Post-Quantum Security of Fiat-Shamir

Dominique Unruh
University of Tartu

May 9, 2017

Abstract. The Fiat-Shamir construction (Crypto 1986) is an efficient transformation in the random oracle model for creating non-interactive proof systems and signatures from sigma-protocols. In classical cryptography, Fiat-Shamir is a zero-knowledge proof of knowledge assuming that the underlying sigma-protocol has the zero-knowledge and special soundness properties. Unfortunately, Ambainis, Rosmanis, and Unruh (FOCS 2014) ruled out non-relativizing proofs under those conditions in the quantum setting.

In this paper, we show under which strengthened conditions the Fiat-Shamir proof system is still post-quantum secure. Namely, we show that if we require the sigma-protocol to have computational zero-knowledge and *perfect* special soundness, then Fiat-Shamir is a zero-knowledge simulation-sound proof system (but not a proof of knowledge!). Furthermore, we show that Fiat-Shamir leads to a post-quantum secure strongly unforgeable signature scheme when additionally assuming a "dual-mode hard instance generator" for generating key pairs.

Finally, we study the extractability (proof of knowledge) property of Fiat-Shamir. While we have no proof of the extractability itself, we show that if we can prove extractability, then other desired properties such as simulation-sound extractability (i.e., non-malleability), and strongly unforgeable signatures follow.

Contents

1 Introduction

1.1 Background

Fiat-Shamir signatures. Signatures are (next to encryption) probably one of the most important constructs in modern cryptography. In search for efficient signature schemes, Fiat-Shamir [FS87] gave a construction for transforming many three-round identification schemes into signatures, using the random oracle. (The transformation was stated only for a specific case, but the general construction is an easy generalization. [FS87] also does not contain a complete security proof, but a proof was later provided by Pointcheval and Stern [PS96b].) The Fiat-Shamir transform and variations thereof have since been used in a large number of constructions (signatures [Sch91, PS00], group signatures [BBS04], anonymous credentials [CL01], e-voting [Adi08], anonymous attestation [BCC04], etc.) The benefit of the Fiat-Shamir transform is that it combines efficiency with universality: The underlying identification scheme can be any so-called sigma-protocol (see below), this allows for great flexibility in how public and secret key are related and enables the construction of more advanced signature schemes and related schemes such as group signatures, etc.

Non-interactive zero-knowledge proofs. At the first glance unrelated, but upon closer inspection intimately connected to signatures are non-interactive zero-knowledge proof of knowledge (NIZKPoK). In fact, Fiat-Shamir can also be seen as a highly efficient construction for NIZKPoKs in the random oracle model [FKMV12]. Basically, a NIZKPoK allows a prover to show his knowledge of a witness sk that stands in a given relation to a publicly known statement pk. From a NIZKPoK, we can derive a signature scheme: To sign a message m, the signer constructs a proof that he knows the secret key corresponding to the public key pk. (Of course, the message m needs to be included in the proof as well, we omit the details for now.) For this construction to work, the NIZKPoK needs to satisfy certain advanced security notions ("simulation-sound extractability");[1] Fiat-Shamir satisfies this notion in the classical setting [FKMV12]. Thus Fiat-Shamir doubles both as a signature scheme and as a NIZKPoK, leading to simple and highly efficient constructions of both.

The construction. In order to understand the rest of this introduction more easily, we sketch the construction of Fiat-Shamir (the precise definition is given in Definition 17). We will express it as a NIZKPoK since this makes the analysis more modular. (We study Fiat-Shamir as a signature scheme in Section 8.)

A sigma-protocol Σ is a three-message protocol: The prover (given a statement x and a corresponding valid witness w) sends a message com, called "commitment", to the verifier. The verifier (who knowns only the statement x) responds with a uniformly random "challenge" ch. Then the prover answers with his "response" $resp$, and the verifier checks whether $(com, ch, resp)$ is a valid interaction. If so, he accepts the proof of the statement x. In the following, we will assume that ch has superlogarithmic length, i.e., there are superpolynomially many different challenges. This can always be achieved by parallel-composing the sigma-protocol.

Given the sigma-protocol Σ, the Fiat-Shamir transform yields a non-interactive proof system: The prover P_{FS} internally executes the prover of the sigma-protocol to get the commitment com. Then he computes the challenge as $ch := H(x \| com)$ where H is a hash function, modeled as a random oracle. That is, instead of letting the verifier generate a random challenge, the prover produces it by hashing. This guarantees, at least on an intuitively level, that the prover does not have any control over the challenge, it is as if it was chosen randomly. Then the prover internally

[1] We do not know where this was first shown, a proof in the quantum case can be found in [Unr15].

produces the response *resp* corresponding to *com* and *ch* and sends the non-interactive proof $com \| resp$ to the verifier.

The Fiat-Shamir verifier V_{FS} computes $ch := H(x \| com)$ and checks whether $(com, ch, resp)$ is a valid interaction of the sigma-protocol.

Note that numerous variants of the Fiat-Shamir are possible. For example, one could compute $ch := H(com)$ (omitting x). However, this variant of Fiat-Shamir is malleable, see [FKMV12].

Difficulties with Fiat-Shamir. The Fiat-Shamir transform is a deceptively simple construction, but proving its security turns out to be more involved that one would anticipate. To prove security (specifically, the unforgeability property in the signature setting, or the extractability in the NIZKPoK setting), we need simulate the interaction of the adversary with the random oracle, and then rerun the same interaction with slightly changed random oracle responses ("rewinding"). The first security proof by Fiat and Shamir [FS87] overlooked that issue.[2] Bellare and Rogaway [BR93, Section 5.2] also prove the security of the Fiat-Shamir transform (as a proof system) but simply claim the soundness without giving a proof (we assume that they also overlooked the difficulties involved).[3] The first complete security proof of the Fiat-Shamir as a signature scheme is by Pointcheval and Stern [PS96b] who introduced the so-called "forking lemma", a central tool for analyzing the security of Fiat-Shamir (it allows us to analyze the rewinding used in the security proof). When considering Fiat-Shamir as a NIZKPoK, the first proof was given by Faust, Kohlweiss, Marson and Venturi [FKMV12]; they showed that Fiat-Shamir is zero-knowledge and simulation-sound extractable.[4] This short history of the security proofs indicates that Fiat-Shamir is more complicated than it may look at the first glance.

Further difficulties were noticed by Shoup and Gennaro [SG02] who point out that the fact that the Fiat-Shamir security proof uses rewinding can lead to considerable difficulties in the analysis of more complex security proofs (namely, it may lead to an exponential blowup in the running time of a simulator; Pointcheval and Stern [PS96a] experienced simular problems). Fischlin [Fis05] notes that the rewinding also leads to less tight reductions, which in turn may lead to longer key sizes etc. for protocols using Fiat-Shamir.

Another example of unexpected behavior: Assume Alice gets a n pairs of public keys (pk_{i0}, pk_{i1}), and then can ask for *one* of the secret keys for each pair (i.e., sk_{i0} or sk_{i1} is revealed, never both), and then Alice is supposed to prove using Fiat-Shamir that he knows *both* secret keys for *one* of the pairs. Intuitively, we expect Alice not to be able to do that (if Fiat-Shamir is indeed a proof of knowledge), but as we show in Appendix A, Fiat-Shamir does not guarantee that Alice cannot successfully produce a proof in this situation!

To circumvent all those problems, Fischlin [Fis05] gave an alternative construction of NIZKPoKs and signature schemes in the random oracle model whose security proof does not use rewinding. However, their construction seems less efficient in terms of the computation performed by the prover (although this is not fully obvious if the tightness of the reduction is taken into account), and their construction requires an additional property (unique responses[5]) from the

[2] The proof of [FS87, Lemma 6] claims without proof that a successful adversary cannot find a square root mod n of $\prod_{j=1}^{k} v_j^{c_j}$. In hindsight, this proof step would implicitly use the forking lemma [PS96b] that was developed only nine years later. [FS87] also mentions a full version of their paper, but to the best of our knowledge no such full version has ever appeared.

[3] A "final paper" is also mentioned, but to the best of our knowledge never appeared.

[4] They only sketch the zero-knowledge property, though. Their proof sketch overlooks one required property of the sigma-protocol: unpredictable commitments (Definition 6). Without this (easy to achieve) property, at least the simulator constructed in [FKMV12] will not work correctly. Concurrently and independently, [BPW12] also claims the same security properties, but the theorems are given without any proof or proof idea.

[5] Unique responses: It is computationally infeasible to find two valid responses for the same commitment/challenge pair. See Definition 6 below.

underlying sigma-protocol.

We do not claim that those difficulties in proving and using Fiat-Shamir necessarily speak against Fiat-Shamir. But they show one needs to carefully analyze which precise properties Fiat-Shamir provably has, and not rely on what Fiat-Shamir intuitively achieves.

Post-quantum security. In this paper we are interested in the post-quantum security of Fiat-Shamir. That is, under what conditions is Fiat-Shamir secure if the adversary has a quantum computer? In the post-quantum setting, the random oracle has to be modeled as a random function that can be queried in superposition[6] since a normal hash function can be evaluated in superposition as well (cf. [BDF$^+$11]). Ambainis, Rosmanis, and Unruh [ARU14] showed that in this model, Fiat-Shamir is insecure in general. More precisely, they showed that relative to certain oracles, there are sigma-protocols such that: The sigma-protocol satisfies the usual security properties. (Such as zero-knowledge and special soundness. These are sufficient for security in the classical case.) But when applying the Fiat-Shamir transform to it, the resulting NIZKPoK is not sound (and thus, as a signature, not unforgeable). Since this negative result is relative to specific oracles, it does not categorically rule out a security proof. However, it shows that no relativizing security proof exists, and indicates that it is unlikely that Fiat-Shamir can be shown post-quantum secure in general. Analogous negative results [ARU14] hold for Fischlin's scheme [Fis05].

Unruh [Unr15] gave a construction of a NIZKPoK/signature scheme in the random oracle model that is avoids these problems and is post-quantum secure (simulation-sound extractable zero-knowledge / strongly unforgeable). However, Unruh's scheme requires multiple executions of the underlying sigma-protocol, leading to increased computational and communication complexity in comparison with Fiat-Shamir which needs one a single execution.[7] Furthermore, Fiat-Shamir is simpler (in terms of the construction, if not the proof), and more established in the crypto community. In fact, a number of papers have used Fiat-Shamir to construct post-quantum secure signature schemes (e.g., [GKV10, LNW15, LLM$^+$16b, BK16, LLM$^+$16a, BrOP16]). The negative results by Ambainis et al. show that the post-quantum security of these schemes is hard to justify.[8] Thus the post-quantum security of Fiat-Shamir would be of great interest, both from a practical and theoretical point of view.

Is there a possibility to show the security of Fiat-Shamir notwithstanding the negative results from [ARU14]? There are two options (besides non-relavizing proofs): (a) Unruh [Unr12] introduced an additional condition for sigma-protocols, so-called "perfectly unique responses".[9] Unique responses means that for any commitment and challenge in a sigma-protocol, there exists at most one valid response. They showed that a sigma-protocol that additionally has perfect unique responses is a proof of knowledge while [ARU14] showed that without unique responses, a sigma protocol will not in general be a proof of knowledge (relative to some oracle). Similarly, [ARU14] does not exclude that Fiat-Shamir is post-quantum secure when the underlying

[6] E.g., the adversary can produce states such as $\sum_x 2^{-|x|/2}|x\rangle \otimes |H(x)\rangle$.

[7] This assumes that the underlying sigma-protocol has a large challenge space. If the underlying sigma-protocol has a small challenge space (e.g., the challenge is a bit) then for Fiat-Shamir the sigma-protocol needs to be parallel composed first to increase its challenge space. In this case, the complexity of Fiat-Shamir and Unruh are more similar. (See, e.g., [GCZ16] that compares (optimizations of) Fiat-Shamir and Unruh for a specific sigma-protocol and concludes that Unruh has an overhead in communication complexity of merely 60% compared to Fiat-Shamir.)

[8] We stress that the *classical* security of these schemes is not in question. Also, not all these papers explicitly claim to have post-quantum security. However, they all give constructions that are based on supposedly quantum hard assumptions. Arguably, one of the main motivations for using such assumptions is post-quantum security. Thus the papers do not claim wrong results, but they would be considerably strengthened by a proof of the post-quantum security of Fiat-Shamir.

[9] It is called "strict soundness" in [Unr12] but we use the term "unique responses" to match the language used elsewhere in the literature, e.g., [Fis05].

sigma-protocol has perfectly unique responses.[10] (b) If we do not require extractability, but only require soundness (i.e., if we only want to prove that there exists a witness, not that we know it), then [ARU14] does not exclude a proof that Fiat-Shamir is sound based on a sigma-protocol with perfect special soundness (but (computational) special soundness is not sufficient). In this paper, we mainly follow approach (b), but we also have some results related to research direction (a).

1.2 Our contribution

Security of Fiat-Shamir as a proof system. We prove that Fiat-Shamir is post-quantum secure as a proof system. More precisely, we prove that it is zero-knowledge (using random-oracle programming techniques from [Unr15]), and that it is sound (i.e., a proof of knowledge, using a reduction to quantum search). More precisely:

Theorem 1 (Post-quantum security of Fiat-Shamir – informal) *Assume that Σ has honest-verifier zero-knowledge and* perfect *special soundness.*

Then the Fiat-Shamir proof system (P_{FS}, V_{FS}) is zero-knowledge and simulation-sound.

The assumptions are the same as in the classical setting, except that instead of computational special soundness (as in in the classical case), we need perfect special soundness.[11] This is interesting, because it means that we need one of the properties of the sigma-protocol to hold unconditionally, even though we only want computational security in the end. However, [ARU14] shows that this is necessary: when assuming only computational special soundness, they construct a counter-example to the soundness of Fiat-Shamir (relative to some oracle).

