is computed from the "raw randomness" $H(\mathsf{inst}, y_1, \ldots, y_\ell, a_\circ)$ in a deterministic way as $c = \gamma \circ H(\mathsf{inst}, y_1, \ldots, y_\ell, a_\circ)$ for an appropriate function $\gamma : \mathcal{Y} \to \mathcal{C}$, mapping a uniformly random hash in $\mathcal{Y}$ to a random challenge-set in $\mathcal{C}$. Obviously, for $H(\mathsf{inst}, y_1, \ldots, y_\ell, a_\circ)$ to be defined, in addition to $\mathcal{M} \subseteq \mathcal{X}$ we also need that $\mathcal{I} \times \mathcal{Y}^\ell \subseteq \mathcal{X}$, which again just means that B needs to be large enough. We write $\mathsf{FS}[\Pi]$ for the Fiat-Shamir transformation of a (C&O) Σ-protocol Π.

Remark 3.6. Additionally, we need that n is sufficiently large, so that there is a sufficient amount of randomness in the hash value $H(\mathsf{inst}, y_1, \ldots, y_\ell)$ in order to be mapped to a random $c \in \mathcal{C}$. The canonical choice for γ is then the function that the *interactive* verifier applies to his local randomness to compute the random challenge $c \in \mathcal{C}$. To simplify the exposition, we assume that n is indeed sufficiently large. Otherwise, one can simply set $\mathcal{Y} := \{0,1\}^{n'}$ instead, for sufficiently large n', and then let y_i be $H(m_i)$ *truncated* to the original number n of bits again. This truncation has no effect on our results.

Remark 3.7. We assume WLOG that the two kinds of inputs to H, i.e., m_i and $(\mathsf{inst}, y_1, \ldots, y_\ell, a_\circ)$, are differently formatted, e.g., bit strings of different respective sizes or prefixes (this is referred to as *domain separation*). In other words, we assume that $\mathcal{M}$ and $\mathcal{I} \times \mathcal{Y}^\ell$ are disjoint.

Remark 3.8. When considering the adaptive security of a Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of a C&O protocol Π for a relation R, the additional string $a_\circ$, which may be part of the first message a of the original protocol Π, may WLOG be considered to be part of the instance inst instead.

Indeed, any dishonest prover $\mathcal{P}^*$ against $\mathsf{FS}[\Pi]$, which (by Definition 3.1) outputs an instance inst and a proof $\pi = (a_\circ, y_1, \ldots y_\ell)$, can alternatively be parsed as a dishonest prover that outputs an instance $\mathsf{inst}' = (\mathsf{inst}, a_\circ)$ and a proof $\pi' = (y_1, \ldots y_\ell)$. Thus, $\mathcal{P}^*$ can be parsed as a dishonest prover against $\mathsf{FS}[\Pi']$, where the C&O protocol Π' works as Π, except that $a_\circ$ is considered as part of the instance, rather than as part of the first message, and thus Π' is a C&O protocol for the relation $((\mathsf{inst}, a_\circ), w) \in R' :\Leftrightarrow (\mathsf{inst}, w) \in R$.[10] Therefore, security (in the sense of Definition 3.1) for $\mathsf{FS}[\Pi']$ implies that of $\mathsf{FS}[\Pi]$.

4 Online Extractability of the FS-Transformation: The Case of Ordinary C&O Protocols

We now consider the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of an ordinary C&O protocol Π. Our goal is to show that $\mathsf{FS}[\Pi]$ admits online extraction. We note that by exploiting Remark 3.8, we may assume WLOG that the first message of Π consists of the commitments $y_1, \ldots, y_\ell$ only, and no additional string $a_\circ$. In Section 5, we then consider the case of Merkle-tree-based C&O protocols.

Our analysis of the online extractability of $\mathsf{FS}[\Pi]$ uses the framework of Chung et al. [CFHL21], discussed and outlined in Section 2. Thus, at the core of our analysis is a bound on a certain quantum transition capacity. This is treated in the upcoming subsection.

4.1 Technical Preface

We first introduce a couple of elementary database properties (related to CoLlisions and the SiZe of the database) that will be useful for us:

$$\mathsf{CL} := \{D \,|\, \exists\, x \neq x' : D(x) = D(x') \neq \bot\} \quad \text{and} \quad \mathsf{SZ}_{\leq s} := \{D \,|\, \#\{z | D(z) \neq \bot\} \leq s\}.$$

[10] We do not specify the local computation of the honest prover $\mathcal{P}'$ in $\Pi' = (\mathcal{P}', \mathcal{V}')$, i.e., how to act when $a_\circ$ is part of the input, and in general it might not be efficient, but this is fine since we are interested in the security against dishonest provers.

Next, for an instance $\mathit{inst} \in \mathcal{I}$, we want to specify the database property that captures a cheating prover that succeeds in producing an accepting proof while fooling the extractor. For the purpose of specifying this database property, we introduce the following notation. For a given database $D \in \mathfrak{D}$ and for a commitment $y \in \mathcal{Y}$, we define $D^{-1}(y)$ to be the smallest $x \in \mathcal{X}$ with $D(x) = y$, with the convention that $D^{-1}(y) := \bot$ if there is no such x, as well as $D^{-1}(\bot) := \bot$. We note that by removing collisions, we ensure that there is at most one such x; thus, taking the smallest one in case of multiple choices is not important but only for well-definedness.

The database property of interest can now be defined as

$$\mathsf{SUC} := \left\{ D \;\middle|\; \begin{array}{c} \exists\, \mathbf{y} \in \mathcal{Y}^\ell \text{ and } \mathit{inst} \in \mathcal{I} \text{ so that } \mathbf{m} := D^{-1}(\mathbf{y}) \text{ satisfies} \\ V(\mathit{inst}, c, \mathbf{m}_c) \text{ for } c := \gamma \circ D(\mathit{inst}, \mathbf{y}) \text{ and } \big(\mathit{inst}, \mathcal{E}^*(\mathit{inst}, \mathbf{m})\big) \notin R \end{array} \right\}. \tag{6}$$

Informally, assuming no collisions (i.e., restricting to $D \notin \mathsf{CL}$), the database property SUC captures whether a database D admits a *valid* proof $\pi = (\mathbf{y}, \mathbf{m}_c)$ for an instance inst for which the (canonical) extractor, which first computes $\mathbf{m}$ by inverting D and then runs $\mathcal{E}^*$, *fails* to produce a witness.

Our (first) goal is to show that $\left[\!\left[\bot \overset{q}{\Longrightarrow} \mathsf{SUC} \cup \mathsf{CL} \right]\!\right]$ is small, capturing that it is unlikely that after q queries the compressed database contains collisions or admits a valid proof upon which the extractor fails. Indeed, we show the following, where $p^{\mathfrak{S}}_{triv}$ is the trivial cheating probability of Π as defined in (5).

Lemma 4.1. $\left[\!\left[\bot \overset{q}{\Longrightarrow} \mathsf{SUC} \cup \mathsf{CL} \right]\!\right] \leq 2eq^{3/2}2^{-n/2} + q\sqrt{10 \max\left(q\ell \cdot 2^{-n}, p^{\mathfrak{S}}_{triv}\right)}$.

The formal proof is given below; we first give some informal outline here. In a first step, by using (3) and union-bound-like properties of the transition capacity, and additionally exploiting a bound from [CFHL21] to control the transition capacity of CL, we reduce the problem to bounding the quantum transition capacity $\left[\!\left[\mathsf{SZ}_{\leq s} \backslash \mathsf{SUC} \to \mathsf{SUC} \right]\!\right]$ for $s < q$. Informally, this capacity is a measure of the "likelihood" — but then in a *quantum*-sense — that a database $D \in \mathfrak{D}$ that is bounded in size and not in SUC turns into a database D' that *is* in SUC, when D is updated to $D' = D[x \mapsto U]$ with U uniformly random in $\mathcal{Y}$.

We emphasize that in the considered quantum setting, the state of the compressed oracle at any point is a *superposition* of databases, and a query is made up of a *superposition* of inputs; nevertheless, due to Theorem 2.4, the above classical intuition is actually very close to what needs to be shown to rigorously bound the considered quantum transition capacity. Formally, as will become clear in the proof below, we need to show that for any database $D \in \mathsf{SZ}_{\leq s} \backslash \mathsf{SUC}$ and for any $x \in \mathcal{X}$ with $D(x) = \bot$, the probability that $D[x \mapsto U] \in \mathsf{SUC}$ is small. Below, this probability is bounded in the *Case 2* and *Case 3* parts of the proof, where the two cases distinguish between x being a "commit query" or a "challenge query".

Informally, for D with $D(x) = \bot$, if x is a "commit query" then assigning a value to $D(x)$ can only make a difference, i.e., turn $D \notin \mathsf{SUC}$ into $D[x \mapsto u] \in \mathsf{SUC}$, if u is a coordinate of some $\mathbf{y} \in \mathcal{Y}^\ell$ for which $D(\mathit{inst}, \mathbf{y}) \neq \bot$ for some inst. Indeed, otherwise, $D[x \mapsto u]$ does not contribute to a valid proof π that did not exist before. Thus, given the bound $s < q$ on the size of D, this happens with probability at most $q\ell/2^n$ for a random u. Similarly, if x is a "challenge query", i.e. of the form $x = (\mathit{inst}, \mathbf{y})$, then assigning a value u to $D(x)$ can only make a difference if $V(\mathit{inst}, c, \mathbf{m}_c)$ is satisfied for $c = \gamma(u)$ and $\mathbf{m} = D^{-1}(\mathbf{y})$, while $\mathcal{E}^*(\mathit{inst}, \mathbf{m})$ is not a witness for inst. However, for a random u, this is bounded by $p^{\mathfrak{S}}_{triv}$.

But then, on top of the above, due to the quantum nature of the quantum transition capacity,[11] Theorem 2.4 requires to also show the "reverse", i.e., that for any $D \in \mathsf{SUC}$ and for any $x \in \mathcal{X}$ with $D(x) \neq \bot$, the probability that $D[x \mapsto U] \in \mathsf{SZ}_{\leq s} \backslash \mathsf{SUC}$ is small; this is analyzed in *Case 1* below.

Thus, by exploiting the framework of [CFHL21], the core of the reasoning is purely classical, very closely mimicking how one would have to reason the classical setting with a classical RO. Due to the rather complex definition of SUC, the formal argument in each case is still somewhat cumbersome.

[11] At the core, this is related to the reversibility of quantum computing and the resulting ability to "uncompute" a query.