Simulation-soundness. In addition to the above, we also show that Fiat-Shamir has simulation-soundness. Simulation-soundness is a property that guarantees non-malleability, i.e., that an adversary cannot take a proof gotten from, say, an honest participant and transform it into a different proof (potentially for a different but related statement).[12] This is particularly important when using Fiat-Shamir to construct signatures (see below) because we would not want the adversary to transform one signature into a different signature. Our result is:

Theorem 2 (Simulation-soundness of Fiat-Shamir – informal) *Assume that Σ has honest-verifier zero-knowledge, perfect special soundness, and unique responses.*

Then the Fiat-Shamir proof system (P_{FS}, V_{FS}) has simulation-soundness.

Note that unique responses are needed for the simulation-soundness of Fiat-Shamir even in the classical case.

Signatures. Normally, the security of Fiat-Shamir signatures is shown by reducing it to the simulation-sound extractability of Fiat-Shamir (implicitly or explicitly). Unfortunately, we do not know whether Fiat-Shamir is extractable in the quantum setting. Thus, we need a new proof of the security of Fiat-Shamir signatures that only relies on simulation-soundness. We can do so by making additional assumptions about the way the key generator works: We call an algorithm G a "dual-mode hard instance generator" if G outputs a key pair (pk, sk) in such a way that pk is computationally indistinguishable from an invalid pk (i.e., a pk that has no corresponding sk). An example of such an instance generator would be: sk is chosen uniformly at random, and $pk := F(sk)$ for a pseudo-random generator F. Then we have:

[10] Interestingly, *computational* unique responses as in footnote 5 are shown not to be sufficient, even when we want only *computational* extractability / unforgeability.

[11] The same property, except that it has to hold against computationally unlimited adversaries.

[12] Formally, simulation-soundness is defined by requiring that soundness holds even when the adversary has access to a simulator that produces fake proofs.

Theorem 3 (Fiat-Shamir signatures – informal) *Assume that G is a dual-mode hard instance generator. Fix a sigma-protocol Σ (for showing that a given public key has a corresponding secret key). Assume that Σ has honest-verifier zero-knowledge, perfect special soundness, and unique responses.*

Then the Fiat-Shamir signature scheme is strongly unforgeable.

Note that classically, we only require that G is a hard instance generator. That is, given pk, it is hard to find sk. We leave it as an open problem whether this is sufficient in the post-quantum setting, too.

On extractability. Although we were not able to prove that Fiat-Shamir is extractable, we make several steps towards a better understanding of the extractability and related questions:

(i) We formalize the definition of extractability of non-interactive zero-knowledge proofs in the quantum random oracle model. (Unruh [Unr15] already gave a definition of extractability in the quantum random oracle model, but their definition is only applicable to so-called online-extractable non-interactive proofs. Fiat-Shamir is not online-extractable, thus Unruh's definition cannot be used here.) The definition of extractability poses non-trivial challenges that do not occur in the classical setting: In the quantum setting, the extractor's actions may disturb the adversary's state (due to its measurements), the definition needs to reflect this.

(ii) We further define simulation-sound extractability in the quantum random oracle model. (Again, Unruh [Unr15] defined this property, but only for online-extractable proofs.) Roughly speaking, simulation-sound extractability guarantees that extractability even holds when the adversary has access to a simulator that produces fake proofs. This property is standard in the classical setting and ensures that proofs are non-malleable. That is, given a proof for some statement, it is not possible to transform it into a proof for a related statement. Simulation-sound extractability is, among other uses, very important for constructing signature schemes from Fiat-Shamir (see below).

(iii) Although we do not know how to prove that Fiat-Shamir is extractable (not even for a subclass of sigma-protocols), we can show: If Fiat-Shamir is extractable (for some sigma-protocol), then Fiat-Shamir is simulation-sound extractable (for the same sigma-protocol).

(iv) We show that if a non-interactive proof system is zero-knowledge and simulation-sound extractable, then it can be used as a strongly unforgeable signature scheme, using only standard assumptions about the key generator (namely, the secret key is hard to guess given the public key). In particular, this implies that if Fiat-Shamir is extractable, then Fiat-Shamir is a post-quantum secure signature scheme.

These latter contributions, although all based on the *assumption* that Fiat-Shamir is extractable, give us valuable insight into future research: They narrow down what is left to prove to a single property (extractability), from which then all remaining desired properties follow (such as simulation-sound extractability or unforgeability). And for existing post-quantum signature schemes that use Fiat-Shamir whose security does not already follow from Theorem 3, our research at least rules out some forms of attacks – if those signature schemes are insecure, then the attacks must be related to the lack of extractability (and not, e.g., to the zero-knowledge property, or to malleability).

Organization. In Section 2, we fix some simple notation. In Section 3, we formalize oracle machines, this is important for a precise formalization of extractability but can be skipped at first reading. In Section 4, we state some auxiliary lemmas needed throughout the paper. In Section 5, we discuss the (relatively standard) security notions for sigma-protocols used in this paper. In Section 6, we define security notions for non-interactive proof systems in the random oracle model. In particular, we give definitions of extractability and simulation-sound extractability, which are novel. In Section 7 we give out main results, the security properties of Fiat-Shamir (zero-knowledge, soundness, simulation-soundness, ...). In Section 8, we show how to construct signature schemes from non-interactive zero-knowledge proof systems, in particular from Fiat-Shamir.

Readers who are interested solely in conditions under which Fiat-Shamir signatures are post-quantum secure but not in the security proofs may restrict their attention to Sections 5 and 8 (in particular Corollary 30).

2 Preliminaries

$\mathrm{Fun}(n, m)$ is the set of all functions from $\{0,1\}^n$ to $\{0,1\}^m$.

A function $f : F \to \mathbb{R}$ with $F = \mathbb{N}, \mathbb{R}$ is *negligible* iff there exist $c > 0$ such that $f(x) \leq x^{-c}$ for sufficiently large x. $a \oplus b$ denotes the bitwise XOR between bitstrings (of the same length).

A function $f : F \to \mathbb{R}$ with $F = \mathbb{N}, \mathbb{R}$ is *noticeable* iff there exist $c > 0$ such that $f(x) \geq x^{-c}$ for sufficiently large x.

A *density operator* is a positive Hermitian operator of trace ≤ 1 on some Hilbert space. I denotes the identity matrix/operator.

The *fidelity* $F(\rho, \sigma)$ between density operators ρ and σ is defined as $F(\rho, \sigma) := \mathrm{tr}\,\sqrt{\rho^{1/2}\sigma\rho^{1/2}}$. $\mathcal{M}$ always stands for a complete measurement in the computational basis.

If H is a function, we write $H(x := y)$ for the function H' with $H'(x) = y$ and $H'(x') = H(x')$ for $x' \neq x$. We call a list $\mathit{ass} = (x_1 := y_1, \dots, x_n := y_n)$ an *assignment-list*. We then write $H(\mathit{ass})$ for $H(x_1 := y_1)(x_2 := y_2)\dots(x_n := y_n)$. (That is, H is updated to return y_i on input x_i, with assignments occurring later in ass taking precedence.)

We write $x \leftarrow A(\dots)$ to denote that the result of the algorithm/measurement A is assigned to x. We write $Q \leftarrow |\Psi\rangle$ or $Q \leftarrow \rho$ to denote that the quantum register Q is initialized with the quantum state $|\Psi\rangle$ or ρ, respectively. We write $x \overset{\$}{\leftarrow} M$ to denote that x is assigned a uniformly randomly chosen element of the set M.

We write $\Pr[P : G]$ for the probability that P holds after executing G. Here P is a predicate, and G is a sequence of instructions that define the free variables of P (i.e., G defines the distribution of those variables). For example $\Pr[a = b : a \overset{\$}{\leftarrow} \{0,1\}, b \overset{\$}{\leftarrow} \{0,1\}] = \frac{1}{2}$ denotes the probability that $a = b$ holds when a and b are uniformly random from $\{0,1\}$.

3 Oracle machines

In this section, we introduce our formalism for modeling oracle algorithms. Some of the definitions are standard, but for defining extractability and simulation-sound extractability, we need some more advanced concepts, e.g., we need to be able to model an oracle algorithm that has access to several oracles and then is in turn passed itself as an oracle to another algorithm. These advanced definitions are only needed for the results related to extractability and simulation-sound extractability. We mark them with "**(extractability only)**", they can be safely skipped for understanding the concepts in this paper that are unrelated to extractability. And when only an informal understanding of the results in this paper is required, the definitions can be skipped altogether.

Oracles. An *oracle* $\mathcal{O}$ consists of a *state space* $\mathcal{H}_\mathcal{O}$ and a quantum operation $\mathcal{E}_\mathcal{O}$ on $\mathbb{C}^{2^n} \otimes \mathcal{H}_\mathcal{O}$ for some n. We call $\mathbb{C}^{2^n}$ its *input/output space*.

The intuition is that $\mathcal{H}_\mathcal{O}$ will contain the hidden state of the oracle, while $\mathbb{C}^{2^n}$ contains the n-qubit oracle input/output before/after a query.

Functions as oracles. Given a function $H : \{0,1\}^n \to \{0,1\}^m$, let $\mathbf{U}_H : |x\|y\rangle \mapsto |x\|(y \oplus H(x))\rangle$ for $x \in \{0,1\}^n$, $y \in \{0,1\}^m$. H then induces an $(n+m)$-bit oracle with state space $\mathcal{H} = \mathbb{C}$ (zero qubit state) and $\mathcal{E}_H(\rho) := \mathbf{U}_H \rho (\mathbf{U}_H)^\dagger$ We call this oracle the *oracle for H* and denote it with H. (That is, we use denote the oracle for a function f with f. Context will always allow to decide whether we mean the oracle or the function.)

Pure oracle circuits (extractability only). A *pure oracle circuit C* consists of the following:

- t – the number of oracles that C expects.
- For each $j = 1, \dots, t$, an integer $\ell_{C,j}^{oracle}$. This indicates that the i-th oracle is expected to be have an $\ell_{C,j}^{oracle}$-qubit input/output register.
- An integer ℓ_C^{output}. This indicates that the circuit has an ℓ_C^{output}-bit classical output.
- ℓ_C^{state} – the number of qubits in the state of C.
- $\mathcal{U}_C$ – a unitary operating on registers $O_C, I_1, \dots, I_t, S_C$, Here O_C has ℓ_C^{output} qubits (and is supposed to contain the final classical output), I_i has $\ell_{C,i}^{oracle}$ qubits (and is used to contain input/output for oracle queries), S_C has ℓ_C^{state} qubits (and contains the internal state of C). $\mathcal{U}_C$ specifies the actual computation performed by C between oracle queries.
- $\mathbf{op}_C$ – the *execution schedule* of C, that is, a sequence $\mathbf{op}_C = op_1, \dots, op_s$ for some s where each op_i is either $\mathsf{compute}$ or call_j for some $j \in \{1, \dots, t\}$. $op_i = \mathsf{compute}$ means that the i-th action in the circuit is to apply $\mathcal{U}_C$, and $op_i = \mathsf{call}_j$ means that the i-th action is to invoke the j-th oracle.

For oracles $\mathcal{O}_1, \dots, \mathcal{O}_t$ with state spaces $\mathcal{H}_1, \dots, \mathcal{H}_t$, we denote an execution of the oracle by $x \leftarrow C^{\mathcal{O}_1(S_1), \dots, \mathcal{O}_t(S_t)}(S_C)$. S_C is an ℓ_C^{state}-qubit register (the initial state of C). S_i is a quantum register with space $\mathcal{H}_i$ (the initial state of the i-th oracle). And x will contain the classical output after the execution. (And S_C will contain a possibly modified state.) Formally, $x \leftarrow C^{\mathcal{O}_1(S_1), \dots, \mathcal{O}_t(S_t)}(S_C)$ is described by the following algorithm:

```
$O_C \leftarrow |0^{\ell_C^{output}}\rangle$
for $j = 1, \dots, t$ do
    $I_j \leftarrow |0^{\ell_{C,j}^{oracle}}\rangle$
for $i = 1$ to $m$ do
    if $op_i = \mathsf{call}_j$   (for some $j$) then
        apply $\mathcal{E}_{\mathcal{O}_j}$ to the registers $I_j, S_j$
    else if $op_i = \mathsf{compute}$ then
        apply $\mathcal{U}_C$ to $O_C, (I_i)_i, S_C$
let $x \leftarrow \mathcal{M}(O_C)$                    // measure $O_C$ in computational basis
```

We write $\mathcal{E}_C^{\mathcal{O}_1, \dots, \mathcal{O}_t}$ for the superoperator operating on registers $O_C, (I_j)_j, S_C, (S_j)_j$ that is given by the second loop in the above algorithm. With that notation, $x \leftarrow C^{\mathcal{O}_1(S_1), \dots, \mathcal{O}_t(S_t)}(S_C)$ denotes the following program: "$O_C \leftarrow |0\rangle$. $I_j \leftarrow |0\rangle$ for all j. Apply $\mathcal{E}_C^{\mathcal{O}_1, \dots, \mathcal{O}_t}$. $x \leftarrow \mathcal{M}(O_C)$."

Let

$$\mathbf{shape}_C := (t, (\ell_{C,j}^{oracle})_{j=1,\dots,t}, \ell_C^{output}, \mathbf{op}_C)$$

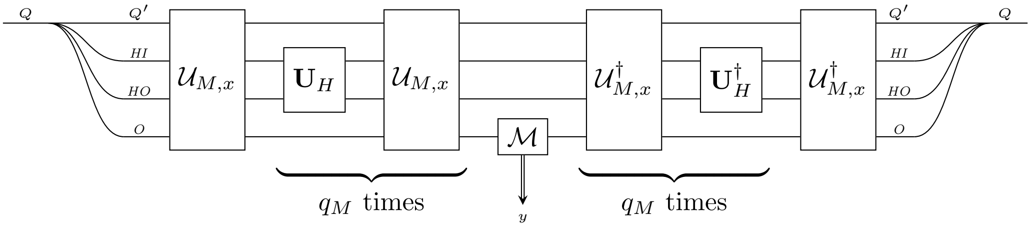
where all integers are encoded in unary. (This ensure that algorithms that run in polynomial-time in the length of $\mathbf{shape}_C$ will run in polynomial-time in those integers, too.) We call $\mathbf{shape}_C$

the *shape* of C. The shape contains all information that refers to communication performed by C (i.e., queries to oracles, input/output), but it does not contain information about the actual computation ($\mathcal{U}_C$ or the size ℓ_C^{state} of the state).

Projective measurement circuits (extractability only). A *projective measurement circuit* M consists of:

- An integer $\ell_M^{outcome} \geq 0$. The length of the outcome of the measurement.
- An integer $\ell_M^{quantum} \geq 0$. Length of the quantum state that is to be measured.
- An integer $\ell_M^{input} \geq 0$. The length of the classical input of the measurement. (I.e., a classical parametrization of the measurement.)
- An integer q_M. This indicates how many oracle queries M performs.
- Integers $\ell_M^{ora,in}, \ell_M^{ora,out} \geq 0$. These describe the number of input/output bits of the function H that is queried by M.
- A family of unitaries $\mathcal{U}_{M,x}$ on $\ell_M^{quantum}$ qubits, parametrized by $x \in \{0,1\}^{\ell_M^{input}}$. This is the computation performed by M between oracle queries, given classical input x.
- We assume that $\ell_M^{quantum} \geq \ell_M^{outcome} + \ell_M^{ora,in} + \ell_M^{ora,out}$.

For any $H : \{0,1\}^{\ell_M^{ora,in}} \to \{0,1\}^{\ell_M^{ora,out}}$ and $x \in \{0,1\}^{\ell_M^{input}}$, M induces the projective measurement M_x^H on $\ell_M^{quantum}$ qubits given by the following circuit:



Here $\mathcal{M}$ is a measurement in the computational basis. Q is an $\ell_M^{quantum}$ qubit register. HI, HO, O are $\ell_M^{ora,in}, \ell_M^{ora,out}, \ell_M^{outcome}$ qubit registers, respectively. Q' is a $(\ell_M^{quantum} - \ell_M^{ora,in} - \ell_M^{ora,out} - \ell_M^{outcome})$ qubit register.

We denote this measurement with M_x^H and write $y \leftarrow M_x^H(Q)$ to denote an application of the measurement on register Q and with outcome y.

Oracle algorithms. An oracle algorithm A consists of:

- t – the number of oracles that A expects.
- T_A – a probabilistic Turing machine with the special states *query* and *quantum*, as well as a special *oracle tape*.

For concrete oracles $\mathcal{O}_1, \dots, \mathcal{O}_t$ with state spaces $\mathcal{H}_i$, and for quantum registers R_i with spaces $\mathcal{H}_i$ (containing the internal states of the oracles) and an quantum register Q (assumed to consist of a finite number of qubits) and a classical value y, we write $x \leftarrow A^{\mathcal{O}_1(R_1),\dots,\mathcal{O}_t(R_t)}(y, Q)$ to denote the following process:

- Let ℓ_Q denote the number of qubits in Q.
- The Turing machine is executed with classical input (y, ℓ_Q) (according to the usual execution semantics of probabilistic Turing machines).
- When the Turing machine reaches the state *query*, then the content of the oracle tape is interpreted as an index $j \in \{1, \dots, t\}$. Let n_j be the number of qubits of the input/output space of $\mathcal{O}_j$. Then $\mathcal{E}_{\mathcal{O}_j}$ is applied to Q', R_j where Q' denotes the first n_j qubits of Q. Then the Turing machine continues execution (the oracle tape is not modified)

- When the Turing machine reaches the state *quantum*, then the content of the oracle tape is parsed as one of the following:
  - ($\mathbf{gate}, G, n_1, \dots, n_m$) where G is an m-qubit gate from some fixed finite universal set of unitary gates. Then G is applied to qubits $n_1, \dots, n_m$ of Q.
  - ($\mathbf{measure}, n$). Then the n-th qubit of Q is measured in the computational basis, and the outcome is written on the oracle tape.
  - ($\mathbf{destroy}$). Then the last qubit of Q (i.e., the qubit with the highest index) is destroyed (traced out).
  - ($\mathbf{create}$). Then one additional qubit will be created in Q (and given index $n+1$ if Q had n qubits before).

  Then the Turing machine continues execution.
- When a final state is reached, the content of the output tape is assigned to y.

Note that in this definition, the Turing machine T_A does not directly get an answer when performing an oracle query. However, it can, e.g., measure the answer or perform quantum computations on the answer by entering the *quantum* state.

Oracle access to pure oracle circuits (extractability only). When studying the rewinding of adversaries that have access to oracles, we will construct algorithms E that have oracle access to the adversary A who is in turn a pure oracle circuit that accesses further oracles. And E will in turn implement those oracles for A. To model this formally, we have to give E oracle access to the unitary $\mathcal{U}_A$ that describes the computation performed by A. Then A can repeatedly invoke $\mathcal{U}_A$ to simulate the computations performed by A, and in between answer A's oracle queries. However, given only $\mathcal{U}_A$, E cannot know which of its oracles A is currently calling. So E will need to get the execution schedule $\mathbf{op}_A$ of A as classical input (as well as the rest of the information provided in $\mathbf{shape}_A$). Furthermore, we want E to be able not only to invoke $\mathcal{U}_A$, but also $\mathcal{U}_A^\dagger$. And finally, we want E to be able to perform a controlled-$\mathcal{U}_A$ operation (i.e., maintain a superposition between invoking A and not invoking A). Thus, given $\mathcal{U}_A$, we define the unitary $\mathcal{U}_A^{\mathbf{rewind}}$ operating on registers $R, (I_j)_j, S_A$. Here R is a 2-qubit register (used for controlling whether $\mathcal{U}_A$ is applied forward, backward, or not at all), and $(I_j)_j, S_A$ are as in the description of pure oracle circuits above. $\mathcal{U}_A^{\mathbf{rewind}}$ is then the following unitary:

$$\mathcal{U}_A^{\mathbf{rewind}} : \begin{cases} |00\rangle_R \otimes |\Psi\rangle_Q \mapsto |00\rangle \otimes |\Psi\rangle, \\ |01\rangle_R \otimes |\Psi\rangle_Q \mapsto |01\rangle \otimes |\Psi\rangle, \\ |10\rangle_R \otimes |\Psi\rangle_Q \mapsto |11\rangle \otimes \mathcal{U}_A|\Psi\rangle, \\ |11\rangle_R \otimes |\Psi\rangle_Q \mapsto |10\rangle \otimes \mathcal{U}_A^\dagger|\Psi\rangle \end{cases}$$

where Q stands for the registers $(I_j)_j, S_A$ together. That is, the first qubit in R controls whether $\mathcal{U}_A$ is to be applied at all, and the second qubit in R controls whether $\mathcal{U}_A$ is supposed to be applied forward or backward. (For convenience, we flip the second qubit in the last two lines. This makes $\mathcal{U}_A^{\mathbf{rewind}}$ self-inverse.)

We define the oracle $A^{\mathbf{rew}}$: Its state space is the space of register S_A (i.e., ℓ_A^{state} qubits), and its input/output space is $R, (I_j)_j$ (i.e., $2 + \sum_j \ell_{C,j}^{oracle}$ qubits). The quantum operation of $A^{\mathbf{rew}}$ is $\mathcal{E}_{A^{\mathbf{rew}}} : \rho \mapsto \mathcal{U}_A^{\mathbf{rewind}} \rho (\mathcal{U}_A^{\mathbf{rewind}})^\dagger$. That is, $A^{\mathbf{rew}}$ gives access to the unitary $\mathcal{U}_A^{\mathbf{rewind}}$ defined above. (Thus the algorithm invoking $A^{\mathbf{rew}}$ has access to $R, (I_j)_j$, but not to S_C. The invoking algorithm can control whether $\mathcal{U}_A$ or its inverse is invoked, and can access the inputs and supply the outputs of the oracles queried by A, but it cannot access the internal state of A.)

Then, to give E oracle access to the oracle circuit A where A has initial state in register S_A, we invoke E as:

$$x \leftarrow E^{A^{\mathbf{rew}}(S_A)}(\mathbf{shape}_A).$$

Note that E provides the responses to the oracle queries performed by A. It is important that E knows $\mathbf{shape}_A$, otherwise it would not know, e.g., how many times to invoke $A^{\mathbf{rew}}$, nor which oracle to simulate after which invocation, or what the sizes of the registers I_j used by $A^{\mathbf{rew}}$ are.

Oracle access to projective measurement circuits (extractability only). Similarly, we need to give our algorithms oracle access to projective measurement circuits. However, there is a crucial difference to the case of oracle access to a pure oracle circuit. Namely, a projective measurement circuit M^H is supposed to implement a projective measurement. If we were to define oracle access to M analogously to oracle access to pure oracle circuits, we would give the invoking algorithm E access to $\mathcal{U}_{M,x}$, and leave it to E to implement the oracle H for M whenever M makes an oracle query. However, if we do this, E can invoke M^H in ways that do not guarantee that M^H implements a projective measurement. (E.g., if the function H changes between different invocations of H by M.) To avoid this, we take a different approach: E is given oracle access to M^H as a whole. (I.e., M^H will be executed within a single oracle query.) And E does not simulate the oracle H for M, instead the oracle M^H comes with H "built-in". This would lead to the following tentative definition: The oracle corresponding to M^H is given by the unitary

$$U_M^{\mathbf{tentative},H} : |x\rangle \otimes |y'\rangle \otimes |\Psi\rangle \mapsto \sum_y |x\rangle \otimes |y' \oplus y\rangle \otimes M_x^{H,y}|\Psi\rangle$$

where $M_x^{H,y}$ is the projector corresponding to outcome $y \in \{0,1\}^{\ell_M^{outcome}}$ of M_x^H. Intuitively, this unitary measures the third register using M_x^H and XORs the outcome onto the second register.

However, we will later need to give the invoking algorithm E the possibility to program H. That is, instead of giving H to the projective measurement circuit M, E will have to give $H(ass)$ to M. Here ass is an assignment-list, and $H(ass)$ is H updated according to ass (see Section 2).

So we use the following definition: Given an integer $\ell \geq 0$, let X be an ℓ_M^{input} qubit register, let Y be an $\ell_M^{outcome}$ qubit register, let T be a register large enough to contain the encoding of an assignment-list ass of length ℓ. Define the unitary

$$U_M^{\mathbf{oracle},H,\ell} : |x\rangle_X \otimes |ass\rangle_T \otimes |y'\rangle_Y \otimes |\Psi\rangle_Q \mapsto \sum_y |x\rangle \otimes |ass\rangle \otimes |y' \oplus y\rangle \otimes M_x^{H(ass),y}|\Psi\rangle$$

where $M_x^{H(ass),y}$ is the projector corresponding to outcome $y \in \{0,1\}^{\ell_M^{outcome}}$ of $M_x^{H(ass)}$, and $H(ass)$ is H updated using assignment-list ass (see Section 2). Intuitively, this unitary measures the register Q using $M_x^{H(ass)}$ (where x comes from register X and ass from register T) and XORs the outcome onto register Y.

We then define the oracle $M^{\mathbf{oracle},H,\ell}$: Its state space is the space of register Q, and its input/output space is X, T, Y. The quantum operation of $A^{\mathbf{rew}}$ is $\mathcal{E}_{A^{\mathbf{rew}}} : \rho \mapsto U_M^{\mathbf{oracle},H,\ell} \rho (U_M^{\mathbf{oracle},H,\ell})^\dagger$.

Note that we have to explicitly specify an upper bound on the length ℓ of the assignment-list ass since according to our definition of oracles, an oracle has fixed length input/output.

Two things are worth noting: In the definition of $A^{\mathbf{rew}}$ (the oracle corresponding to the pure oracle circuit A) we explicitly encoded the possibility to control with another qubit whether the oracle is invoked or not. In the present case this is not necessary: $U_M^{\mathbf{oracle},H,\ell}$ is self-inverse, and when register Y contains $|+\rangle^{\otimes \ell_M^{outcome}}$, then $U_M^{\mathbf{oracle},H,\ell}$ acts as the identity.

Polynomial-time families (extractability only). We call a family C_η of pure oracle circuits (parametrized by an integer η) *polynomial-time* if there exist a deterministic polynomial-time Turing machine M such that $M(1^\eta) = (t, (\ell_{C,j}^{oracle})_j, \ell_C^{output}, \ell_C^{state}, \mathbf{op}C, desc)$ where $desc$ is a description of $\mathcal{U}_{C_\eta}$. In this context, a *description* of a unitary is an explicit description of a circuit D that implements $\mathcal{U}_{C_\eta}$, using only the CNOT, Toffoli, Hadamard, and phase gate (which is a universal set of gates), and arbitrarily many auxiliary qubits (which are assumed to be initialized with $|0\rangle$, and are required to be in state $|0\rangle$ after the execution of the circuit D).

We call a family M_η of projective measurement circuits *polynomial-time* if there exist deterministic polynomial-time Turing machines T_1, T_2 such that $T_1(1^\eta) = (\ell_M^{outcome}, \ell_M^{quantum}, \ell_M^{input}, q_M, \ell_M^{ora,in}, \ell_M^{ora,out})$, and $T_2(1^\eta, x)$ is a description of $\mathcal{U}_{M_\eta,x}$.