Proof. We first observe that, by (3) (which is Lemma 5.6 in [CFHL21]) and basic properties of the quantum transition capacity as in (4),

$$\begin{aligned}\left[\!\left[\bot \xRightarrow{q} \mathsf{SUC}\cup\mathsf{CL}\right]\!\right] &\leq \sum_{s=0}^{q-1}\left[\!\left[\mathsf{SZ}_{\leq s}\backslash\mathsf{SUC}\backslash\mathsf{CL}\to\mathsf{SUC}\cup\mathsf{CL}\cup\neg\mathsf{SZ}_{\leq s+1}\right]\!\right]\\ &\leq \sum_{s=0}^{q-1}\Big(\left[\!\left[\mathsf{SZ}_{\leq s}\backslash\mathsf{SUC}\backslash\mathsf{CL}\to\neg\mathsf{SZ}_{\leq s+1}\right]\!\right]+\left[\!\left[\mathsf{SZ}_{\leq s}\backslash\mathsf{SUC}\backslash\mathsf{CL}\to\mathsf{CL}\right]\!\right]\\ &\qquad +\left[\!\left[\mathsf{SZ}_{\leq s}\backslash\mathsf{SUC}\backslash\mathsf{CL}\to\mathsf{SUC}\right]\!\right]\Big)\\ &\leq \sum_{s=0}^{q-1}\Big(\left[\!\left[\mathsf{SZ}_{\leq s}\to\neg\mathsf{SZ}_{\leq s+1}\right]\!\right]+\left[\!\left[\mathsf{SZ}_{\leq s}\backslash\mathsf{CL}\to\mathsf{CL}\right]\!\right]+\left[\!\left[\mathsf{SZ}_{\leq s}\backslash\mathsf{SUC}\to\mathsf{SUC}\right]\!\right]\Big)\,. \end{aligned} \tag{7}$$

The first term, $\left[\!\left[\mathsf{SZ}_{\leq s}\to\neg\mathsf{SZ}_{\leq s+1}\right]\!\right]$, vanishes, while the second term was shown to be bounded as

$$\left[\!\left[\mathsf{SZ}_{\leq s}\backslash\mathsf{CL}\to\mathsf{CL}\right]\!\right]\leq 2e\sqrt{(s+1)/|\mathcal{Y}|}\leq 2e\sqrt{q/2^n} \tag{8}$$

in Example 5.28 in [CFHL21]. Thus, it remains to control the third term, which we will do by means of Theorem 2.4 with $\mathsf{P}:=\mathsf{SZ}_{\leq s}\setminus\mathsf{SUC}$ and $\mathsf{P}':=\mathsf{SUC}$.

To this end, we consider arbitrary but fixed $D\in\mathfrak{D}$ and input $x\in\mathcal{X}$. By Remark 2.5, we may assume that $D(x)=\bot$. Furthermore, for $\mathsf{P}|_{D|^x}$ to be non-empty, it must be that $D\in\mathsf{SZ}_{\leq s}$, i.e., D is bounded in size. We now distinguish between the following cases for the considered D and x.

Case 1: $D\in\mathsf{SUC}$. In particular, $\bot\in\mathsf{SUC}|_{D|^x}=\mathsf{P}'_{D|^x}$. So, Theorem 2.4 instructs us to set $\mathsf{L}:=\mathsf{P}_{D|^x}$, where we leave the dependency of L on D and x implicit to simplify notation. Given that $D\in\mathsf{SUC}$, we can consider inst and $\mathbf{y}$ as promised by the definition of SUC in (6), i.e., such that $V(\mathit{inst},c,\mathbf{m}_c)$ and $\big(\mathit{inst},\mathcal{E}^*(\mathit{inst},\mathbf{m})\big)\notin R$ for

$$c:=\gamma\circ D(\mathit{inst},\mathbf{y})\quad\text{and}\quad m_i:=D^{-1}(y_i)\,,$$

where it is understood that $\mathbf{m}=(m_1,\ldots,m_\ell)$. Recall that $D(x)=\bot$; thus, by definition of the m_i's, it must be that $x\neq m_i$ for all i, and the fact that $V(\mathit{inst},c,\mathbf{m}_c)$ is satisfied for c as defined implies that $x\neq(\mathit{inst},\mathbf{y})$. Furthermore,

$$u\in\mathsf{L}\iff D[x\mapsto u]\in\mathsf{P}\implies D[x\mapsto u]\notin\mathsf{SUC}\implies u\in\{y_1,\ldots,y_\ell\}\,,$$

where the last implication is easiest seen by contraposition: Assume that $u\notin\{y_1,\ldots,y_\ell\}$. Then, also recalling that $x\neq m_i$, we have that $m_i=D^{-1}(y_i)=D[x\mapsto u]^{-1}(y_i)$. But also $c=\gamma\circ D(\mathit{inst},\mathbf{y})=\gamma\circ D[x\mapsto u](\mathit{inst},\mathbf{y})$. Together, this implies that the defining property of SUC is also satisfied for $D[x\mapsto u]$, i.e., $D[x\mapsto u]\in\mathsf{SUC}$, as was to be shown. Thus, we can bound

$$P[U\in\mathsf{L}]\leq P[U\in\{y_1,\ldots,y_\ell\}]\leq\frac{\ell}{|\mathcal{Y}|}\,. \tag{9}$$

Case 2: $D\notin\mathsf{SUC}$, and x is a "commit query", i.e., $x=m\in\mathcal{M}$. In particular, $\bot\notin\mathsf{P}'|_{D|^x}$ (by the assumption that $D(x)=\bot$) and so in light of Theorem 2.4 we may choose $\mathsf{L}:=\mathsf{P}'|_{D|^x}$. We then have

$$u\in\mathsf{L}\iff D[x\mapsto u]\in\mathsf{P}'=\mathsf{SUC}\implies\exists\,\mathit{inst},\mathbf{y},i:D(\mathit{inst},\mathbf{y})\neq\bot\wedge u=y_i\,. \tag{10}$$

This final implication can be seen as follows. By definition of SUC, the assumption $D[x\mapsto u]\in\mathsf{SUC}$ implies the existence of inst and $\mathbf{y}=(y_1,\ldots,y_\ell)$ with $V(\mathit{inst},c,\mathbf{m}_c)$ and $\big(\mathit{inst},\mathcal{E}^*(\mathit{inst},\mathbf{m})\big)\notin R$ for

$$c:=\gamma\circ D[x\mapsto u](\mathit{inst},\mathbf{y})=\gamma\circ D(\mathit{inst},\mathbf{y})\quad\text{and}\quad m_i:=D[x\mapsto u]^{-1}(y_i)\,,$$

where the equality in the definition of c exploits that x is not a "challenge" query. With the goal to reach a contradiction, assume that $u\neq y_i$ for all i. This assumption implies that $D[x\mapsto u](x)=u\neq y_i$. But also $D(x)=\bot\neq y_i$, and hence for all $\xi\in\mathcal{X}$ and $i\in\{1,\ldots,\ell\}$: $D(\xi)=y_i\Leftrightarrow D[x\mapsto u](\xi)=y_i$. Therefore, $m_i=D[x\mapsto u]^{-1}(y_i)=D^{-1}(y_i)$ for all i, and the above then implies that $D\in\mathsf{SUC}$, a contradiction.

Thus, there exists i for which $u = y_i$; furthermore, $D(\mathsf{inst}, \mathbf{y}) \neq \bot$ given that $V(\mathsf{inst}, u, \mathbf{m}_c)$ is satisfied for $c = \gamma \circ D(\mathsf{inst}, \mathbf{y})$. This shows the claimed implication.

Thus, we can bound

$$P[U \in \mathsf{L}] \leq P[\exists\, \mathsf{inst}, \mathbf{y}, i : D(\mathsf{inst}, \mathbf{y}) \neq \bot \wedge u = y_i] \leq \frac{s\ell}{|\mathcal{Y}|} \leq \frac{q\ell}{|\mathcal{Y}|}\,. \tag{11}$$

Case 3: $D \notin \mathsf{SUC}$, and x is a "challenge query", i.e., $x = (\mathsf{inst}, \mathbf{y}) \in \mathcal{I} \times \mathcal{Y}^\ell$. Set $\mathbf{m} = (m_1, \ldots, m_\ell)$ for $m_i := D^{-1}(y_i)$. Again, we have that $\bot \notin \mathsf{SUC}|_{D|^x} = \mathsf{P}'_{D|^x}$, and so by Theorem 2.4 we may set $\mathsf{L} := \mathsf{P}'_{D|^x}$. Here, we can argue that

$$\begin{aligned} u \in \mathsf{L} &\iff D[x \mapsto u] \in \mathsf{P}' = \mathsf{SUC} \\ &\implies V(\mathsf{inst}, u, \mathbf{m}_{\gamma(u)}) \text{ and } \big(\mathsf{inst}, \mathcal{E}^*(\mathsf{inst}, \mathbf{m})\big) \notin R\,, \end{aligned}$$

where the final implication can be seen as follows. By definition of SUC, the assumption $D[x \mapsto u] \in \mathsf{SUC}$ implies the existence of inst' and $\mathbf{y}' = (y'_1, \ldots, y'_\ell)$ with $V(\mathsf{inst}', u, \mathbf{m}'_c)$ and $\mathcal{E}^*(\mathsf{inst}', \mathbf{m}') \neq w$ for

$$c := \gamma \circ D[x \mapsto u](\mathsf{inst}', \mathbf{y}') \quad \text{and} \quad m'_i := D[x \mapsto u]^{-1}(y'_i) = D^{-1}(y'_i)\,,$$

where the very last equality exploits that x is not a "commit" query. With the goal to come to a contradiction, assume that $(\mathsf{inst}', \mathbf{y}') \neq (\mathsf{inst}, \mathbf{y}) = x$. Then, $c = \gamma \circ D[x \mapsto u](\mathsf{inst}', \mathbf{y}') = \gamma \circ D(\mathsf{inst}', \mathbf{y}')$, and the above then implies that $D \in \mathsf{SUC}$, a contradiction. Thus, $(\mathsf{inst}', \mathbf{y}') = (\mathsf{inst}, \mathbf{y}) = x$. In particular, $\mathbf{m}' = \mathbf{m}$ and $c = \gamma \circ D[x \mapsto u](\mathsf{inst}', \mathbf{y}') = \gamma \circ D[x \mapsto u](x) = \gamma(u)$. Hence, the claimed implication holds.