4 Auxiliary lemmas

Theorem 4 (Random oracle programming [Unr15]) *Let $\ell^{in}, \ell^{out} \geq 1$ be a integers. Let A_C be an algorithm, and A_0, A_2 be oracles algorithms, where A_0^H makes at most q_A queries to H, A_C is classical, and the output of A_C has collision-entropy at least k given A_C's initial state (which is classical). A_0, A_C, A_2 may share state.*

Then

$$\Big| \Pr[b = 1 : H \xleftarrow{\$} \mathrm{Fun}(\ell_\eta^{in}, \ell_\eta^{out}), A_0^H(), xcom \leftarrow A_C(), ch := H(xcom), b \leftarrow A_2^H(ch)]$$

$$- \Pr[b = 1 : H \xleftarrow{\$} \mathrm{Fun}(\ell_\eta^{in}, \ell_\eta^{out}), A_0^H(), xcom \leftarrow A_C(), ch \xleftarrow{\$} \{0,1\}^m, H(xcom) := ch, b \leftarrow A_2^H(ch)] \Big|$$

$$\leq (4 + \sqrt{2})\sqrt{q_A}\, 2^{-k/4}.$$

Lemma 5 *Let $H : \{0,1\}^n \to \{0,1\}^m$ be a uniformly random function. For any q-query algorithm A, it holds that $\Pr[H(x) = 0 : x \leftarrow A^H()] \leq 32 \cdot 2^{-m} \cdot (q+1)^2$.*

Proof. Let $p := \Pr[H(x) = 0 : x \leftarrow A^H()]$.

Let $f : \{0,1\}^n \to \{0,1\}$ be a random function such that $f(x) = 1$ with probability 2^{-m}. (And all $f(x)$ are independent.) And the algorithm $B^f()$ does the following: It picks a uniformly random function $G : \{0,1\}^n \to \{0,1\}^m \setminus \{0^m\}$. It defines1 $H'(x) := G(x)$ if $f(x) = 0$ and $H'(x) := 0^m$ otherwise. Then B^f executes $x \leftarrow A^{H'}()$ and returns x.

Notice that given oracle access to f, one can implement H' using two queries to f. Thus B makes $\leq 2q$ queries. Furthermore, if f is distributed as described above, then H' is uniformly distributed. Thus $x \leftarrow A^{H'}()$ returns x with $H'(x) = 0^m$ with probability p. Since $H'(x) = 0^m$ iff $f(x) = 1$, $x \leftarrow B^f()$ returns x with $f(x) = 1$ with probability p. [HRS16, Theorem 1] states that any q-query algorithm finds a 1-preimage in a function distributed like f with probability at most $8(2^{-m})(q+1)^2$. Since B makes $2q$ queries, it follows that $p \leq 8 \cdot 2^{-m} \cdot (2q+1)^2 \leq 32 \cdot 2^{-m} \cdot (q+1)^2$. □

5 Sigma protocols

In this paper, we will consider only proof systems for *fixed-length relations*. A fixed-length relation R_η is a family of relations on bitstrings such that:

For every η, there are values ℓ_η^x and ℓ_η^w such that $(x, w) \in R_\eta$ implies $|x| = \ell_\eta^x$ and $|w| = \ell_\eta^w$, and such that ℓ_η^x, ℓ_η^w can be computed in time polynomial in η. Given x, w, it can be decided in polynomial-time in η whether $(x, w) \in R_\eta$.

We now define sigma protocols and related concepts. The notions in this section are standard in the classical setting, and easy to adapt to the quantum setting. Note that the definitions are formulated without the random oracle, we only use the random oracle later for constructing non-interactive proofs out of sigma protocols.

A *sigma protocol* for a fixed-length relation R_η is a three-message proof system. It is described by the lengths $\ell^{com}_\eta, \ell^{ch}_\eta, \ell^{resp}_\eta$ of the "commitments", "challenges", and "responses" (those lengths may depend on η), by a quantum-polynomial-time[13] prover (P^1_Σ, P^2_Σ) and a deterministic polynomial-time verifier V_Σ. We will commonly denote statement and witness with x and w (with $(x,w) \in R$ in the honest case). The first message from the prover is $com \leftarrow P^1_\Sigma(1^\eta, x, w)$ and is called the *commitment* and satisfies $com \in \{0,1\}^{\ell^{com}}$, the uniformly random reply from the verifier is $ch \xleftarrow{\$} \{0,1\}^{\ell^{ch}}$ (called *challenge*), and the prover answers with a message $resp \leftarrow P^2_\Sigma(1^\eta, x, w, ch)$ (the *response*) that satisfies $resp \in \{0,1\}^{\ell^{resp}}$. We assume P^1_Σ, P^2_Σ to share classical or quantum state. Finally $V_\Sigma(1^\eta, x, com, ch, resp)$ outputs 1 if the verifier accepts, 0 otherwise.

Definition 6 (Properties of sigma protocols) *Let $(\ell^{com}_\eta, \ell^{ch}_\eta, \ell^{resp}_\eta, P^1_\Sigma, P^2_\Sigma, V_\Sigma)$ be a sigma protocol. We define:*

- ***Completeness:*** *For any quantum-polynomial-time algorithm A, there is a negligible μ such that for all η,*

$$\begin{aligned}\Pr[(x,w) \in R_\eta \ \wedge \ V_\Sigma(1^\eta, x, com, ch, resp) = 0 : (x,w) \leftarrow A(1^\eta),\\ com \leftarrow P^1_\Sigma(1^\eta, x, w),\ ch \xleftarrow{\$} \{0,1\}^{\ell^{ch}_\eta},\ resp \leftarrow P^2_\Sigma(1^\eta, x, w, ch)] \leq \mu(\eta).\end{aligned}$$

- ***Perfect special soundness:*** *There is a quantum-polynomial-time algorithm E_Σ such that for all $\eta, x, com, ch, resp, ch', resp'$ with $ch \neq ch'$ and $V_\Sigma(1^\eta, x, com, ch, resp) = V_\Sigma(1^\eta, x, com, ch', resp') = 1$, we have that*

$$\Pr[(x,w) \in R_\eta : w \leftarrow E_\Sigma(1^\eta, x, com, ch, resp, ch', resp')] = 1.$$

- ***Special soundness:*** *There is a quantum-polynomial-time algorithm E_Σ such that for any quantum-polynomial-time A, the following is negligible:*

$$\begin{aligned}&\Pr[(x,w) \notin R_\eta \wedge ch \neq ch' \wedge ok = ok' = 1 : (x, com, ch, resp, ch', resp') \leftarrow A(1^\eta),\\ &\quad ok \leftarrow V_\Sigma(1^\eta, x, com, ch, resp), ok' \leftarrow V_\Sigma(1^\eta, x, com, ch', resp'),\\ &\quad w \leftarrow E_\Sigma(1^\eta, x, com, ch, resp, ch', resp')].\end{aligned}$$

- ***Honest-verifier zero-knowledge (HVZK):*** *There is a quantum-polynomial-time algorithm S_Σ (the simulator) such that for any stateful quantum-polynomial-time algorithm A there is a negligible μ such that for all η and $(x,w) \in R_\eta$,*

$$\begin{aligned}&\Big|\Pr[b=1 : (x,w) \leftarrow A(1^\eta), com \leftarrow P^1_\Sigma(1^\eta, x, w), ch \xleftarrow{\$} \{0,1\}^{\ell^{ch}_\eta}, resp \leftarrow P^2_\Sigma(1^\eta, x, w, ch),\\ &\qquad b \leftarrow A(1^\eta, com, ch, resp)]\\ &- \Pr[b=1 : (x,w) \leftarrow A(1^\eta), (com, ch, resp) \leftarrow S(1^\eta, x), b \leftarrow A(1^\eta, com, ch, resp)]\Big| \leq \mu(\eta).\end{aligned}$$

- ***Perfectly unique responses:*** *There exist no values $\eta, x, com, ch, resp, resp'$ with $resp \neq resp'$ and $V_\Sigma(1^\eta, x, com, ch, resp) = 1$ and $V_\Sigma(1^\eta, x, com, ch', resp') = 1$.*
- ***Unique responses:*** *For any quantum-polynomial-time A, the following is negligible:*

$$\begin{aligned}\Pr\big[resp \neq resp' \wedge V_\Sigma(1^\eta, x, com, ch, resp) = 1 \wedge V_\Sigma(1^\eta, x, com, ch', resp') = 1 :\\ (x, com, ch, resp, resp') \leftarrow A(1^\eta)\big].\end{aligned}$$

[13] Typically, P^1_Σ and P^2_Σ will be classical, but we do not require this since our results also hold for quantum P^1_Σ, P^2_Σ. But the inputs and outputs of P^1_Σ, P^2_Σ are classical.

- ***Unpredictable commitments:*** *The commitment has superlogarithmic collision-entropy. In other words, there is a negligible μ such that for all η and $(x,w) \in R_\eta$,*

$$\Pr[com_1 = com_2 : com_1 \leftarrow P^1_\Sigma(1^\eta, x, w),\ com_2 \leftarrow P^1_\Sigma(1^\eta, x, w)] \leq \mu(\eta).$$

Note: the "unpredictable commitments" property is non-standard, but satisfied by all sigma-protocols we are aware of. However, any sigma-protocol without unpredictable commitments can be transformed into one with unpredictable commitments by appending superlogarithmically many random bits to the commitment (that are then ignored by the verifier).

6 Non-interactive proof systems (Definitions)

In the following, let H always denote a function $\{0,1\}^{\ell^{in}_\eta} \to \{0,1\}^{\ell^{out}_\eta}$ where $\ell^{in}_\eta, \ell^{out}_\eta$ may depend on the security parameter η. Let $\mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta)$ denote the set of all such functions.

A non-interactive proof system (P,V) for a relation R_η consists of a quantum-polynomial-time algorithm P and a deterministic polynomial-time algorithm V, both taking an oracle $H \in \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta)$. $\pi \leftarrow P^H(1^\eta, x, w)$ is expected to output a proof π for the statement x using witness w. We require that $|\pi| = \ell^\pi_\eta$ for some length ℓ^π_η. (I.e., the length of a proof π depends only on the security parameter.) And $ok \leftarrow V^H(1^\eta, x, \pi)$ is supposed to return $ok = 1$ if the proof π is valid for the statement x. Formally, we define:

Definition 7 (Completeness) *(P,V) has* completeness *for a fixed-length relation R_η iff for any polynomial-time oracle algorithm A there is a negligible μ such that for all η,*

$$\Pr[(x,w) \in R_\eta \ \wedge\ V^H(1^\eta, x, \pi) = 0 : H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta),\\ (x,w) \leftarrow A^H(1^\eta),\ \pi \leftarrow P^H(1^\eta, x, w)] \leq \mu(\eta).$$

For the following definition, a *simulator* is a classical stateful algorithm S. Upon invocation, $S(1^\eta, x)$ returns a proof π. Additionally, S may reprogram the random oracle. That is, S may choose an assignment-list ass, and H will then be replaced by $H(ass)$.

Definition 8 (Zero-knowledge) *Given a simulator S, the oracle $S'(x,w)$ runs $S(1^\eta, x)$ and returns the latter's output. Given a prover P, the oracle $P'(x,w)$ runs $P(1^\eta, x, w)$ and returns the latter's output.*

A non-interactive proof system (P,V) is zero-knowledge *iff there is a quantum-polynomial-time simulator S such that for every quantum-polynomial-time oracle algorithm A and a negligible μ such that for all η and all normalized density operators ρ,*

$$\Big| \Pr[b = 1 : H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta),\ b \leftarrow A^{H,P}(1^\eta, \rho)]\\ - \Pr[b = 1 : H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta),\ b \leftarrow A^{H,S'}(1^\eta, \rho)] \Big| \leq \mu(\eta). \tag{1}$$

Here we quantify only over A that never query $(x,w) \notin R$ from the P' or S'-oracle.

Definition 9 (Soundness) *A non-interactive proof system (P,V) is* sound *iff for any quantum-polynomial-time oracle algorithm A, there is a negligible function μ, such that for all η and all normalized density operators ρ,*

$$\Pr[ok_V = 1 \wedge x \notin L_R : (x,\pi) \leftarrow A^H(1^\eta, \rho),\ ok_V \leftarrow V^H(1^\eta, x, \pi)] \leq \mu(\eta).$$

Here $L_R := \{x : \exists w.(x,w) \in R\}$.

In some applications, soundness as defined above is not sufficient. Namely, consider a security proof that goes along the following lines: We start with a game in which the adversary interacts with an honest prover. We replace the honest prover by a simulator. From the zero-knowledge property it follows that this leads to an indistinguishable game. And then we try to use soundness to show that the adversary in the new game cannot prove certain statements.

The last proof step will fail: soundness guarantees nothing when the adversary interacts with a simulator that constructs fake proofs. Namely, it could be that the adversary can take a fake proof for some statement and changes it into a fake proof for another statement of its choosing. (Technically, soundness cannot be used because the simulator programs the random oracle, and Definition 9 provides no guarantees if the random oracle is modified.)

An example where this problem occurs is the proof of Theorem 27 below (unforgeability of Fiat-Shamir signatures).

To avoid these problems, we adapt the definition of simulation-soundness [Sah99] to the quantum setting. Roughly speaking, simulation-soundness requires that the adversary cannot produce wrong proofs π, even if it has access to a simulator that it can use to produce arbitrary fake proofs. (Of course, it does not count if the adversary simply outputs one of the fake proofs it got from the simulator. But we require that the adversary cannot produce any other wrong proofs.)