Thus, we can bound

$$\begin{aligned} P[U \in \mathsf{L}] &\leq P[V(\mathsf{inst}, \gamma(U), \mathbf{m}_{\gamma(U)}) \wedge \mathcal{E}^*(\mathsf{inst}, \mathbf{m}) \neq w] \\ &\leq P[V(\mathsf{inst}, \gamma(U), \mathbf{m}_{\gamma(U)}) \wedge S := \{c \,|\, V(\mathsf{inst}, c, \mathbf{m}_c)\} \notin \mathfrak{S}] \\ &\leq P[\gamma(U) \in S := \{c \,|\, V(\mathsf{inst}, c, \mathbf{m}_c)\} \notin \mathfrak{S}] \\ &\leq \max_{S \notin \mathfrak{S}} P[\gamma(U) \in S] \\ &\leq p^{\mathfrak{S}}_{triv}\,. \end{aligned} \tag{12}$$

By Theorem 2.4, we now get

$$\begin{aligned} [\![\mathsf{SZ}_{\leq s} \backslash \mathsf{SUC} \backslash \mathsf{CL} \to \mathsf{SUC}]\!] &\leq \max_{x,D} \sqrt{10 P\big[U \in \mathsf{L}^{x,D}\big]} \\ &\leq \sqrt{10}\sqrt{\max\left(\frac{\ell}{|\mathcal{Y}|}, \frac{q\ell}{|\mathcal{Y}|}, p^{\mathfrak{S}}_{triv}\right)} \\ &\leq \sqrt{10}\sqrt{\max\left(q\ell \cdot 2^{-n}, p^{\mathfrak{S}}_{triv}\right)}, \end{aligned}$$

where we have used Equations (9), (11) and (12) in the second inequality. Combining with Equations (8) and (7) yields the desired bound. □

4.2 Online Extractability of the Fiat-Shamir Transformation

We are now ready to state and proof the claimed online-extractability result for the Fiat-Shamir transformation of (ordinary) C&O protocols.

Theorem 4.2. *Let Π be a $\mathfrak{S}$-sound* ordinary C&O protocol with challenge space $\mathcal{C}_\lambda$ and $\ell = \ell(\lambda)$ commitments, and set $\kappa = \kappa(\lambda) := \max_{c \in \mathcal{C}_\lambda} |c|$. Then, $\mathsf{FS}[\Pi]$ is a proof of knowledge with online extractability in the QROM (as in Definition 3.1), with $\varepsilon_{\mathrm{sim}}(\lambda, q, n) = 0$ and*

$$\begin{aligned} \varepsilon_{\mathrm{ex}}(\lambda, q, n) &\leq 2(\kappa + 1) \cdot 2^{-n} + \left(2eq^{3/2} 2^{-n/2} + q\sqrt{10 \max\left(q\ell \cdot 2^{-n}, p^{\mathfrak{S}}_{triv}\right)}\right)^2 \\ &\leq (22\ell + 60) q^3 2^{-n} + 20 q^2 p^{\mathfrak{S}}_{triv}\,. \end{aligned}$$

The runtime of the extractor is dominated by running the compressed oracle, which has complexity $O(q^2) \cdot poly(n, B)$, and running $\mathcal{E}^*$.

We note that the above bound on ε_{ex} is asymptotically tight, except for the factor ℓ. Indeed, the binding property of the hash-based commitment can be invalidated by means of a collision finding attack, which succeeds with probability $\Omega(q^3/2^n)$. Furthermore the trivial soundness attack, which potentially applies to a $\mathfrak{S}$-sound* C&O protocol Π, can be complemented with a Grover search, yielding an attack against $\mathsf{FS}[\Pi]$ that succeeds with probability $\Omega(q^2 p^{\mathfrak{S}}_{triv})$. The non-tightness by a factor of ℓ is very mild in most cases. In particular, the number of commitments ℓ is polynomial in λ and thus in n. For the most common case of a parallel repetition of a protocol with a constant number of commitments, using a hash function with output length linear in λ (e.g. $n = 3\lambda$) results in $\ell = O(n) = O(\lambda)$.

Proof. We consider an arbitrary but fixed $\lambda \in \mathbb{N}$. For simplicity, we assume that $|c|$ is the same for all $c \in \mathcal{C}_\lambda$, and thus equal to $\kappa = \kappa(\lambda)$. If it is not, we could always make the prover output a couple of dummy outputs m_i to match the upper bound on $|c|$. Let $\mathcal{P}^*$ be a dishonest prover that, after making q queries to a RO H, outputs $(\mathsf{inst}, \pi) = (\mathsf{inst}, \mathbf{y}, \mathbf{m}_\circ)$ plus some (possibly quantum) auxiliary output Z. In the experiment $\mathcal{V}^{\mathcal{E}} \circ \mathcal{P}^{*\mathcal{E}}(\lambda)$, our extractor $\mathcal{E}$ works as follows while simulating all queries to H (by $\mathcal{P}^*$ and $\mathcal{V}$) with the compressed oracle:

1. Run $\mathcal{P}^*(\lambda)$ to obtain (inst, π, Z) where $\pi = (\mathbf{y}, \mathbf{m}_\circ)$ with $\mathbf{m}_\circ = (m_1, \ldots, m_\kappa)$.
2. Run $\mathcal{V}(\lambda, \mathsf{inst}, \pi)$ to obtain v. In detail: obtain $h_0 := H(\mathsf{inst}, \mathbf{y})$ and $h_j := H(m_j)$ for $j \in \{1, \ldots, \kappa\}$, and set $v := \mathtt{accept}$ if and only if the pair consisting of $\mathbf{x} = \big((\mathsf{inst}, \mathbf{y}), m_1, \ldots, m_\kappa\big)$ and $\mathbf{h} = (h_0, h_1, \ldots, h_\kappa)$ satisfies the relation $\tilde{R}$, defined to hold if and only if
$$(h_1, \ldots, h_\kappa) = \mathbf{y}_c \quad \wedge \quad V(\mathsf{inst}, c, \mathbf{m}_\circ) \quad \text{where} \quad c := \gamma(h_0)\,.$$
3. Measure the internal state of the compressed oracle to obtain D.
4. Run $\mathcal{E}^*(\mathsf{inst}, \mathbf{m})$ on input inst and $\mathbf{m} := D^{-1}(\mathbf{y})$ to obtain w.

Note that in the views of both $\mathcal{P}^*$ and $\mathcal{V}$, the interaction with H and the interaction with $\mathcal{E}$ differ only in that their oracle queries are answered by a compressed oracle instead of a real random-oracle in the latter case. This simulation is perfect and therefore $\varepsilon_{\text{sim}}(\lambda, q, n) = 0$.

Considering $\mathcal{P}^*$ as the algorithm $\mathcal{A}$ in Lemma 2.6, the additional classical oracle queries that $\mathcal{V}$ performs in $\mathcal{V} \circ \mathcal{P}^*$ then match up with the algorithm $\tilde{\mathcal{A}}$, with $h_0, \ldots, h_\kappa$ here playing the role of $y_1, \ldots, y_\ell$ in Lemma 2.6. Thus,
$$\Pr\big[\mathbf{h} \neq D(\mathbf{x})\big] \leq 2(\kappa(\lambda) + 1) \cdot 2^{-n}\,.$$

Therefore, we can bound the figure of merit ε_{ex} as
$$\begin{aligned}
&\varepsilon_{\text{ex}}(\lambda, q, n) = \Pr\big[v = \mathtt{accept} \wedge (\mathsf{inst}, w) \notin R\big] = \Pr\big[(\mathbf{x}, \mathbf{h}) \in \tilde{R} \wedge (\mathsf{inst}, w) \notin R\big] \\
&\leq \Pr\big[\big(\mathbf{x}, D(\mathbf{x})\big) \in \tilde{R} \wedge (\mathsf{inst}, w) \notin R\big] + 2(\kappa(\lambda) + 1) \cdot 2^{-n} \\
&\leq \Pr\big[\big(\mathbf{x}, D(\mathbf{x})\big) \in \tilde{R} \wedge (\mathsf{inst}, w) \notin R \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}\big] + \Pr[D \in \mathsf{SUC} \cup \mathsf{CL}] + 2(\kappa(\lambda) + 1) \cdot 2^{-n}
\end{aligned}$$

Using the definition of $\tilde{R}$, understanding that $c := \gamma \circ D(\mathsf{inst}, \mathbf{y})$, we can write the first term as
$$\begin{aligned}
&\Pr\big[D(\mathbf{m}_\circ) = \mathbf{y}_c \wedge V(\lambda, \mathsf{inst}, c, \mathbf{m}_\circ) \wedge (\mathsf{inst}, w) \notin R \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}\big] \\
&\quad \leq \Pr\big[V(\lambda, \mathsf{inst}, c, \mathbf{m}_c) \text{ for } \mathbf{m} := D^{-1}(\mathbf{y}) \wedge (\mathsf{inst}, w) \notin R \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}\big] \\
&\quad \leq \Pr\big[D \in \mathsf{SUC} \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}\big] \\
&\quad = 0\,,
\end{aligned}$$

where the first equality exploits that $D(m) = y$ iff $m = D^{-1}(y)$ for $D \notin \mathsf{CL}$.

We may thus conclude that
$$\begin{aligned}
\varepsilon_{\text{ex}}(\lambda, q, n) &\leq (2\kappa(\lambda) + 1) \cdot 2^{-n} + \Pr\big[D \in \mathsf{SUC} \cup \mathsf{CL}\big] \\
&\leq (2\kappa(\lambda) + 1) \cdot 2^{-n} + \big[\!\big[\bot \overset{q}{\Longrightarrow} \mathsf{SUC} \cup \mathsf{CL}\big]\!\big]^2\,,
\end{aligned}$$

where the last inequality is by definition (1) of $[\![\bot \overset{q}{\Longrightarrow} \cdot]\!]$. The claimed bound now follows from Lemma 4.1. □

4.3 The Unruh-Transformation with a Compressing Hash Function

We conclude this section by showing an improvement to the *Unruh transformation* [Unr15], which follows directly from our result above. At the core of the Unruh transformation is a generic technique to transform any Σ-protocol into a C&O protocol. In [Unr15], this transformation is presented in combination with parallel repetition and the Fiat-Shamir transformation as a means to construct (online-extractable) NIZK proofs of knowledge in the QROM. The entire transformation was later dubbed the Unruh transformation.

In fact, the Unruh transformation was the first NIZK proof of knowledge in the QROM; the QROM security of the Fiat-Shamir transformation was only established several years later [DFMS19, LZ19a]. Despite being significantly less efficient than the Fiat-Shamir transformation, the Unruh transformation is still useful in certain cases because it puts weaker requirements on the underlying Σ-protocol.

Here, to allow for a modular analysis, we consider the first step of the Unruh transformation, i.e., the transformation from a Σ-protocol into a C&O protocol, as an individual transformation, which we refer to as the *pre-Unruh transformation*, formally defined below. We stress that we allow the RO H to be *compressing*, i.e. $|\mathcal{Y}| < |\mathcal{X}|$, while the extraction technique of [Unr15] required H to be a *length-preserving* RO. This obviously has a significant positive impact on the efficiency of the Unruh transformation.