Definition 10 (Simulation-soundness) *A non-interactive proof system (P, V) is* simulation-sound *with respect to the simulator S iff for any quantum-polynomial-time oracle algorithm A, there is a negligible function μ, such that for all η and all normalized density operators ρ,*

$$\begin{aligned}&\Pr[ok_V = 1 \land x \notin L_R \land (x,\pi) \notin \textsf{S-queries} : \\ &\quad (x,\pi) \leftarrow A^{H,S''}(1^\eta,\rho),\ ok_V \leftarrow V^{H_{\mathit{final}}}(1^\eta,x,\pi)] \le \mu(\eta).\end{aligned} \tag{2}$$

Here the oracle $S''(x)$ invokes $S(1^\eta, x)$. And H_{final} refers to the value of the random oracle H at the end of the execution (recall that invocations of S may change H). S-queries *is a list containing all queries made to S'' by A, as pairs of input/output. (Note that the input and output of S'' are classical, so such a list is well-defined.)*

When considering simulation-sound zero-knowledge proof systems, we will always implicitly assume that the same simulator is used for the simulation-sound extractability and for the zero-knowledge property.

6.1 Extractability

The basic idea behind extractability is to model the idea that whenever an adversary manages to produce a valid proof for a statement x, it also knows a corresponding witness w with $(x, w) \in R$. This is typically formalized (see, e.g., [BG93]) by requiring that an extractor (with blackbox access to the adversary) can compute a valid witness for x with high probability when the adversary can produce a proof.

A very first attempt to formalize this in the quantum setting with random oracle would lead to, roughly, the following definition:

Definition 11 (Extractability, too weak, informal) *A non-interactive proof system (P, V) is* extractable *iff there is a quantum polynomial-time oracle algorithm E and a constant $d > 0$, such that for any polynomial-time adversary A there is a polynomial $p > 0$ such that for all initial states in the register S_A:*

$$\Pr\big[(x,w) \in R : \mathbf{Extract}\big] \ge \frac{1}{p}\Pr\big[ok_V = 1 : \mathbf{Prove}\big]^d - \textit{negligible}.$$

Here **Prove** *is the following game:*

$$(x, \pi) \leftarrow A^H(S_A), \quad ok_V \leftarrow V^H(x, \pi)$$

And **Extract** *is the following game:*

$$(x, w) \leftarrow E^{A(S_A),H}()$$

Basically, this requires that when the adversary (with oracle access to the random oracle H, and initial state in S_A) produces a valid proof with probability q, then the extractor (with blackbox access to A) manages to compute a valid witness with probability $q/poly - negligible$. This ensures that the success probability of the extractor is non-negligible whenever that of A is. However, this definition has one very big problem: There is no guarantee that the statement x chosen by the extractor E is the same as the statement x chosen by A. For example, the definition would be satisfied if the extractor E always outputs some fixed pair $(x_0, w_0) \in R$, regardless of what the adversary does. So, obviously, we need to somehow enforece that the x returned by E is "the same" as the x returned by A. One way is the following definition sketch where x is fixed (via an all-quantifier) and given to both A and E:

Definition 12 (Extractability, non-adaptive, too weak, informal) *A non-interactive proof system* (P, V) *is* extractable *iff there is a quantum polynomial-time oracle algorithm* E *and a constant* $d > 0$*, such that for any polynomial-time adversary* A *there is a polynomial* $p > 0$ *such that for all* x *and all initial states in the register* S_A*:*

$$\Pr\big[(x, w) \in R : \mathbf{Extract}\big] \geq \frac{1}{p} \Pr\big[ok_V = 1 : \mathbf{Prove}\big]^d - negligible.$$

Here **Prove** *is the following game:*

$$\cancel{x,}\, \pi \leftarrow A^H(x, S_A), \quad ok_V \leftarrow V^H(x, \pi)$$

And **Extract** *is the following game:*

$$\cancel{x,}\, w \leftarrow E^{A(S_A),H}(x)$$

This definition indeed enforces that A and E use the same x. However, it has one big problem:

- It is non-adaptive. That is, it does not capture attacks in which the adversary produces the statement x depending on, e.g., the results of random oracle queries. Thus, this definition would only be useful in a context where the statement is used without any adversarial influence. For example, when we try to use Fiat-Shamir to construct signature schemes (Section 8), the statement x will contain the public key (which is indeed not under adversarial influence) and the message m that is to be signed. In case of a chosen message attack, m would be chosen or influenced by the adversary, and therefore could, e.g., contain outputs of the random oracle. Thus, also in the case of constructing signatures, Definition 12 is not suitable.
- In addition, we have a phenomenon unique to the quantum setting. Namely, when we run the extractor with oracle access to the adversary A, this indirectly affects the register S_A that contains the state of A. This means that we cannot simultaneously get access to the final state of the adversary, and to the extracted x, w. This is very different from the classical setting: in the classical setting, we can simply copy the adversary's initial state before running the extractor (and get the final state by rerunning the adversary). For example, in our security proof for signatures (Theorem 28), we make use of the fact that the adversary's state is available simultaneously with the extracted (x, w).

One solution to both problems is to require "online-extractability" [Fis05, Unr15]. Online-extractability requires that the extractor can run side-by-side with the adversary, without in any way disturbing (or even accessing) the adversary's state. (For example, in the classical setting this can be achieved by using an extractor that extracts given merely a list of oracle-queries [Fis05], and in the quantum setting it can be achieved by using an extractor that sets up a fake random oracle with a trapdoor, and then can extract a witness from any valid proof by using that trapdoor [Unr15].) Since an online-extractor does not influence the adversary, both problems are easily: Since extractor and adversary are part of the same execution, it makes sense to require that the extractor uses the same x as was chosen by the adversary. And the adversary's state is not modified by the extractor since the extractor never interacts with the adversary (except for passively receiving x and the proof).

Unfortunately, for proof systems such as Fiat-Shamir, constructing online-extractors does not seem possible, even in the classical case. This is because to extract a witness from the underlying sigma-protocol (using the special soundness property), we need to extract *two* valid sigma-protocol interactions from the adversary. Although we have no proof of this, it seems unlikely that *two* such interactions can be extracted without rewinding the adversary (or doing something comparable to it). In fact, the difficulty of constructing online-extractors for Fiat-Shamir was what motivated the construction of the classical online-extractable non-interactive proof system from [Fis05].

One alternative solution to the two problems would be the following definition. In this definition, we require that there is an extractor that produces a faithful simulation of the uninfluenced execution of the adversary A, while at the same time extracting a witness. More precisely, we require that when running the extractor (with oracle access to an adversary A running with an internal state register S_A), the extractor outputs x, w, π such that: x, π, S_A are indistinguishable from the results of a normal execution of the adversary (without extractor). And whenever π is a valid proof for x, then w is a valid witness for x.

The following definition makes this more precise:

Definition 13 (Extractability, too strong, informal) *A non-interactive proof system* (P, V) *is* extractable *iff there is an expected-polynomial-time quantum oracle algorithm E such that for any polynomial-time adversary A and all initial states in the register S_A:*

$$\Pr\big[(x, w) \notin R \wedge ok_V = 1 : \textbf{Extract}\big] \text{ is negligible}$$

and

$$(x, \pi, S_A) \text{ after } \textbf{Prove} \quad \text{is indistinguishable from} \quad (x, \pi, S_A) \text{ after } \textbf{Extract}$$

Here **Prove** *is the following game:*

$$(x, \pi) \leftarrow A^H(S_A), \quad ok_V \leftarrow V^H(x, \pi)$$

And **Extract** *is the following game:*

$$(x, w, \pi) \leftarrow E^{A(S_A), H}(), \quad ok_V \leftarrow V^H(x, \pi)$$

We believe that having a proof system that satisfies this definition would be ideal. And indeed, it is easy to see that online-extractable proof systems (satisfying the definition from [Unr15]) satisfy this definition. However, we believe that in the quantum setting, this definition is too much to hope for when it comes to the security of Fiat-Shamir. As discussed above, it seems that the only way to extract a witness from a Fiat-Shamir proof is to rewind the adversary. Even in the case of extraction from direct executions of sigma-protocols [Unr12] (which can be seen as a special case of Fiat-Shamir where the adversary can make only a single classical query to the random oracle) we do not know how to perform such a rewinding without at least partially disturbing the state. Thus, although we have no proof that Definition 13 cannot be satisfied by Fiat-Shamir, it would at least require major breakthroughs in quantum rewinding techniques.

To avoid running into this problem, we weaken the Definition 13 somewhat.

In many situations, we would use the extractability property as in Definition 13 in roughly the following way: We start out with an adversary that produces some statement x and proves knowledge of a corresponding witness w in a situation S where the adversary cannot produce such a witness (e.g., x contains the public key of the signature scheme, and w would then have to be the secret key that is unguessable by assumption). Then we apply extractability and get an extractor that produces both x and w. Since the output of the extractor (without w) is indistinguishable from the output of the adversary, we conclude that the extractor is also in situation S. (Of course, whether this is a valid conclusion depends very much on how S is defined in the specific proof.) Finally, we conclude that an extractor that find x, w in situation S is a contradiction to our assumptions (e.g., the extractor would compute the secret key from the public key).

In such a proof, we will usually not need that the extractor's output is indistinguishable from the adversary's output. Instead, it is sufficient to be able to conclude that, if the adversary produces a valid proof while being in situation S with non-negligible probability, then the extractor produces a valid witness while being in situation S with (possibly smaller) non-negligible probability. In the classical setting, a "situation" might be described by a predicate on the adversaries output and final state. (E.g., the predicate could be "the first part of x is the public key stored in the adversary's state".) Thus, we would require a definition roughly as follows:

For any predicate P and adversary A we have: If with probability p, the adversary produces a valid proof π for a statement x and has final state S_A, such that (x, π, S_A) satisfies the predicate P, then the extractor (with black-box access to A and P) will output x, π, w such that $(x, w) \in R$, and (x, π, S_A) satisfies P (where S_A is the final state of the black-box adversary A).[14] In the quantum setting, S_A would be a quantum state, so we cannot define a classical predicate P that is applied to S_A. Instead, in the quantum setting we use a projective measurement Π on x, w, S_A instead to define which final states are acceptable.

These ideas lead to the following definition:

Definition 14 (Extractability, informal) *A non-interactive proof system (P, V) is extractable iff there is a quantum polynomial-time oracle algorithm E and a constant $d > 0$, such that for any polynomial-time adversary A_η there exists a polynomial $\ell \geq 0$ such that for any polynomial-time measurement $\Pi^H_{x,\pi}$ (that may depend on the oracle H and on some values x, π), there exist a polynomial $p > 0$ such that for all initial states in the register S_A:*

$$\Pr\bigl[(x,w) \in R \land ok_A = 1 : \textbf{Extract}\bigr] \geq \frac{1}{p(\eta)} \Pr\bigl[ok_V = 1 \land ok_A = 1 : \textbf{Prove}\bigr]^d - \mu(\eta).$$

Here **Prove** *is the following game:*

$$(x, \pi) \leftarrow A^H(S_A), \quad ok_V \leftarrow V^H(x, \pi), \quad ok_A \leftarrow \Pi^H_{x,\pi}(S_A).$$

And **Extract** *is the following game:*

$$(x, w, \pi, ass) \leftarrow E^{A(S_A), \Pi(S_A), H}(), \quad ok_A \leftarrow \Pi^{H(ass)}_{x,\pi}(S_A).$$

where ass is an assignment-list and $H(ass)$ is the result of assigning ass in H (see Section 2).

[14] We give the extractor access to the predicate P. One can also conceive a stronger definition where the extractor does not get access to P. However, we do not know of a situation where this stronger definition would be helpful, and we believe that in the quantum case, giving the extractor access to P might help in constructing an extractor. (For example, some amplitude amplification technique might need to know what property the final state is supposed to have.)

In this definition, the game **Prove** represents a normal execution of the adversary A^H. The Boolean ok_V represents whether the proof π is valid, and ok_A represents whether the projective measurement $\Pi^H_{x,\pi}$ on the state of the adversary succeeded. The projective measurement is parametrized by x, π, which means that is can check conditions that depend on x and π. (It is even possible that $\Pi^H_{x,\pi}$ is just a predicate on x, π. To achieve this, we let $\Pi^H_{x,\pi}$ be a measurement that always succeeds for certain x, π, and always fails for other x, π.) In addition, we allow Π to depend on the random oracle H. Namely we assume Π to be implemented by a circuit making a polynomial-number of queries to H. In the game **Extract**, the extractor tries to mimic the results from the game **Prove**, while additionally trying to extract the witness w. The extractor E has black-box access to A and the family $\Pi^H_{x,w}$. Both black-box oracles operate on the same register S_A (to which E has no direct access). E can provide the original or a reprogrammed oracle H to the black-box A and Π. E can also provide the values x, w used by the black-box Π. (The details of these oracle access mechanisms are formalized in Section 3.) Finally, when the extractor returns x, π, w, we check whether $(x, w) \in R$ and whether the final state S_A of the adversary satisfies the measurement Π (represented by the Boolean ok_A). This final measurement Π gets the values x, π produced by the extractor, and it gets oracle access to $H(ass)$ instead of H. $H(ass)$ is the oracle H reprogrammed at a list ass of locations chosen by E. (Because E may have to reprogram the oracle H during its extraction process.)

We stress that we also do not have a proof that this definition is satisfied by Fiat-Shamir. However, it seems more likely that we can show that Fiat-Shamir satisfies Definition 14 than Definition 13. Namely, Definition 13 had the problem that it requires the extractor to perform its extraction without disturbing the state S_A of the adversary in the least. (We require computational indistinguishability.) This seems to make rewinding very difficult. In contrast, in Definition 14, the extractor may disturb S_A to some degree, as long as the extractor can make sure that there is a small probability that S_A will still satisfy Π.