Let $\Sigma = (\mathcal{P}_\circ, \mathcal{V}_\circ)$ be a Σ-protocol. We write $a_\circ \leftarrow \mathcal{P}_\circ$ to denote the first message in $\Pi_\circ$ as produced by $\mathcal{P}_\circ$ (for a given instance inst). Furthermore, we write $z(a_\circ, c)$ for $\mathcal{P}_\circ$'s response then upon receiving challenge $c \in \mathcal{C}$.[12]

Definition 4.3 (Pre-Unruh transformation). *Let $\Sigma = (\mathcal{P}_\circ, \mathcal{V}_\circ)$ be a Σ-protocol as above. Then, the pre-Unruh-transformation $\mathsf{pU}[\Sigma] = (\mathcal{P}, \mathcal{V})$ of $\Pi_\circ$ is the C&O protocol with first message*

$$a := (a_\circ, (y_i)_{i \in \mathcal{C}})$$

where $a_\circ \leftarrow \mathcal{P}_\circ$ and for each $i \in \mathcal{C}$, $y_i := H(z_i)$ for $z_i := z(a_\circ, i)$), and with response $z := z_c$ upon challenge $c \in \mathcal{C}$. To verify, $\mathcal{V}$ runs $\mathcal{V}_\circ$ on $(a_\circ, c, z)$ and checks if $H(z) = y_c$; if both are true, it accepts, otherwise it rejects.

Clearly, $\mathsf{pU}[\Sigma]$ is only efficient if Σ has at most polynomially many possible challenges (which can always be obtained by restricting the challenge space). As mentioned, the resulting C&O protocol can then be repeated in parallel and made non-interactive using the Fiat-Shamir transformation. We will now provide a fairly straightforward corollary to conclude the security of the more efficient variant of the (full) Unruh transformation that allows for a compressing RO, given by the composition of the pre-Unruh transformation introduced above, parallel repetition and the Fiat-Shamir transformation. In the following, denote the r-fold parallel repetition of a (C&O) Σ-protocol Π by Π^r and use the notation $\mathsf{Unr}_r[\Sigma] := \mathsf{FS}\,[\mathsf{pU}[\Sigma]^r]$ for the Unruh transformation with r-fold parallel repetition.

Remark 4.4. A proof in $\Pi = \mathsf{Unr}_r[\Sigma]$ can be generated in time $T_{\mathcal{P}}^{\Pi} = rT_{\mathcal{P}}^{\Sigma} + (\ell_0 r + 1)T_H$, and verified in time $T_{\mathcal{V}}^{\Pi} = rT_{\mathcal{V}}^{\Sigma} + (1+r)T_H$, where $T_{\mathcal{P}}^{\Sigma}, T_{\mathcal{V}}^{\Sigma}$ and T_H are the prover and verifier runtime of Σ, and the time required for computing one hash, respectively.

It is straightforward to verify that the pre-Unruh transformation does not harm most security properties of the Σ-protocol. In particular, it tightly preserves soundness and honest-verifier zero-knowledge (in the QROM). It also preserves $\mathfrak{S}$-soundness in a certain sense.

Proposition 4.5. *Let Σ be an $\mathfrak{S}$-sound Σ-protocol with challenge space size $\ell = \ell(\lambda)$ with extractor runtime T. Then $\Pi := \mathsf{pU}[\Sigma]$ is $\mathfrak{S}$-sound as a C&O protocol with extractor runtime $T' \leq T + O(\ell)$. Furthermore, suppose that membership in $\mathfrak{S}$ is checkable in time $T_{\mathfrak{S}}$. Then Π is $\mathfrak{S}$-sound* with extractor runtime $T'' \leq T' + \ell^2 T_{\mathfrak{S}} + \ell T_{\mathcal{V}}$, where $T_{\mathcal{V}}$ is the runtime of Π's verification predicate $\mathcal{V}$.*

Proof. Let $\mathcal{E}_\Sigma$ be the extractor for Σ guaranteed to exist by Definition A.1. Note that for $\Pi = \mathsf{pU}[\Sigma]$ regarded as a C&O protocol, for each challenge exactly one of the commitments has to be opened. For such protocols, we use c and $\{c\}$ interchangeably (where c is a challenge in Π). We define an extractor $\mathcal{E}_\Pi$ as follows. On input $(\mathsf{inst}, m_1, ..., m_\ell, a_\circ, S)$, run $w = \mathcal{E}_\Sigma(\mathsf{inst}, a_\circ, S, \{m_c\}_{c \in S})$, then output w. The only runtime overhead of $\mathcal{E}_\Pi$ results from having to parse its input and preparing the input for $\mathcal{E}_\Sigma$.

[12] We note that $z(a_\circ, c)$ may be a *randomized* function of $a_\circ$ and c. Furthermore, $z(a_\circ, c)$ is typically computed by $\mathcal{P}_\circ$ by means of the *randomness used to produce* $a_\circ$.

We continue to define an $\mathfrak{S}$-soundness* extractor $\mathcal{E}^*_{\Pi}$ for Π as follows. On input $(\mathsf{inst}, m_1, ..., m_\ell, a_\circ)$, compute $b_c = \mathcal{V}(\mathsf{inst}, a_\circ, c, m_c)$ for all $c \in \mathcal{C}$, and set $\hat{S} = \{c \in \mathcal{C} \,|\, b_c = 1\}$. Using at most $\ell(\ell+1)/2$ membership tests for $\mathfrak{S}$, find $S \subseteq \hat{S}$ such that $S \in \mathfrak{S}_{\min}$. Finally, run $w = \mathcal{E}_{\Pi}(\mathsf{inst}, m_1, ..., m_\ell, a_\circ, S)$ and output w. The runtime statement is straightforward. □

Using Proposition 4.5 above and Lemma 5.3 from [DFMS21] to argue $\mathfrak{S}^{\vee r}$-soundness* of the parallel repetition of $\mathsf{pU}[\Pi]$, and using Theorem 4.2 to argue online extractability of its Fiat-Shamir transformation, we obtain the online-extractability of the Unruh transformation with computationally binding commitments, i.e., when using a *compressing* hash function for the commitments.

Corollary 4.6. *Let Σ be an $\mathfrak{S}$-sound Σ-protocol with challenge space size ℓ_0. Then $\Pi := \mathsf{Unr}_r[\Sigma] = \mathsf{FS}[\mathsf{pU}[\Sigma]^r]$ is a proof of knowledge with online extractability in the QROM (as in Definition 3.1) with $\varepsilon_{\mathrm{sim}} = 0$ and*

$$\varepsilon_{\mathrm{ex}}(\lambda, q, n) \leq (22r\ell_0 + 60)q^3 2^{-n} + 20q^2 \left(p^{\mathfrak{S}}_{triv}\right)^r . \tag{13}$$

The online extractor for Π runs in time $T^{\Pi}_{\mathcal{E}} \leq rT^{\mathsf{pU}[\Sigma]}_{\mathcal{E}} + O(q^2) \cdot poly(n, B)$, where $T^{\mathsf{pU}[\Sigma]}_{\mathcal{E}}$ is the runtime of $\mathsf{pU}[\Sigma]$'s $\mathfrak{S}$-soundness extractor as given in Proposition 4.5.*

5 Online Extractability of the FS-Transformation: The Case of Merkle-tree-based C&O Protocols

For an ordinary C&O protocol with reasonable concrete security (e.g., 128 bits), the number of commitments ℓ might be considerable. In this case, the communication complexity of the protocol (and thus the size of the non-interactive proof system, or digital-signature scheme, obtained via the Fiat-Shamir transformation) can be reduced by using a *Merkle tree* to collectively commit to the ℓ strings m_i. Such a construction is mentioned in [Fis05], and it is used in the construction of the digital-signature schemes Picnic2 and Picnic3 [KKW18, KZ20, CDG+19a]. The Merkle-tree-based C&O mechanism shrinks the commitment information from $\ell \cdot n$ to n, at the expense of increasing the cost of opening $|c|$ values m_i by an additive term of about $\lessapprox |c| \cdot n \cdot \log \ell$.

The cost of opening can, in fact, be slightly reduced again, by streamlining the opening information. When opening several leaves of a Merkle tree, the authentication paths overlap, so opening requires a number of hash values less than h per leaf, where h is the height of the tree. This overlap was observed and exploited in the octopus authentication algorithm which constitutes one of the optimizations of the stateless hash-based signature scheme gravity-SPHINCS [AE18], as well as in Picnic2 and Picnic3 [KZ20]. In the following section, we formalize tree-based collective commitment schemes with "octopus" opening.

5.1 Merkle-tree-based C&O Protocols

In line with Remark 3.3, we can consider C&O protocols with a different choice of commitment scheme, compared to the default choice of committing by element-wise hashing. Here, we discuss a particular choice of an alternative commitment scheme, which gives rise to more efficient C&O protocols in certain cases when ℓ is large. Informally, we consider C&O protocols where $m_1, \ldots, m_\ell$ is committed to by using a *Merkle tree*, and individual m_i's are opened by announcing the corresponding authentication paths.

To make this more formal, we introduce the following notation. For simplicity, we assume that ℓ is a power of 2, and thus $\ell = 2^h$ for $h \in \mathbb{N}$. We then consider the *full binary tree* $\mathsf{Tree} = \{0,1\}^{\leq h}$ of depth h, where the vertices are identified by bit strings. The root is denoted by $\emptyset$; the i-th leave is denoted by $\mathrm{lf}(i) \in \{0,1\}^h$ and is given by the binary representation of $i \in [\ell]$. The *authentication path* for the i-th leaf is the subtree that consists of all the ancestors of $\mathrm{lf}(i)$ and their siblings:

$$\mathsf{Auth}(i) := \mathsf{Anc}(\mathrm{lf}(i)) \cup \{\mathsf{sib}(v) \,|\, \emptyset \neq v \in \mathsf{Anc}(\mathrm{lf}(i))\} ,$$

where $\mathsf{Anc}(v) := \{u \in \mathsf{Tree} \,|\, \exists\, w : u \| w = v\}$ and $\mathsf{sib}(u\|b) := u\|(1-b)$ for any $b \in \{0,1\}$. Finally, for any subset $c \subseteq \{1, \ldots, \ell\}$, we let $\mathsf{Auth}(c) := \bigcup_{i \in c} \mathsf{Auth}(i)$ be the union of the authentication paths of the considered leaves, and we define the *octopus* $\mathsf{Octo}(c)$ to be the restriction of $\mathsf{Auth}(c)$ to its leaves, but excluding the leaves $\mathrm{lf}(i)$ for $i \in c$, i.e.,

$$\mathsf{Octo}(c) := \mathsf{leaves}(\mathsf{Auth}(c)) \setminus \{\mathrm{lf}(i) \,|\, i \in c\}$$

where, for any subtree T of Tree, $\mathsf{leaves}(T) := \{v \in T \,|\, (v\|0), (v\|1) \notin T\}$.

Extending on the above notation, for a given hash function $H : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X} = \{0,1\}^{\leq B}$ and $\mathcal{Y} = \{0,1\}^n$ for sufficiently large B, we define the *Merkle tree* of $\mathbf{m} = (m_1, ..., m_\ell) \in \mathcal{X}^\ell$ to be the *labeled* binary tree that has its leaves $\mathrm{lf}(1), \ldots, \mathrm{lf}(\ell)$ labeled by $H(m_1), ..., H(m_\ell)$, respectively, and each internal vertex is labeled by the hash of the labels of its two children. Formally,

$$\mathsf{MTree}_H(\mathbf{m}) := \big\{\big(v, l_v(\mathbf{m})\big) \,\big|\, v \in \mathsf{Tree}\big\}$$

with the labeling $l_v(\mathbf{m})$ recursively defined as

$$l_v(\mathbf{m}) := H\big(l_{v\|0}(\mathbf{m})\|l_{v\|1}(\mathbf{m})\big) \;\text{ for }\; v \in \{0,1\}^{<h}$$

and

$$l_{\mathrm{lf}(i)}(\mathbf{m}) := H(m_i) \;\text{ for }\; i \in \{1, \ldots, \ell\}\,,$$

where we leave the dependency of the labeling on H, i.e., $l_v = l_v^H$, implicit. We also write $\mathsf{MRoot}_H(\mathbf{m})$ then for the root label $l_\emptyset(\mathbf{m})$. In the same spirit, we write $\mathsf{MAuth}_H(c, \mathbf{m}) := \big\{\big(v, l_v(\mathbf{m})\big) \,\big|\, v \in \mathsf{Auth}(c)\big\}$ for the labeled authentication path and $\mathsf{MOcto}_H(c, \mathbf{m}) := \big\{\big(v, l_v(\mathbf{m})\big) \,\big|\, v \in \mathsf{Octo}(c)\big\}$ for the labeled octopus, using the same labeling function as for the Merkle tree.

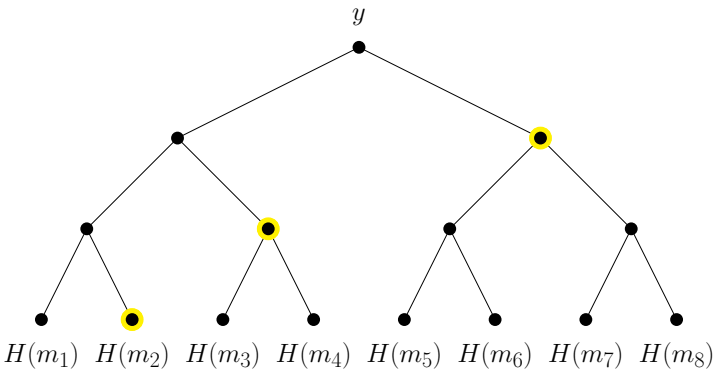

Fig. 3. The Merkle tree $\mathsf{MTree}_H(\mathbf{m})$ for $\mathbf{m} = (m_1, \ldots, m_8)$ with $\mathsf{MRoot}_H(\mathbf{m}) = y$. The yellow vertices mark the octopus $\mathsf{MOcto}_H(\{1\}, \mathbf{m})$, which is revealed (along with m_1) when opening the commitment y to m_1.

A *Merkle-tree-based C&O* protocol is now defined to be a variation of a C&O protocol, as hinted at in Remark 3.3, where the first message of the protocol, i.e., the commitment of $\mathbf{m} = (m_1, \ldots, m_\ell)$, is computed as $y = \mathsf{MRoot}_H(\mathbf{m})$, and the response z for challenge-set c then consists of the messages $\mathbf{m}_c = (m_i)_{i \in c}$ together with $O = \mathsf{MOcto}_H(c, \mathbf{m})$. The verifier $\mathcal{V}$ then accepts if and only if $\mathbf{m}_c$ and O "hash down to" y and the predicate $V(\lambda, \mathit{inst}, c, \mathbf{m}_c, a)$ is satisfied. More formally, the former means that $\mathcal{V}$ computes $\mathsf{MAuth}_H(c, \mathbf{m})$ from $O \cup \{(\mathrm{lf}(i), H(m_i)) \,|\, i \in c\}$ in the obvious way, and then checks whether $l_\emptyset(\mathbf{m}) = y$. This verification is denoted by $\mathit{OctoVerify}^H(c, y, \mathbf{m}_c, O)$, see Fig. 4.

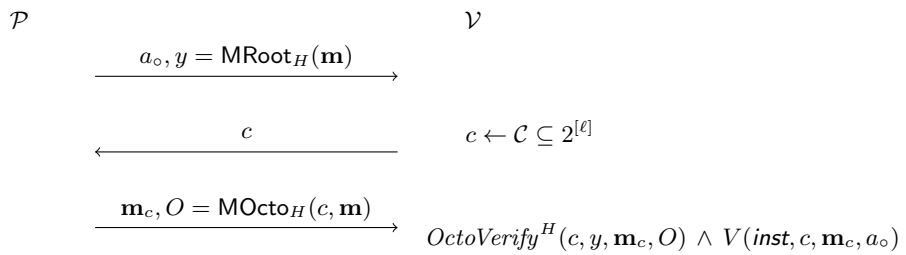

Fig. 4. A Merkle-tree based C&O Σ-protocol, formally introduced in Section 5.1.

Looking ahead, we may also consider a variation where the verifier resamples the challenge c if the resulting octopus is bigger than a given bound. Formally, this means that the challenge space of the Merkle-tree-based C&O protocol is restricted to those challenges $c \in [\ell]$ for which $\mathsf{Octo}(c)$ is not too large.

5.2 Online Extractability of the Fiat-Shamir Transformation

The analysis in Section 4 can be generalized to the case of FS-transformed Merkle-tree-based C&O protocols. To that end, we generalize the notation from that section as follows. Let Π be a Merkle-tree-based C&O protocol with number of messages to be committed equal to $\ell = 2^h$ where h is the height of the commitment Merkle tree.[13]

For a given database $D \in \mathfrak{D}$, recall from Section 4 the definition of D^{-1}; applied to a tuple $\mathbf{y} = (y_1, \ldots, y_\ell) \in \mathcal{Y}^\ell$ of commitments, D^{-1} attempts to recover the corresponding committed messages $m_1, \ldots, m_\ell$. Here, in a similar spirit but now considering the Merkle-tree commitment, MRoot_D^{-1} attempts to recover the committed messages from the root label of the Merkle tree.

In more detail, for a commitment $y \in \mathcal{Y} = \{0,1\}^n$ we reverse engineer the Merkle tree in the obvious way (see Fig. 5 for an example); namely, accepting a small clash in notation with the labeling function $l_v(\mathbf{m})$ defined for a tuple $\mathbf{m} \in \mathcal{M}^\ell$, we set the root label $l_\emptyset(y) := y$, and recursively define

$$\big(l_{v\|0}(y), l_{v\|1}(y)\big) := \mathsf{split} \circ D^{-1}\big(l_v(y)\big) \in \mathcal{Y} \times \mathcal{Y}$$

for $\emptyset \neq v \in \{0,1\}^{\leq h}$, where split maps any $2n$-bit string, parsed as $y_1 \| y_2$ with $y_1, y_2 \in \{0,1\}^n$, to the pair (y_1, y_2) of n-bit strings, while it maps anything else to $(\bot, \bot)$. Then, accepting a small clash in notation again, we set

$$\mathsf{MTree}_D(y) := \{l_v(y) \,|\, v \in \{0,1\}^{\leq h}\}\,,$$

and finally

$$\mathsf{MRoot}_D^{-1}(y) := \big(D^{-1}\big(l_{\mathrm{lf}(1)}(y)\big), \ldots, D^{-1}\big(l_{\mathrm{lf}(\ell)}(y)\big)\big)\,.$$

Following the strategy we used in Section 4, we define the database property

$$\mathsf{SUC} := \left\{ D \;\middle|\; \begin{array}{c} \exists\, y \in \mathcal{Y} \text{ and } \mathit{inst} \in \mathcal{I} \text{ so that } \mathbf{m} := \mathsf{MRoot}_D^{-1}(y) \text{ satisfies} \\ V(\mathit{inst}, c, \mathbf{m}_c) \text{ for } c := \gamma \circ D(\mathit{inst}, y) \text{ and } \big(\mathit{inst}, \mathcal{E}^*(\mathit{inst}, \mathbf{m})\big) \notin R \end{array} \right\},$$

and our first goal is to show that $[\![\bot \overset{q}{\Longrightarrow} \mathsf{SUC} \cup \mathsf{CL}]\!]$ is small.

Lemma 5.1. *Let Π be an $\mathfrak{S}$-sound C&O protocol with $p^{\mathfrak{S}}_{triv}$ as defined in (5). Then*

$$[\![\bot \overset{q}{\Longrightarrow} \mathsf{SUC} \cup \mathsf{CL}]\!] \leq 2eq^{3/2}2^{-n/2} + q\sqrt{10\max\left(q\ell \cdot 2^{-n+1}, p^{\mathfrak{S}}_{triv}\right)}\,.$$

The proof works exactly as the proof of Lemma 4.1, accounting for some syntactic differences due to the Merkle tree commitment. In particular, where in Case 1 and 2 of the proof of Lemma 4.1 we have to exclude U from falling on one of the hash values $y_1, \ldots, y_\ell$ in order to keep the $\mathbf{m}$ that was constructed from the database intact, we now have a similar restriction for U, but with respect to the whole tree $\mathsf{MTree}_D(y)$.

Proof. As in the proof of of Lemma 4.1, we can bound

$$[\![\bot \overset{q}{\Longrightarrow} \mathsf{SUC} \cup \mathsf{CL}]\!] \leq \sum_{s=0}^{q-1} \left([\![\mathsf{SZ}_{\leq s} \backslash \mathsf{CL} \to \mathsf{CL}]\!] + [\![\mathsf{SZ}_{\leq s} \backslash \mathsf{SUC} \to \mathsf{SUC}]\!]\right) \tag{14}$$

and use that

$$[\![\mathsf{SZ}_{\leq s} \backslash \mathsf{CL} \to \mathsf{CL}]\!] \leq 2e\sqrt{(s+1)/2^n} \leq 2e\sqrt{q/2^n}\,. \tag{15}$$

Thus, it remains to control the second term, which we will do again by means of Theorem 2.4 with $\mathsf{P} := \mathsf{SZ}_{\leq s} \setminus \mathsf{SUC}$ and $\mathsf{P}' := \mathsf{SUC}$.