At the same time, the definition seems still strong enough for non-trivial proofs such as the security of signatures (when combined with the concept of simulation-soundness, see Theorem 28).

Note that many variations of this definition are possible. For example, we could weaken it if we only quantify over $\Pi_{x,\pi}$ that measure in the computational basis (representing a classical predicate), or by not giving $\Pi_{x,\pi}$ access to H (i.e., the predicate that is checked does not depend on the random oracle). At least our proof of unforgeability of signatures (Theorem 28 still works with this weakened definition.

We now state Definition 14 precisely. Definition 15 is essentially the same definition as Definition 14, except that we use precise notation, and do not elide any arguments to the various algorithms (e.g., $\mathbf{shape}_{A_\eta}$ which informs E about the number and kind of oracle queries performed by A, see Section 3).

Definition 15 (Extractability) *A non-interactive proof system (P, V) is* extractable *iff there is a quantum polynomial-time oracle algorithm E and a constant $d > 0$, such that for any polynomial-time family of pure oracle circuits A_η (with $\ell^{output}_{A_\eta} = \ell^x_\eta + \ell^\pi_\eta$) there exists a polynomial $\ell \geq 0$ such that for any polynomial-time family of projective measurement circuits Π_η, there exist a polynomial $p > 0$ and a negligible function μ such that for all η and all $\ell^{state}_{A_\eta}$-qubit density operators ρ, we have that:*

$$\Pr\big[(x, w) \in R \wedge ok_A = 1 : \mathbf{Extract}\big] \geq \frac{1}{p(\eta)} \Pr\big[ok_V = 1 \wedge ok_A = 1 : \mathbf{Prove}\big]^d - \mu(\eta). \quad (3)$$

Here **Prove** *is the following game:*

$$\begin{aligned}
&H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta),\\
&S_A \leftarrow \rho,\\
&x\|\pi \leftarrow A^H_\eta(S_A),\\
&ok_V \leftarrow V^H(1^\eta, x, \pi),\\
&ok_A \leftarrow \Pi^H_{\eta,x\|\pi}(S_A).
\end{aligned}$$

Here $|x| = \ell^x_\eta, |\pi| = \ell^\pi_\eta$.

And **Extract** *is the following game:*

$$\begin{aligned}
&H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta),\\
&S_A \leftarrow \rho,\\
&(x, w, \pi, ass) \leftarrow E^{A^{\mathbf{rew}}_\eta(S_A), \Pi^{\mathbf{oracle},H,\ell(\eta)}_\eta(S_A), H}(1^\eta, \ell(\eta), \mathbf{shape}_{A_\eta}),\\
&ok_A \leftarrow \Pi^{H(ass)}_{\eta,x\|\pi}(S_A).
\end{aligned}$$

where ass *is an assignment-list and* $H(ass)$ *is the result of assigning* ass *in* H *(see Section 2).*

Finally, we would like to stress that showing that Fiat-Shamir satisfies *any* of the definitions in this section (except the trivial Definition 11) would constitute a major step forward in understanding quantum Fiat-Shamir. We leave it as an open problem to prove extractability of Fiat-Shamir with respect to any of those definitions.

6.2 Simulation-sound extractability

As described above after Definition 10 (simulation-soundness), there are cases where in a proof one needs to use the soundness property in a situation where the adversary has access to a simulator that produces fake proofs. Analogously, we may also need the extractability property in a situation where the adversary has access to a simulator. (For example in the proof of Theorem 28 (unforgeability of Fiat-Shamir signatures) below.) Otherwise, it would be possible for the adversary to receive a proof for a statement x from some party which it does not know the witness of (in the signature setting, an honestly signed signature would constitute such a proof), and transform it into a new proof for a related statement x' without knowing the witness of the new statement x' (in the signature setting, that new proof might constitute a forged signature for a different message).

We highlight the differences to Definition 15 (extractability) in blue.

Definition 16 (Simulation-sound extractability) *A non-interactive proof system* (P, V) *is* simulation-sound extractable *with respect to the simulator* S *iff there is a quantum polynomial-time oracle algorithm* E *and a constant* $d > 0$*, such that for any polynomial-time family of pure oracle circuits* A_η *(with* $\ell^{output}_{A_\eta} = \ell^x_\eta + \ell^\pi_\eta$*) there is a polynomial* $\ell \geq 0$ *such that for any polynomial-time family of projective measurement circuits* Π_η*, there exists a polynomial* $p > 0$ *and a negligible function* μ *such that for all* η *and all* $\ell^{state}_{A_\eta}$*-qubit density operators* ρ*, we have that:*

$$\begin{aligned}
\Pr\bigl[(x, w) \in R \wedge ok_A = 1 : \mathbf{Extract}\bigr]&\\
\geq \frac{1}{p(\eta)} \Pr\bigl[ok_V = 1 \wedge ok_A = 1 \wedge (x, \pi) \notin \mathsf{S\text{-}queries} : \mathbf{Prove}\bigr]^d - \mu(\eta).& \quad (4)
\end{aligned}$$

Here **Prove** *is the following game:*

$$\begin{aligned}
&H \xleftarrow{\$} \mathrm{Fun}(\ell_\eta^{in}, \ell_\eta^{out}),\\
&S_A \leftarrow \rho,\\
&x\|\pi \leftarrow A_\eta^{H,S''}(S_A),\\
&ok_V \leftarrow V^{H_{\mathit{final}}}(1^\eta, x, \pi),\\
&ok_A \leftarrow \Pi_{\eta,x\|\pi}^{H_{\mathit{final}}}(S_A).
\end{aligned}$$

Here the oracle $S''(x)$ invokes $S(1^\eta, x)$. And H_{final} refers to the value of the random oracle H at the end of the execution (recall that invocations of S may change H). S-queries *is a list containing all queries made to S'' by A, as pairs of input/output. (Note that the input and output of S'' are classical, so such a list is well-defined.) And $|x| = \ell_\eta^x, |\pi| = \ell_\eta^\pi$.*

And **Extract** *is the following game:*

$$\begin{aligned}
&H \xleftarrow{\$} \mathrm{Fun}(\ell_\eta^{in}, \ell_\eta^{out}),\\
&S_A \leftarrow \rho,\\
&(x, w, \pi, ass) \leftarrow E^{A_\eta^{\mathbf{rew}}(S_A), \Pi_\eta^{\mathbf{oracle}, H, \ell(\eta)}(S_A), H}(1^\eta, \ell(\eta), \mathbf{shape}_{A_\eta}),\\
&ok_A \leftarrow \Pi_{\eta,x\|\pi}^{H(ass)}(S_A).
\end{aligned}$$

where ass is an assignment-list and $H(ass)$ is the result of assigning ass in H (see Section 2).

When considering simulation-sound extractable zero-knowledge proof systems, we will always implicitly assume that the same simulator is used for the simulation-sound extractability and for the zero-knowledge property.

7 Fiat-Shamir

For the rest of this paper, fix a sigma-protocol $\Sigma = (\ell_\eta^{com}, \ell_\eta^{ch}, \ell_\eta^{resp}, P_\Sigma^1, P_\Sigma^2, V_\Sigma)$ for a fixed-length relation R_η. Let $H : \{0,1\}^{\ell_\eta^x + \ell_\eta^{com}} \to \{0,1\}^{\ell_\eta^{ch}}$ be a random oracle.

Definition 17 *The* Fiat-Shamir proof system (P_{FS}, V_{FS}) *consists of the algorithms P_{FS} and V_{FS} defined in Figure 1.*

7.1 Completeness

Lemma 18 *If Σ has completeness and unpredictable commitments, then Fiat-Shamir (P_{FS}, V_{FS}) has completeness.*

Interestingly, without unpredictable commitments, the lemma does not hold. Consider the following example sigma-protocol: Let $R_\eta := \{(x, w) : |x| = |w| = \eta\}$, $\ell^{com} := \ell^{ch} := \ell^{resp} := \eta$. Let $P_\Sigma^1(1^\eta, x, w)$ output $com := 0^\eta$. Let $P_\Sigma^2(1^\eta, x, w, ch)$ output $resp := ch$ if $ch \neq w$, and $resp := \overline{ch}$ else ($\overline{ch}$ is the bitwise negation of ch). Let $V_\Sigma(1^\eta, x, com, ch, resp) = 1$ iff $|x| = \eta$ and $ch = resp$. This sigma-protocol has all the properties from Definition 6 except unpredictable commitments. Yet (P_{FS}, V_{FS}) does not have completeness: A can chose $x := 0^\eta$ and $w := H(0^\eta\|0^\eta)$. For those choices of (x, w), $P_{FS}(x, w)$ will chose $com = 0^\eta$ and $ch = H(x\|com) = w$ and thus $resp = \overline{ch}$ and return $\pi = (com, \overline{ch})$. This proof will be rejected by V_{FS} with probability 1.

Proof of Lemma 18. Fix a polynomial-time oracle algorithm A. We need to show that $\Pr[win = 1 : \text{Game 1}]$ is negligible for the following game:

Game 1 (Completeness) *$H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta)$, $(x,w) \leftarrow A^H(1^\eta)$, $\pi \leftarrow P^H_{FS}(1^\eta, x, w)$, $ok_V \leftarrow V^H_{FS}(1^\eta, x, \pi)$, $win := ((x,w) \in R_\eta \wedge ok_V = 0)$.*

Let $P^{1,class}_\Sigma, P^{2,class}_\Sigma$ be classical implementations of P^1_Σ, P^2_Σ. (I.e., $P^{1,class}_\Sigma, P^{2,class}_\Sigma$ have the same output distribution but do not perform quantum computations or keep a quantum state. $P^{1,class}_\Sigma, P^{2,class}_\Sigma$ might not be polynomial-time, and the state they keep might not be polynomial space.)

We use Theorem 4 to transform Game 1. For a fixed η, let A^H_0 run $(x,w) \leftarrow A^H(1^\eta)$ (and return nothing). Let $A_C()$ run $com \leftarrow P^{1,class}_\Sigma(1^\eta, x, w)$ and return $x\|com$. Let $A^H_2(ch)$ run $resp \leftarrow P^{2,class}_\Sigma(1^\eta, x, w, ch)$ and $ok_V \leftarrow V_\Sigma(1^\eta, x, com, ch, resp)$ and return $b := win := ((x,w) \in R_\eta \wedge ok_V = 0)$. (Note: A_C and A^H_2 are not necessarily polynomial-time, we will only use that A^H_0 is polynomial-time.)

Let p_1, p_2 denote the first and second probability in Theorem 4, respectively. By construction, $p_1 = \Pr[win = 1 : \text{Game 1}]$.

Furthermore, $p_2 = \Pr[win = 1 : \text{Game 2}]$ for the following game:

Game 2 *$H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}_\eta, \ell^{out}_\eta)$, $(x,w) \leftarrow A^H(1^\eta)$, $com \leftarrow P^1_\Sigma(1^\eta, x, w)$, $ch \xleftarrow{\$} \{0,1\}^{\ell^{ch}}$, $resp \leftarrow P^2_\Sigma(1^\eta, x, w, ch)$, $ok_V \leftarrow V_\Sigma(1^\eta, x, com, ch, resp)$, $win := ((x,w) \in R_\eta \wedge ok_V = 0)$.*

Then Theorem 4 implies that

$$\big|\Pr[win = 1 : \text{Game 1}] - \Pr[win = 1 : \text{Game 2}]\big| = |p_1 - p_2| \leq (4 + \sqrt{2})\sqrt{q_A}2^{-k/4} =: \mu \qquad (5)$$

where q_A is the number of queries performed by A^H_0, and k the collision-entropy of $x\|com$. Since A is polynomial-time, q_A is polynomially bounded. And since Σ has unpredictable commitments, k is superlogarithmic. Thus μ is negligible.

Since Σ has completeness, $\Pr[win = 1 : \text{Game 2}]$ is negligible. From (5) it then follows that $\Pr[win = 1 : \text{Game 1}]$ is negligible. This shows that (P_{FS}, V_{FS}) has completeness. □

7.2 Zero-knowledge

Theorem 19 (Fiat-Shamir is zero-knowledge) *Assume that Σ is honest-verifier zero-knowledge and has completeness and unpredictable commitments.*

Then the Fiat-Shamir proof system (P_{FS}, V_{FS}) is zero-knowledge.

Proof. In this proof, we will in many places omit the security parameter η for readability. (E.g., we write $\{0,1\}^{\ell^{ch}}$ instead of $\{0,1\}^{\ell^{ch}_\eta}$ and $S_\Sigma(x)$ instead of $S_\Sigma(1^\eta, x)$.) It is to be understood that this is merely a syntactic omission, the variables and algorithms still depend on η.

To show that Fiat-Shamir is zero-knowledge, we first define a simulator S_{FS}, see Figure 1. In the definition of S_{FS} we use the honest-verifier simulator S_Σ for Σ (see Definition 6) which exists since Σ is HVZK by assumption. Fix a quantum-polynomial-time adversary A, and a quantum state ρ (that may depend on η). Let q_H and q_P denote polynomial upper bounds on the number of queries performed by A to the random oracle H and the prover/simulator, respectively. We need to show that (1) is negligible (with $P := P_{FS}$ and $S := S_{FS}$). For this, we transform the lhs of (1) into the rhs of (1) using a sequences of games.

Game 1 (Real world) *$b \leftarrow A^{H,P_{FS}}(\rho)$.*

Game 2 (Programming H) *$b \leftarrow A^{H,P^*}(\rho)$ with the following oracle P^*:*

$P^*(x,w)$ runs $com \leftarrow P^1_\Sigma(x,w)$, $ch \xleftarrow{\$} \{0,1\}^{\ell^{ch}}$, $H(x\|com) := ch$, $resp \leftarrow P^2_\Sigma(x,w,ch)$. Then it returns $\pi := com\|resp$.

$\boldsymbol{P_{FS}}$:

```
Input: $1^\eta$, $x$, $w$
Oracles: Classical queries to $H$.

$com \leftarrow P^1_\Sigma(1^\eta, x, w)$
$ch := H(x\|com)$
$resp \leftarrow P^2_\Sigma(1^\eta, x, w, ch)$
return $\pi := com\|resp$
```

$\boldsymbol{V_{FS}}$:

```
Input: $1^\eta$, $x$, $\pi$
Oracles: Classical queries to $H$.

$com\|resp := \pi$
$ch := H(x\|com)$
return $V_\Sigma(1^\eta, x, com, ch, resp)$
```

$\boldsymbol{S_{FS}}$:

```
Input: $1^\eta$, $x$
Oracles: Write access to $H$.

$(com, ch, resp) \leftarrow S_\Sigma(1^\eta, x)$
if $V_\Sigma(1^\eta, x, com, ch, resp) = 1$ then
    $H(x\|com) := ch$
return $\pi := com\|resp$
```

Figure 1: Prover P_{FS} and verifier V_{FS} of the Fiat-Shamir proof system. S_{FS} is the simulator constructed in the proof of Theorem 19.

Notice that P^* reprograms the random oracle in a similar way as the simulator does. Thus, P^* is not a valid prover any more, but the game is well-defined nonetheless.

In order to relate Game 1 and Game 2, we define a hybrid game:

Game 3$_i$ **(Hybrid)** *$b \leftarrow A^{H,P'}(\rho)$ where P' behaves as P_{FS} in the first i invocations, and as P^* (see Game 2) in all further invocations.*

Fix some $i \geq 0$ and some η. We will now bound $\bigl|\Pr[b = 1 : \text{Game } 3_i] - \Pr[b = 1 : \text{Game } 3_{i+1}]\bigr|$ by applying Theorem 4. Let $A^H_0()$ be an algorithm that executes $A^{H,P'}(\rho)$ until just before the i-th query to P'.[15] Note that at that point, the query input x, w for the $(i+1)$-st P'-query are fixed. Let $P^{1,class}_\Sigma, P^{2,class}_\Sigma$ be classical implementations of P^1_Σ, P^2_Σ. (I.e., $P^{1,class}_\Sigma, P^{2,class}_\Sigma$ have the same output distribution but do not perform quantum computations or keep a quantum state. $P^{1,class}_\Sigma, P^{2,class}_\Sigma$ might not be polynomial-time.) Let $A_C()$ compute $com \leftarrow P^{1,class}_\Sigma(x, w)$ and return $x\|com$ if $(x, w) \in R$. (If $(x, w) \notin R$, $A_C()$ instead outputs a η uniformly random bits.) Let $A^H_2(ch)$ compute $resp \leftarrow P^{2,class}_\Sigma(x, w, ch)$, set $\pi := com\|resp$, and then finish the execution of A^H using π as the response of the $(i+1)$-st P'-query. A^H_2 outputs the output of A^H. Note that in the execution of A^H_2, P' will actually behave like P^* and thus reprogram the random oracle H. A^H_2 does not actually reprogram H (it only has readonly access to it), but instead maintains a list of all changes performed by P^* to simulate queries to H performed by A accordingly).

Since Σ has unpredictable commitments, the output of P^1_Σ has collision-entropy $\geq k(\eta)$ for some superlogarithmic k, assuming $(x, w) \in R$. Hence the output of A_C has collision-entropy $\geq k' := \min\{\eta, k\}$.

Since A makes at most q_H queries to H, and at most q_P queries to the prover, and since P_{FS} and P^* make one and zero queries to H, respectively, A^H_0 makes at most $q_A := q_H + q_P$ queries to H.

[15] Note that A^H_0 has both ρ and the security parameter η hardcoded. This is no problem in the present case because we Theorem 4 does not need A^H_0, A_C, A^H_2 to be efficient.

Let

$$\begin{aligned}P_{lhs} := \Pr[b = 1 : H \xleftarrow{\$} \mathrm{Fun}(\ell^x + \ell^{com}, \ell^{ch}), A_0^H(), x\|com \leftarrow A_C(),\\ ch := H(x\|com), b \leftarrow A_2^H(ch)],\\ P_{rhs} := \Pr[b = 1 : H \xleftarrow{\$} \mathrm{Fun}(\ell^x + \ell^{com}, \ell^{ch}), A_0^H(), x\|com \leftarrow A_C(),\\ ch \xleftarrow{\$} \{0,1\}^{\ell^{ch}}, H(x\|com) := ch, b \leftarrow A_2^H(ch)]\end{aligned}$$

Then, by Theorem 4,

$$\left|P_{lhs} - P_{rhs}\right| \le (4 + \sqrt{2})\sqrt{q_A}2^{-k/4} =: \mu_1. \tag{6}$$

Since k is superlogarithmic, and $q_A = q_H + q_P$ is polynomially bounded, we have that μ_1 is negligible.

With those definitions, we have that

$$P_{lhs} = \Pr[b = 1 : \text{Game } 3_{i+1}] \tag{7}$$

because $x\|com \leftarrow A_C()$, $ch := H(x\|com)$ together with the steps $resp \leftarrow P_\Sigma^{2,class}(x, w, ch)$ and $\pi := com\|resp$ executed by A_2^H compute what P_{FS} would compute,[16] hence the $(i + 1)$-st query is exactly what it would be in Game 3_{i+1}.

And we have that

$$P_{rhs} = \Pr[b = 1 : \text{Game } 3_i] \tag{8}$$

because $x\|com \leftarrow A_C()$, $ch \xleftarrow{\$} \{0,1\}^{\ell^{ch}}$, $H(x\|com) := ch$, together with the steps $resp \leftarrow P_\Sigma^{2,class}(x, w, ch)$ and $\pi := com\|resp$ executed by A_2^H compute what P^* would compute, hence the i-st query is exactly what it would be in Game 3_i.

From (6)–(8), we have (for all i and η):

$$\left|\Pr[b = 1 : \text{Game } 3_{i+1}] - \Pr[b = 1 : \text{Game } 3_i]\right| \le \mu_1 \tag{9}$$

Furthermore, we have that

$$\begin{aligned}&\Pr[b = 1 : \text{Game } 3_0] = \Pr[b = 1 : \text{Game } 2]\\ \text{and}\qquad &\Pr[b = 1 : \text{Game } 3_{q_P}] = \Pr[b = 1 : \text{Game } 1]\end{aligned} \tag{10}$$

by definition of the involved games. (For the second equality, we use that $A^{H,P'}$ makes at most q_P queries to P'.)

Thus we have

$$\begin{aligned}&\left|\Pr[b = 1 : \text{Game } 1] - \Pr[b = 1 : \text{Game } 2]\right|\\ &\overset{(10)}{=} \left|\Pr[b = 1 : \text{Game } 3_{q_P}] - \Pr[b = 1 : \text{Game } 3_0]\right|\\ &\le \sum_{i=0}^{q_P-1} \left|\Pr[b = 1 : \text{Game } 3_{i+1}] - \Pr[b = 1 : \text{Game } 3_i]\right|\\ &\overset{(9)}{\le} \sum_{i=0}^{q_P-1} \mu_1 = q_P\mu_1 =: \mu_2.\end{aligned} \tag{11}$$

Since μ_1 is negligible and q_P is polynomially bounded, μ_2 is negligible.

[16] The case that $A_C()$ outputs η random bits when $(x, w) \notin R$ does not occur since A queries the prover only with $(x, w) \in R$ by Definition 8, and hence A_0^H only chooses x, w with $(x, w) \in R$.

Game 4 $b \leftarrow A^{H,P^{**}}(\rho)$ *with the following oracle* P^{**}*:*
$P^{**}(x,w)$ *runs:* $com \leftarrow P^1_\Sigma(x,w)$, $ch \overset{\$}{\leftarrow} \{0,1\}^{\ell^{ch}}$, $resp \leftarrow P^2_\Sigma(x,w,ch)$, *if* $V_\Sigma(x, com, ch, resp) = 1$ *then* $H(x\|com) := ch$. *Then it returns* $\pi := com\|resp$.

By assumption, Σ has completeness. Furthermore, A never queries $(x,w) \notin R$ from P^{**} (see Definition 8). Thus with overwhelming probability, $V_\Sigma(x, com, ch, resp) = 1$ holds in each query to P^{**}. Thus with overwhelming probability, the condition $V_\Sigma(x, com, ch, resp) = 1$ in the if-statement is satisfied in each invocation of P^{**}, and P^{**} performs the same steps as P^*. Thus for some negligible μ_3 we have

$$\big|\Pr[b = 1 : \text{Game 2}] - \Pr[b = 1 : \text{Game 4}]\big| \leq \mu_3. \tag{12}$$

Let S_{FS} be as in Figure 1.

Game 5 $b \leftarrow A^{H,S'_{FS}}$. *(Here* $S'_{FS}(x,w)$ *runs* $S_{FS}(x)$*, analogous to* S' *in Definition 8.)*

By definition, $P^{**}(x,w)$ performs the following steps:

- $com \leftarrow P^1_\Sigma(x,w)$, $ch \leftarrow \{0,1\}^{\ell^{ch}}$, $resp \leftarrow P^2_\Sigma(x,w,ch)$, if $V_\Sigma(x, com, ch, resp) = 1$ then $H(x\|com) := ch$.

In constract, S'_{FS} performs:

- $(com, ch, resp) \leftarrow S_\Sigma(x)$, if $V_\Sigma(x, com, ch, resp) = 1$ then $H(x\|com) := ch$.

By definition of honest-verifier zero-knowledge, $(com, ch, resp)$ as chosen in the first item is indistinguishable by a quantum-polynomial-time algorithm from $(com, ch, resp)$ as chosen second item, assuming $(x,w) \in R$. (And $(x,w) \in R$ is guaranteed since by Definition 8, A only queries $(x,w) \in R$ from the prover/simulator.) A standard hybrid argument then shows that no quantum-polynomial-time adversary can distinguish oracle access to P^* from oracle access to S_{FS}. Hence

$$\big|\Pr[b = 1 : \text{Game 4}] - \Pr[b = 1 : \text{Game 5}]\big| \leq \mu_4 \tag{13}$$

for some negligible μ_4.

Altogether, we have

$$\big|\Pr[b = 1 : \text{Game 1}] - \Pr[b = 1 : \text{Game 5}]\big| \overset{(11)-(13)}{\leq} \mu_2 + \mu_3 + \mu_4.$$

Since μ_2, μ_3, and μ_4 are negligible, so is $\mu_2 + \mu_3 + \mu_4$. Thus (1) from Definition 8 is negligible. This shows that S_{FS} is a simulator as required by Definition 8, thus Fiat-Shamir is zero-knowledge. □

7.3 Soundness

Theorem 20 *Assume that* Σ *has perfect special soundness, and that* ℓ^{ch}_η *is superlogarithmic. Then the Fiat-Shamir proof system* (P_{FS}, V_{FS}) *is sound.*

It may seem surprising that we need an information-theoretical property (perfect special soundness of Σ) to get a computational property (soundness of (P_{FS}, V_{FS})). Might it not be sufficient to assume that Σ has computational special soundness? Unfortunately, [ARU14] shows that (relative to certain oracles), there is a sigma-protocol Σ with computational special soundness such that (P_{FS}, V_{FS}) is not sound. So, we cannot expect Theorem 20 to hold assuming only computational special soundness, at least not with a relativizing proof.[17]

[17] [ARU14] leaves the possibility of a relativizing proof that Fiat-Shamir is secure if Σ has perfectly unique responses and (computational) special soundness, though. But then we have another information-theoretical assumption, namely perfectly unique responses.

Proof. In this proof, we will in most places omit the security parameter η for readability. (E.g., we write ℓ^{ch} instead of ℓ^{ch}_η and $S_\Sigma(x)$ instead of $S_\Sigma(\eta, x)$.) It is to be understood that this is merely a syntactic omission, the variables and algorithms still depend on η.

Let $x \in \{0,1\}^{\ell^x}$, $com \in \{0,1\}^{com}$. We call a $ch \in \{0,1\}^{\ell^{ch}}$ *promising for* (x, com) iff there exists a $resp \in \{0,1\}^{\ell^{resp}}$ such that $V_\Sigma(x, com, ch, resp) = 1$.

Claim 1 *For any $x \in \{0,1\}^{\ell^x} \setminus L_R$ and any $com \in \{0,1\}^{\ell^{com}}$, there exists at most one promising ch.*

This claim follows directly from the fact that Σ has perfect special soundness: If there would be $x \notin L_R$ and com, and two promising $ch \neq ch'$ for (x, com), then by definition of promising, there would also be $resp, resp'$ such that $V_\Sigma(x, com, ch, resp) = V_\Sigma(x, com, ch', resp') = 1$. Thus the extractor from the definition of perfect special soundness (see Definition 6) would return some w with $(x, w) \in R$ upon input $(x, com, ch, resp, ch', resp')$. But $(x, w) \in R$ implies $x \in L_R$, in contradiction to the assumption of the claim.

We now define an auxiliary function $f_\eta : \{0,1\}^{\ell^x_\eta + \ell^{com}_\eta} \to \{0,1\}^{\ell^{ch}_\eta}$ as follows: If $x \notin L_R$, and there exists a promising ch for (x, com), let $f(x\|com)$ be that ch. (The value ch is unique due to Claim 1.) Otherwise $f(x\|com) := 0^{\ell^{ch}}$.