To this end, we consider arbitrary but fixed $D \in \mathfrak{D}$ and input $x \in \mathcal{X}$. By Remark 2.5, we may assume that $D(x) = \bot$. Furthermore, for $\mathsf{P}|_{D^x}$ to be non-empty, it must be that $D \in \mathsf{SZ}_{\leq s}$, i.e., D is bounded in size. We now distinguish between the following cases for the considered D and x.

[13] As in the previous section we assume that ℓ is a power of 2 for ease of exposition.

Case 1: $D \in \mathsf{SUC}$. In particular, $\perp \in \mathsf{SUC}|_{D|^x} = \mathsf{P}'_{D|^x}$. So, Theorem 2.4 instructs us to set $\mathsf{L} := \mathsf{P}_{D|^x}$, where we leave the dependency of L on D and x implicit. Given that $D \in \mathsf{SUC}$, we can consider inst and y as promised by the definition of SUC above, i.e., such that $V(\mathit{inst}, c, \mathbf{m}_c)$ and $(\mathit{inst}, \mathcal{E}^*(\mathit{inst}, \mathbf{m})) \notin R$ for

$$c := \gamma \circ D(\mathit{inst}, y) \quad \text{and} \quad \mathbf{m} := \mathsf{MRoot}_D^{-1}(y)\,. \tag{16}$$

Note that, since $D(x) = \perp$ and $V(\mathit{inst}, c, \mathbf{m}_c)$ holds, which in particular means that c must be defined, it must be that $x \neq (\mathit{inst}, y)$. Therefore

$$\gamma \circ D(\mathit{inst}, y) = \gamma \circ D[x \mapsto u](\mathit{inst}, y)\,. \tag{17}$$

Our goal now is to show the final implication in

$$u \in \mathsf{L} \iff D[x \mapsto u] \in \mathsf{P} \implies D[x \mapsto u] \notin \mathsf{SUC} \implies u \in \mathsf{MTree}_D(y)\,.$$

We will do this by showing that $u \notin \mathsf{MTree}_D(y)$ implies

$$\mathsf{MRoot}_D^{-1}(y) = \mathsf{MRoot}_{D[x \mapsto u]}^{-1}(y)\,. \tag{18}$$

Indeed, the contraposition $u \notin \mathsf{MTree}_D(y) \Rightarrow D[x \mapsto u] \in \mathsf{SUC}$ of the claimed implication then follows from the fact that (17) and (18) together imply that c and $\mathbf{m}$ remain unchanged when replacing D by $D[x \mapsto u]$ in (16), and so $D[x \mapsto u] \in \mathsf{SUC}$ as well.

Towards showing (18), exploiting again that $D(x) = \perp$, it follows by definition of the reverse engineered labeling function $l_v(y)$ that $x \neq (l_{v||0}(y), l_{v||1}(y))$ for any v with $l_{v||0}(y) \neq \perp \neq l_{v||1}(y)$, i.e., x is not equal to any pair of siblings in $\mathsf{MTree}_D(y)$ with non-$\perp$ labeling (see Figure 5). Due to a similar reasoning, $x \neq m_i$ for any i. It now follows by definition of the reverse engineered Merkle tree and of MRoot^{-1} that if $u \notin \mathsf{MTree}_D(y)$ then $\mathsf{MTree}_D(y) = \mathsf{MTree}_{D[x \mapsto u]}(y)$ and $\mathsf{MRoot}_D^{-1}(y) = \mathsf{MRoot}_{D[x \mapsto u]}^{-1}(y)$, as claimed.

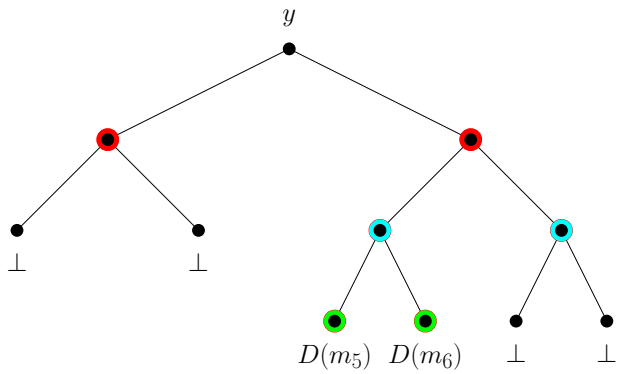

Fig. 5. Example of a reverse engineered Merkle tree $\mathsf{MTree}_D(y)$, with the $\perp$-children of the $\perp$-labels omitted. Since $D(x) = \perp$, $x \neq (l_u(y), l_w(y))$ for any two siblings (u, w) in $\mathsf{MTree}_D(y)$, i.e., nodes with the same color. Assuming that $u \notin \mathsf{MTree}_D(y)$ then implies that reprogramming D to $D[x \mapsto u]$ does not affect the reverse engineered Merkle tree.

Thus, we can bound

$$P[U \in \mathsf{L}] \leq P[U \in \mathsf{MTree}_D(y)] \leq \frac{2 \cdot 2^h - 1}{|\mathcal{Y}|} = \frac{2\ell - 1}{|\mathcal{Y}|}\,. \tag{19}$$

Case 2: $D \notin \mathsf{SUC}$, and x is a "commit query", i.e., $x = m \in \mathcal{M}$ or $x = (l_{v\|0}, l_{v\|1})$ for two labels $l_{v\|0}, l_{v\|1} \in \mathcal{Y}$. In particular, $\perp \notin \mathsf{P}'|_{D|^x}$ (given that $D(x) = \perp$) and so in the light of Theorem 2.4 we may choose $\mathsf{L} := \mathsf{P}'|_{D|^x}$. We then have

$$u \in \mathsf{L} \iff D[x \mapsto u] \in \mathsf{P}' = \mathsf{SUC} \implies \exists\, \mathit{inst}, y : D(\mathit{inst}, y) \neq \perp \wedge u \in \mathsf{MTree}_D(y)$$

where final implication can be seen as follows. By definition of SUC, the assumption $D[x \mapsto u] \in \mathsf{SUC}$ implies the existence of inst and y with $V(\mathit{inst}, c, \mathbf{m}_c)$ and $\big(\mathit{inst}, \mathcal{E}^*(\mathit{inst}, \mathbf{m})\big) \notin R$ for

$$c := \gamma \circ D[x \mapsto u](\mathit{inst}, y) = \gamma \circ D(\mathit{inst}, y) \quad \text{and} \quad \mathbf{m} := \mathsf{MRoot}^{-1}_{D[x \mapsto u]}(y)\,,$$

where the equality in the definition of c exploits that x is not a "challenge" query. The fact that $V(\mathit{inst}, c, \mathbf{m_c})$ is satisfied for this c thus implies that $D(\mathit{inst}, y) \neq \bot$. Next, with the goal to reach a contradiction, assume that $u \notin \mathsf{MTree}_D(y)$. Then for all $\bot \neq h \in \mathsf{MTree}_D(y)$ we have that $D^{-1}(h) = D[x \mapsto u]^{-1}(h)$ except if $D(x) = h$, but this cannot be since $D(x) = \bot$. It follows that $\mathsf{MTree}_D(y) = \mathsf{MTree}_D[x \mapsto u](y)$ and $\mathsf{MRoot}_D^{-1}(y) = \mathsf{MRoot}^{-1}_{D[x \mapsto u]}(y)$. The above then implies that $D \in \mathsf{SUC}$, a contradiction.

Thus, we can bound

$$P[U \in \mathsf{L}] \leq P[\,\exists\, \mathit{inst}, y : D(\mathit{inst}, y) \neq \bot \wedge U \in \mathsf{MTree}_D(y)] \leq \frac{s(2\ell - 1)}{|\mathcal{Y}|} \leq \frac{q(2\ell - 1)}{|\mathcal{Y}|}\,. \tag{20}$$

Case 3: $D \notin \mathsf{SUC}$, and x is a "challenge query", i.e., $x = (\mathit{inst}, y) \in \mathcal{I} \times \mathcal{Y}$. Set $\mathbf{m} := \mathsf{MRoot}_D^{-1}(y)$. Again, we have that $\bot \notin \mathsf{SUC}|_{D|^x} = \mathsf{P}'_{D|^x}$, and so by Theorem 2.4 we may set $\mathsf{L} := \mathsf{P}'_{D|^x}$. Here, we can argue that

$$\begin{aligned} u \in \mathsf{L} &\iff D[x \mapsto u] \in \mathsf{P}' = \mathsf{SUC} \\ &\implies V(\mathit{inst}, \gamma(u), \mathbf{m}_{\gamma(u)}) \text{ and } \big(\mathit{inst}, \mathcal{E}^*(\mathit{inst}, \mathbf{m})\big) \notin R\,, \end{aligned}$$

where the final implication can be seen as follows. By definition of SUC, the assumption $D[x \mapsto u] \in \mathsf{SUC}$ implies the existence of inst' and y' with $V(\mathit{inst}', c, \mathbf{m}'_c)$ and $\big(\mathit{inst}', \mathcal{E}^*(\mathit{inst}', \mathbf{m}')\big) \notin R$ for

$$c := \gamma \circ D[x \mapsto u](\mathit{inst}', y') \quad \text{and} \quad \mathbf{m}' := \mathsf{MRoot}^{-1}_{D[x \mapsto u]}(y') = \mathsf{MRoot}^{-1}_{D}(y')\,,$$

where the very last equality exploits that x is not a "commit" query. With the goal to come to a contradiction, assume that $(\mathit{inst}', y') \neq (\mathit{inst}, y) = x$. Then, $c = \gamma \circ D[x \mapsto u](\mathit{inst}', y') = \gamma \circ D(\mathit{inst}', y')$, and the above then implies that $D \in \mathsf{SUC}$, a contradiction. Thus, $(\mathit{inst}', y') = (\mathit{inst}, y) = x$. In particular, $\mathbf{m}' = \mathbf{m}$ and $c = \gamma \circ D[x \mapsto u](\mathit{inst}', y') = \gamma \circ D[x \mapsto u](x) = \gamma(u)$. Hence, the claimed implication holds.

Thus, we can bound

$$\begin{aligned} P[U \in \mathsf{L}] &\leq P[V(\mathit{inst}, \gamma(U), \mathbf{m}_{\gamma(U)}) \wedge \big(\mathit{inst}, \mathcal{E}^*(\mathit{inst}, \mathbf{m})\big) \notin R] \\ &\leq P[V(\mathit{inst}, \gamma(U), \mathbf{m}_{\gamma(U)}) \wedge S := \{c \,|\, V(\mathit{inst}, c, \mathbf{m}_c)\} \notin \mathfrak{S}] \\ &\leq P[\gamma(U) \in S := \{c \,|\, V(\mathit{inst}, c, \mathbf{m}_c)\} \notin \mathfrak{S}] \\ &\leq \max_{S \notin \mathfrak{S}} P[\gamma(U) \in S] \\ &\leq p^{\mathfrak{S}}_{triv}\,. \end{aligned} \tag{21}$$

By Theorem 2.4, we now get

$$\begin{aligned} [\![\mathsf{SZ}_{\leq s} \backslash \mathsf{SUC} \backslash \mathsf{CL} \to \mathsf{SUC}]\!] &\leq \max_{x,D} \sqrt{10 P\big[U \in \mathsf{L}^{x,D}\big]} \\ &\leq \sqrt{10}\sqrt{\max\left(\frac{2\ell - 1}{|\mathcal{Y}|}, \frac{q(2\ell - 1)}{|\mathcal{Y}|}, p^{\mathfrak{S}}_{triv}\right)} \\ &\leq \sqrt{10}\sqrt{\max\left(q\ell \cdot 2^{-n+1}, p^{\mathfrak{S}}_{triv}\right)}, \end{aligned}$$

where we have used Equations (19), (20) and (21) in the second inequality. Combining with Equations (15) and (14) yields the desired bound. □

Similarly to Theorem 4.2, we now obtain the following.

Theorem 5.2. *Let Π be an $\mathfrak{S}$-sound* Merkle-tree-based C&O protocol with challenge space $\mathcal{C}_\lambda$. Then* $\mathsf{FS}[\Pi]$ *is a proof of knowledge with online extractability in the QROM (as in Definition 3.1), with* $\varepsilon_{\mathrm{sim}}(\lambda, q, n) = 0$ *and*

$$\begin{aligned}\varepsilon_{\mathrm{ex}}(\lambda, q, n) &\leq 2(\kappa \log \ell + 1) \cdot 2^{-n} + \left(2eq^{3/2}2^{-n/2} + q\sqrt{10 \max\left(q\ell \cdot 2^{-n+1}, p^{\mathfrak{S}}_{triv}\right)}\right)^2 \\ &\leq (22\ell \log \ell + 60)\, q^3 2^{-n} + 20q^2 p^{\mathfrak{S}}_{triv}\end{aligned}$$

where $\kappa = \kappa(\lambda) := \max_{c \in \mathcal{C}_\lambda} |c|$ and ℓ is the number of leaves of the Merkle-tree-based commitment. The running time of the extractor is dominated by running the compressed oracle, which has complexity $O(q^2) \cdot poly(n, B)$, and by computing $\mathsf{MRoot}_D^{-1}(y)$ and running $\mathcal{E}^*$.

Here again the proof follows exactly the outline of its counterpart from Section 4.2, with some minor alterations to cope with the formalism of a Merkle-tree based C&O Σ-protocol. The difference in the bound is simply due to the difference between Lemmas 4.1 and 5.1.

Proof. We consider an arbitrary but fixed $\lambda \in \mathbb{N}$. Let $\mathcal{P}^*$ be a dishonest prover that, after making q queries to a random oracle H, outputs and instance inst and a proof $\pi = (y, \mathbf{m}_\circ, O)$ plus some (possibly quantum) auxiliary output Z, where O is an authentication octopus as defined in Section 5.1. For simplicity, we assume that $|c|$ is the same for all $c \in \mathcal{C}_\lambda$, and thus equal to κ. If it is not, we could always make the prover output a couple of dummy outputs m_i to match the upper bound on $|c|$. In the experiment $\mathcal{V}^{\mathcal{E}} \circ \mathcal{P}^{*\mathcal{E}}(\lambda)$, our extractor $\mathcal{E}$ works as follows while simulating all queries to H (by $\mathcal{P}^*$ and $\mathcal{V}$) with the compressed oracle:

1. Run $\mathcal{P}^*(\lambda)$ to obtain (inst, π, Z) with $\pi = (y, \mathbf{m}_\circ, O)$.
2. Compute $v \leftarrow \mathcal{V}^H(\mathsf{inst}, \pi)$, given by the truth value of

$$\mathit{OctoVerify}^H(c, y, \mathbf{m}_\circ, O) \quad \wedge \quad V(\mathsf{inst}, c, \mathbf{m}_\circ) \quad \text{with} \quad c := \gamma(H(\mathsf{inst}, y))\,.$$

3. Measure the internal state of the compressed oracle to obtain D.
4. Run $\mathcal{E}^*$ on input $\mathsf{MRoot}_D^{-1}(y)$ to obtain w.

Note that in the views of both $\mathcal{P}^*$ and $\mathcal{V}$, the interaction with H and the interaction with $\mathcal{E}$ differ only in that their oracle queries are answered by a compressed oracle instead of a real RO in the latter case. This simulation is perfect and therefore $\varepsilon_{\mathrm{sim}}(\lambda, q, n) = 0$.

Considering $\mathcal{P}^*$ as the algorithm $\mathcal{A}$ in Corollary 2.7, the composition $\mathcal{V} \circ \mathcal{P}^*$ then matches up with the algorithm $\tilde{\mathcal{A}}$ for $\mathcal{F} = \mathcal{V}$. Thus, noting that $\kappa(\log \ell + 1)$ is an upper bound on the amount of queries that *OctoVerify* makes,

$$\Pr\left[v \neq \mathcal{V}^D(\mathsf{inst}, \pi)\right] \leq 2(\kappa \log \ell + 1) \cdot 2^{-n}\,.$$

Therefore, we can bound bound the figure of merit $\varepsilon_{\mathrm{ex}}$ as

$$\begin{aligned}\varepsilon_{\mathrm{ex}}(\lambda, q, n) &= \Pr\left[v = 1 \wedge (\mathsf{inst}, w) \notin R\right] \\ &\leq \Pr\left[\mathcal{V}^D(\mathsf{inst}, \pi) \wedge (\mathsf{inst}, w) \notin R\right] + 2(\kappa \log \ell + 1) \cdot 2^{-n} \\ &\leq \Pr[\mathcal{V}^D(\mathsf{inst}, \pi) \wedge (\mathsf{inst}, w) \notin R \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}] \\ &\quad + \Pr[D \in \mathsf{SUC} \cup \mathsf{CL}] + 2(\kappa \log \ell + 1) \cdot 2^{-n}\,.\end{aligned}$$

Using the definition of $\mathcal{V}^D(\mathsf{inst}, \pi)$, understanding that $c := \gamma \circ D(\mathsf{inst}, y)$, we can write the first term as

$$\begin{aligned}&\Pr\left[\mathit{OctoVerify}^D(c, y, \mathbf{m}_\circ, O) \wedge V(\mathsf{inst}, c, \mathbf{m}_\circ) \wedge (\mathsf{inst}, w) \notin R \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}\right] \\ &\quad \leq \Pr\left[V(\mathsf{inst}, c, \mathbf{m}_c) \text{ for } \mathbf{m} := \mathsf{MRoot}_D^{-1}(y) \wedge (\mathsf{inst}, w) \notin R \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}\right] \\ &\quad \leq \Pr\left[D \in \mathsf{SUC} \,|\, D \notin \mathsf{SUC} \cup \mathsf{CL}\right] \\ &\quad = 0\,,\end{aligned}$$

where the first equality exploits that $D(m) = h$ iff $m = D^{-1}(h)$ for $D \notin \mathsf{CL}$.

We may thus conclude that

$$\begin{aligned}\varepsilon_{\mathrm{ex}}(\lambda, q, n) &\leq 2(\kappa \log \ell + 1) \cdot 2^{-n} \cdot 2^{-n} + \Pr\left[D \in \mathsf{SUC} \cup \mathsf{CL}\right] \\ &\leq 2(\kappa \log \ell + 1) \cdot 2^{-n} + \left[\!\left[\bot \overset{q}{\Longrightarrow} \mathsf{SUC} \cup \mathsf{CL}\right]\!\right]^2,\end{aligned}$$

where the last inequality is by definition of $\left[\!\left[\bot \overset{q}{\Longrightarrow} \cdot\right]\!\right]$. The claimed bound now follows from Lemma 5.1. □

5.3 Discussion: Application to Picnic, and Limiting the Proof Size

Application to Picnic. A prominent use case of C&O protocols is the construction of digital signature schemes via the Fiat-Shamir transformation. An important example is Picnic [CDG+17] currently under consideration as an alternate candidate in the NIST standardization process for post-quantum cryptographic schemes [NIS]. On a high level, the design of Picnic can be described as follows. A C&O Σ-protocol is constructed using the MPC-in-the-head paradigm [IKOS07]. Then, the Fiat-Shamir transformation is applied in the usual way to obtain a digital signature scheme. There are three evolutions of Picnic: Picnic-FS, Picnic 2 and Picnic 3.[14] Picnic-FS uses plain hash-based commitments, while Picnic 2 and Picnic 3 use a Merkle-tree-based collective commitment.

All three evolutions enjoy provable post-quantum security when the hash function used for the Fiat-Shamir transformation is modeled as a (quantum-accessible) RO. The best reduction applying to all of them proceeds as follows. First, Unruh's rewinding lemma [Unr12] is used to construct a knowledge extractor for the underlying Σ-protocol based on an appropriate $\mathfrak{S}$-soundness notion. Then, the *generic* QROM reduction for the Fiat-Shamir transformation from [DFMS19] is used to construct a knowledge extractor for the signature scheme in the QROM from the extractor for the Σ-protocol. Finally, the technique from [GHHM21] is used for simulating the chosen-message oracle to reduce breaking NMA (no-message attack) security to breaking CMA (chosen-message attack) security. This final step connects to the previous one because for the signature scheme the witness extracted from an NMA attacker is the secret key.

The first two steps in this chain of reductions, i.e. Unruh's rewinding and [DFMS19], are, however, not tight: The former loses at least a fifth power in the Picnic case, and the latter a factor of q^2, where q is the number of RO queries. This means that an NMA attacker with success probability ϵ can be used to break the underlying hard problem with probability $\Omega(\epsilon^5/q^{10})$ (or worse, depending on the Picnic variant).

For Picnic-FS (only), when in addition modeling the hash function used for the commitments as a RO, Unruh's rewinding can be replaced with the tight online extraction technique from [DFMS21]. The remaining loss due to the Fiat-Shamir reduction is of order ϵ/q^2, up to some additive terms accounting for search and collision finding in the RO, a sizable improvement over the above but still not tight.

By analyzing the Fiat-Shamir transformation of a C&O protocol (with or without Merkle tree commitments) directly, our results provide a tight alternative to the above lossy reductions. Using Theorems 4.2 (for Picnic-FS) and 5.2 (for Picnic 2 and Picnic 3) we can avoid all multiplicative/power losses in the reduction for NMA security. An NMA attacker with success probability ϵ can, in other words, be used to break the underlying hard problem with probability ϵ, up to some unavoidable additive terms accounting for search and collision finding in the RO.

An observation about octopus opening sizes. Depending on the parameters of the C&O protocol, the octopus opening information, $\mathsf{MOcto}(c, \mathbf{m})$ can be significantly smaller than the concatenation of the individual authentication paths. On the other hand, it is also *variable in size* (namely dependent on the choice of the challenge c), and the variance can be significant (see e.g. the computations for gravity SPHINCS in [AE18]). In the context of a digital signature scheme constructed via the Fiat-Shamir transformation of a Merkle-tree-based C&O protocol, like, e.g., Picnic 2 and Picnic 3, this leads to the undesirable property of a variable signature size, where signatures can be quite a bit larger in the worst case than on average. This might, e.g., lead to problems when looking for a drop-in replacement

[14] The original evolution also came with a variant using the Unruh transformation, Picnic-Ur. We restrict our attention to the variants using the Fiat-Shamir transformation.

for quantum-broken digital signature schemes for use in a larger protocol, where signatures need to be stored in a data field of fixed size.

One option to mitigate this situation is to cut off the tail of the octopus size distribution, i.e. to restrict the challenge space of the Merkle-tree-based C&O protocol to the set of challenges whose octopus is not larger than some bound. This can be done before applying the Fiat-Shamir transformation, e.g. using rejection sampling. In that way, one obtains a digital signature scheme with significantly reduced worst case signature size, at the expense of a tiny security loss.

5.4 The Merkle-Tree-Based Unruh Transformation

The Merkle tree based commitment mechanism can replace plain RO based commitments in *any* ordinary C&O protocol, in particular in $\Pi := \mathsf{pU}[\Sigma]$ for any Σ-protocol Σ. The result is a Merkle-tree-based C&O protocol and we obtain a corollary analogous to Corollary 4.6.

Corollary 5.3. *Let Σ be an $\mathfrak{S}$-sound Σ-protocol with challenge space size ℓ_0. Then $\mathsf{FS}[\mathsf{MPpU}_r[\Sigma]]$ is online-extractable with*

$$\varepsilon_{\mathrm{ex}} \leq (22r\ell_0 \log(r\ell_0) + 60)\, q^3 2^{-n} + 20q^2 \left(p^{\mathfrak{S}}_{triv}\right)^r \tag{22}$$

*where $\mathsf{MPpU}_r[\Sigma]$ is the **M**erkle-tree-based, **P**arallel-repeated, **p**re-**U**nruh transformation of Σ, i.e., the Merkle-tree-based C&O protocol obtained by replacing the commitments of $\mathsf{pU}[\Pi]^r$ with a Merkle-tree-based collective commitment.*

6 Acknowledgement

JD was funded by ERC-ADG project 740972 (ALGSTRONGCRYPTO). CM was funded by a NWO VENI grant (Project No. VI.Veni.192.159). CS was supported by a NWO VIDI grant (Project No. 639.022.519).

References

AE18. Jean-Philippe Aumasson and Guillaume Endignoux. Improving stateless hash-based signatures. In Nigel P. Smart, editor, *Topics in Cryptology – CT-RSA 2018*, pages 219–242, Cham, 2018. Springer International Publishing.

BDF+11. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, pages 41–69, Berlin, Heidelberg, 2011. Springer.

BdSGK+21. Carsten Baum, Cyprien Delpech de Saint Guilhem, Daniel Kales, Emmanuela Orsini, Peter Scholl, and Greg Zaverucha. Banquet: Short and fast signatures from aes. In Juan A. Garay, editor, *Public-Key Cryptography – PKC 2021*, pages 266–297, Cham, 2021. Springer International Publishing.

BLZ21. Jeremiah Blocki, Seunghoon Lee, and Samson Zhou. On the security of proofs of sequential work in a post-quantum world, 2021.

CDG+17. Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1825–1842, New York, NY, USA, 2017. ACM.

CDG+19a. Melissa Chase, David Derler, Steven Goldfeder, Jonathan Katz, Vladimir Kolesnikov, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, Xiao Wang, et al. The picnic signature scheme. *Submission to NIST Post-Quantum Cryptography project*, 2019.

CDG+19b. Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Picnic. https://www.microsoft.com/en-us/research/project/picnic/, retrieved on 9.4.2019, 2019.

CFHL21. Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 598–629, Cham, 2021. Springer International Publishing.

CGLQ20. Kai-Min Chung, Siyao Guo, Qipeng Liu, and Luowen Qian. Tight quantum time-space tradeoffs for function inversion. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 673–684, 2020.

Cha19. André Chailloux. Tight quantum security of the Fiat-Shamir transform for commit-and-open identification schemes with applications to post-quantum signature schemes. Cryptology ePrint Archive, Report 2019/699, version 1 Jul 2019, 2019. https://eprint.iacr.org/2019/699/20190701:091436.

Cha21. André Chailloux. Tight quantum security of the Fiat-Shamir transform for commit-and-open identification schemes with applications to post-quantum signature schemes. Cryptology ePrint Archive, Report 2019/699, version 16 Mar 2021, 2021. https://eprint.iacr.org/2019/699/20210316:124850.

CMS19. Alessandro Chiesa, Peter Manohar, and Nicholas Spooner. Succinct arguments in the quantum random oracle model. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography*, pages 1–29, Cham, 2019. Springer International Publishing.

DFM20. Jelle Don, Serge Fehr, and Christian Majenz. The measure-and-reprogram technique 2.0: Multi-round Fiat-Shamir and more. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020*, pages 602–631, Cham, 2020. Springer International Publishing.

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, *Advances in Cryptology – CRYPTO 2019*, pages 356–383, Cham, 2019. Springer International Publishing.

DFMS21. Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Online-extractability in the quantum random-oracle model. Cryptology ePrint Archive, Report 2021/280, 2021. https://eprint.iacr.org/2021/280.

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 Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018.

DKR+21. Christoph Dobraunig, Daniel Kales, Christian Rechberger, Markus Schofnegger, and Greg Zaverucha. Shorter signatures based on tailor-made minimalist symmetric-key crypto. Cryptology ePrint Archive, Report 2021/692, 2021. https://ia.cr/2021/692.

Fis05. Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, pages 152–168, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer.

GHHM21. Alex B. Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight adaptive reprogramming in the qrom. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 637–667, Cham, 2021. Springer International Publishing.

HM21. Yassine Hamoudi and Frédéric Magniez. Quantum Time-Space Tradeoff for Finding Multiple Collision Pairs. In Min-Hsiu Hsieh, editor, *16th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2021)*, volume 197 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:21, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

IKOS07. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, page 21–30, New York, NY, USA, 2007. Association for Computing Machinery.

KKW18. Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 525–537, New York, NY, USA, 2018. Association for Computing Machinery.

KZ20. Daniel Kales and Greg Zaverucha. Improving the performance of the picnic signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 154–188, 2020.

LZ19a. Qipeng Liu and Mark Zhandry. On finding quantum multi-collisions. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, pages 189–218, Cham, 2019. Springer International Publishing.

LZ19b. Qipeng Liu and Mark Zhandry. Revisiting post-quantum Fiat-Shamir. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 326–355, Cham, 2019. Springer International Publishing.

NIS. Nist post-quantum cryptography standardization. https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions,.

Unr12. Dominique Unruh. Quantum proofs of knowledge. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 135–152, Berlin, Heidelberg, 2012. Springer.

Unr15. Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, pages 755–784, Berlin, Heidelberg, 2015. Springer.

Wik18. Douglas Wikström. Special soundness revisited. Cryptology ePrint Archive, Report 2018/1157, 2018. https://ia.cr/2018/1157.

Zha19. Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 239–268, Cham, 2019. Springer International Publishing. Full Version (1 March 2019): https://eprint.iacr.org/2018/276/20190301:184107.

APPENDIX

A $\mathfrak{S}$-Soundness for Plain Σ-Protocols

Similar to Definition 3.5, for plain Σ-protocols (i.e., Σ-protocols in the standard model) the standard notion of special-soundness and k-soundness generalize as follows. Also here, $\mathfrak{S}$ is a non-empty, monotone increasing set of subsets $S \subseteq \mathcal{C}$, and $\mathfrak{S}_{\min} := \{S \in \mathfrak{S} \,|\, S_\circ \subsetneq S \Rightarrow S_\circ \notin \mathfrak{S}\}$, but now, a challenge $c \in \mathcal{C}$ is not (necessarily) a subset of $[\ell]$ anymore.

Definition A.1. *A Σ-protocol Π is called* $\mathfrak{S}$-sound *if there exists an efficient deterministic algorithm $\mathcal{E}_{\mathfrak{S}}(\mathsf{inst}, a, S, \{z_c\}_{c\in S})$ that takes as input an instance inst, a first message a, a subset $S \subseteq \mathcal{C}$ of challenges, and responses z_c for $c \in S$, and it outputs a witness for inst if $S \in \mathfrak{S}_{\min}$ and $\mathcal{V}(\mathsf{inst}, a, c, z_c)$ for all $c \in S$.* [15]

The common notion of a *special-sound* Σ-protocol is then the special case of a $\mathfrak{S}$-sound Σ-protocol with $\mathfrak{S} := \{S \subseteq \mathcal{C} \,|\, |S| \geq 2\}$, and similarly a *$k$-sound* Σ-protocol is a $\mathfrak{S}$-sound Σ-protocol with $\mathfrak{S} := \{S \subseteq \mathcal{C} \,|\, |S| \geq k\}$. Also here, using syntactically the same definition as in Equation (5),

$$p^{\mathfrak{S}}_{triv} := \frac{1}{|\mathcal{C}|} \max_{\hat{S}\notin\mathfrak{S}} |\hat{S}|$$

then captures the "trivial" attack that may potentially (and typically does) apply to a $\mathfrak{S}$-sound Σ-protocol, where the dishonest prover prepares a first message a so that he has valid responses z ready for all the challenges c in some arbitrarily chosen set $\hat{S} \notin \mathfrak{S}$.

[15] We note the clash in terminology with Definition 3.4. However, Definition 3.4 applies exclusively to C&O Σ-protocols in the (Q)ROM, whereas the definition here applies exclusively to Σ-protocol in the plain model; so there should be no confusion. The two definitions are of course related: a $\mathfrak{S}$-sound C&O Σ-protocol becomes a $\mathfrak{S}$-sound plain Σ-protocol when the commitments are instantiated with a perfectly binding commitment scheme (rather than with a hash function).