Let A be a quantum-polynomial-time q-query adversary that breaks the soundness of Fiat-Shamir given some initial state ρ. That is, δ is non-negligible where

$$\delta := \Pr[ok_V = 1 \wedge x \notin L_R : (x, com\|resp) \leftarrow A^H(\rho),\ ok_V \leftarrow V^H_{FS}(x, com\|resp)].$$

By definition of V_{FS}, we have that $ok_V = 1$ implies that $V_\Sigma(x, com, ch, resp) = 1$ where $ch := H(x\|com)$. In particular, $ch = H(x\|com)$ is promising for (x, com). Thus, $ok_V = 1 \wedge x \notin L_R$ implies that $f(x\|com) = H(x\|com)$. Hence

$$\Pr[f(x\|com) = H(x\|com) : (x, com\|resp) \leftarrow A^H(\rho)] \geq \delta. \tag{14}$$

Let $B^H(\rho)$ perform the following steps: It defines $H'(x\|com) := H(x\|com) \oplus f(x\|com)$. It invokes $(x, com\|resp) \leftarrow A^{H'}(\rho)$. It returns $x\|com$.

Although B may not be quantum-polynomial-time (f may not be efficiently computable), B performs only q queries since each query to H' can be implemented using on query to H.[18]

If H is uniformly random, then H' is uniformly random. Thus by (14), $H'(x\|com) = f(x\|com)$ with probability $\geq \delta$. Thus $H(x\|com) = 0^{\ell^{ch}}$ with probability $\geq \delta$. In other words, B finds a zero-preimage of H with probability $\geq \delta$. By Lemma 5, this implies that $\delta \leq 32 \cdot 2^{-\ell^{ch}} \cdot (q+1)^2$. Since q is polynomially bounded (as A is quantum-polynomial-time) and ℓ^{ch} is superlogarithmic, we have that δ is negligible.

Since this holds for all quantum-polynomial-time A, it follows that (P_{FS}, V_{FS}) is sound. □

7.4 Simulation-soundness

Theorem 21 (Fiat-Shamir is simulation-sound) *Assume that Σ has perfect special soundness, unique responses, and that ℓ^{ch}_η is superlogarithmic.*

Then the Fiat-Shamir proof system (P_{FS}, V_{FS}) is simulation-sound with respect to the simulator S_{FS} from Figure 1.

[18] To implement the unitary $\mathbf{U}_{H'} : |a\|b\rangle \mapsto |a\|(b \oplus H'(a))\rangle$, B first invokes $\mathbf{U}_H : |a\|b\rangle \mapsto |a\|(b \oplus H(a))\rangle$ by using the oracle H, and then $\mathbf{U}_f : |a\|b\rangle \mapsto |a\|(b \oplus f(a))\rangle$ which B implements on its own.

Unique responses are necessary in this theorem. As pointed out in [FKMV12], if Σ does not have unique responses, it cannot be simulation-sound, even in the classical case. Namely, if we do not require unique responses, it could be that whenever $(\mathit{com}, \mathit{ch}, \mathit{resp}\|0)$ is a valid proof in Σ, so is $(\mathit{com}, \mathit{ch}, \mathit{resp}\|1)$, and vice versa. Thus any valid Fiat-Shamir proof $\mathit{com}\|(\mathit{resp}\|0)$ could be efficiently transformed into another valid Fiat-Shamir proof $\mathit{com}\|(\mathit{resp}\|1)$ for the same statement. This would contradict the simulation-soundness of (P_{FS}, V_{FS}).

Proof. In this proof, we will in most places omit the security parameter η for readability. (E.g., we write ℓ^{ch} instead of ℓ^{ch}_η and $S_\Sigma(x)$ instead of $S_\Sigma(\eta, x)$.) It is to be understood that this is merely a syntactic omission, the variables and algorithms still depend on η. For brevity, we will also omit the choosing of the random oracle H from all games. That is, every game implicitly starts with $H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}, \ell^{out})$.

Fix a quantum-polynomial-time adversary A, and a density operator ρ. Let q_H and q_P denote polynomial upper bounds on the number of queries performed by A to the random oracle H and the prover/simulator, respectively. We need to show that (2) holds with $V := V_{FS}$ and $S := S_{FS}$ for some negligible μ. For this, we transform the game from (2) using a sequence of games until we reach a game where the adversary has a negligible success probability. The following game encodes the game from (2): (We write $\mathit{com}\|\mathit{resp}$ instead of π to be able to explicitly refer to the two components of π.)

Game 1 (Real world) $S_A \leftarrow \rho$. $x\|\mathit{com}\|\mathit{resp} \leftarrow A^{H,S_{FS}}(S_A)$. $\mathit{ok}_V \leftarrow V_{FS}^{H_{\mathit{final}}}(x, \mathit{com}\|\mathit{resp})$. $\mathit{win} := \big(\mathit{ok}_V = 1 \land x \notin L_R \land (x, \mathit{com}\|\mathit{resp}) \notin \textsf{S-queries}\big)$.

Here we use H to refer to the initial value of the random oracle H, and H_{final} to the value of H after it has been reprogrammed by S_{FS}. (See Definition 10.)

We define a variant of the random variable S-queries. Let S-queries* be the list of all S_{FS}-queries $(x', \mathit{com}'\|\mathit{resp}', \mathit{ch}')$ where x' was the input to S_{FS}, $\mathit{com}'\|\mathit{resp}'$ was the response of S_{FS}, and ch' was the value of $H(x'\|\mathit{com}')$ right after the query to S_{FS}. (Note that $H(x'\|\mathit{com}')$ may change later due to reprogramming.) Notice that the only difference between S-queries and S-queries* is that in the latter, we additionally track the values $\mathit{ch}' = H(x'\|\mathit{com}')$.

Let RespConflict denote the event that $V_\Sigma(x, \mathit{com}, H_{\mathit{final}}(x\|\mathit{com}), \mathit{resp}) = 1$ and that there is a query $(x', \mathit{com}'\|\mathit{resp}', \mathit{ch}') \in \textsf{S-queries}$ with $x' = x$, $\mathit{com}' = \mathit{com}$, $\mathit{ch}' = H_{\mathit{final}}(x\|\mathit{com})$, and $\mathit{resp}' \neq \mathit{resp}$ and $V_\Sigma(x, \mathit{com}, \mathit{ch}', \mathit{resp}') = 1$.

Since Σ has unique responses, it follows that

$$\Pr[\textsf{RespConflict} : \text{Game 1}] \leq \mu'$$

for some negligible μ'. (Otherwise, we could construct an adversary that simulates Game 1, and then searches for $(x, \mathit{com}\|\mathit{resp}', \mathit{ch}) \in \textsf{S-queries}$ with $V_\Sigma(x, \mathit{com}, \mathit{ch}, \mathit{resp}') = 1$ and $\mathit{resp}' \neq \mathit{resp}$.

Thus

$$\Big|\Pr[\mathit{win} : \text{Game 1}] - \Pr[\mathit{win} \land \neg\textsf{RespConflict} : \text{Game 1}]\Big| \leq \mu'.$$

We now show that in Game 1, we have

$$V_{FS}^{H_{\mathit{final}}}(x, \mathit{com}\|\mathit{resp}) = 1 \;\land\; (x, \mathit{com}\|\mathit{resp}) \notin \textsf{S-queries} \;\land\; \neg\textsf{RespConflict} \\ \implies V_{FS}^{H}(x, \mathit{com}\|\mathit{resp}) = 1. \quad (15)$$

Assume for contradiction that (15) does not hold, i.e., that $V_{FS}^{H_{\mathit{final}}}(x, \mathit{com}\|\mathit{resp}) = 1$ and $(x, \mathit{com}\|\mathit{resp}) \notin \textsf{S-queries}$ and $\neg$RespConflict, but $V_{FS}^{H}(x, \mathit{com}\|\mathit{resp}) = 0$ in some execution of Game 1. Since V_{FS}^H queries H only for input $x\|\mathit{com}$, this implies that $H_{\mathit{final}}(x\|\mathit{com}) \neq H(x\|\mathit{com})$.

Since H is only reprogrammed by invocations of S_{FS}, $H(x\|com)$ must have been reprogrammed by S_{FS}. Consider the last query to S_{FS} that programmed $H(x\|com)$ (in case there are several). By construction of S_{FS}, that query had input x, and returns $(com, resp')$ for some $resp'$. In particular, $(x, com\|resp') \in \mathsf{S\text{-}queries}$. Let ch be the challenge chosen by S_{FS} in that query. Then $(x, com\|resp', ch) \in \mathsf{S\text{-}queries}^*$. By construction of S_{FS}, we have $V_\Sigma(x, com, ch, resp') = 1$ (else H would not have been reprogrammed in that query) and $H_{\mathit{final}}(x\|com) = ch$ (because we are considering the last S_{FS}-query that programmed $H(x\|com)$). Since $(x, com\|resp) \notin \mathsf{S\text{-}queries}$ and $(x, com\|resp') \in \mathsf{S\text{-}queries}$, we have $resp \neq resp'$. Since $V_{FS}^{H_{\mathit{final}}}(x, com\|resp) = 1$ and $ch = H_{\mathit{final}}(x\|com)$, we have that $V_\Sigma(x, com, ch, resp) = 1$ by definition of V_{FS}. Summarizing, we have $V_\Sigma(x, com, ch, resp) = 1$ and $ch = H_{\mathit{final}}(x\|com)$ and $V_\Sigma(x, com, ch, resp') = 1$ and $(x, com\|resp', ch) \in \mathsf{S\text{-}queries}^*$ and $resp \neq resp'$. By definition of $\mathsf{RespConflict}$, this contradicts $\neg\mathsf{RespConflict}$. Thus our assumption that (15) does not hold was false. Thus (15) follows.

We now consider a variant of Game 1 where the verifier in the end gets access to H instead of H_{final}. (That is, we can think of H being reset to its original state without the simulator's changes.)

(In this and the following games, we will not need to refer to com and $resp$ individually any more, so we just write π instead of $com\|resp$.)

Game 2 (Unchanged H) $S_A \leftarrow \rho.\ x\|\pi \leftarrow A^{H,S_{FS}}(S_A).\ ok_V \leftarrow V_{FS}^{H}(x,\pi).\ win := \big(ok_V = 1 \land x \notin L_R \land (x,\pi) \notin \mathsf{S\text{-}queries}\big).$

By (15), we get

$$\Pr[win : \text{Game 2}] \geq \Pr[win \land \neg\mathsf{RespConflict} : \text{Game 1}]. \tag{16}$$

Furthermore, we have

$$\Pr[ok_V = 1 \land x \notin L_R : \text{Game 2}] \geq \Pr[win : \text{Game 2}].$$

We define an oracle algorithm B. When invoked as $B^H(S_A)$, it simulates an execution of $A^{H,S_{FS}}(S_A)$. Note that S_{FS} can program the random oracle H. In order to simulate this, B^H keeps track of the assignments ass_S made by S_{FS}, and then provides A with the oracle $H(ass_S)$ (i.e., H reprogrammed according to the assignment-list ass_S) instead of H. Then $B^H(S_A)$ will have the same distribution of outputs as $A^{H,S_{FS}}(S_A)$. (But of course, any reprogramming of H performed by the S_{FS} simulated by B will not have any effect beyond the execution of B. That is, the function H before and after the invocation of B^H will be the same.)

By construction of B (and because V_{FS} gets access to H and not H_{final} in (16)), we then have

$$\Pr[win : \text{Game 3}] = \Pr[ok_V = 1 \land x \notin L_R : \text{Game 2}].$$

Game 3 (Adversary B) $S_A \leftarrow \rho.\ x\|\pi \leftarrow B^H(S_A).\ ok_V \leftarrow V_{FS}^{H}(x,\pi).\ win := \big(ok_V = 1 \land x \notin L_R$ ~~$\land (x,\pi) \notin$ S-queries~~$\big).$

By Theorem 20, (P_{FS}, V_{FS}) is sound. Furthermore, since A and S_{FS} are quantum-polynomial-time, B is quantum-polynomial-time. Thus by definition of soundness (Definition 9), there is a negligible μ such that

$$\Pr[win : \text{Game 3}] \leq \mu.$$

Combining the inequalities from this proof, we get

$$\Pr[win : \text{Game 1}] \leq \mu + \mu'.$$

And $\mu + \mu'$ is negligible. Since Game 1 is the game from Definition 10 for (P_{FS}, V_{FS}), and since A was an arbitrarily quantum-polynomial-time oracle algorithm, it follows that (P_{FS}, V_{FS}) is simulation-sound. □

7.5 Simulation-sound extractability

Theorem 22 (Fiat-Shamir is simulation-sound extractable) *Assume that Σ has unique responses. Assume that the Fiat-Shamir proof system (P_{FS}, V_{FS}) based on Σ is extractable.*

Then the Fiat-Shamir proof system (P_{FS}, V_{FS}) is simulation-sound extractable with respect to the simulator S_{FS} from Figure 1.

Proof. In this proof, we will in most places omit the security parameter η for readability. (E.g., we write ℓ^{ch} instead of ℓ^{ch}_η and $S_\Sigma(x)$ instead of $S_\Sigma(\eta, x)$.) It is to be understood that this is merely a syntactic omission, the variables and algorithms still depend on η. For brevity, we will also omit the choosing of the random oracle H from all games. That is, every game implicitly starts with $H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}, \ell^{out})$.

Fix a quantum-polynomial-time adversary A, and a density operator ρ. Let q_H and q_P denote polynomial upper bounds on the number of queries performed by A to the random oracle H and the prover/simulator, respectively. We need to show that (4) holds with $V := V_{FS}$ and $S := S_{FS}$ for some constant $d > 0$, some polynomials p, ℓ, and some negligible μ, and for some quantum-polynomial-time E (where ℓ is independent of Π, and E is independent of Π, A). For this, we transform the game **Prove** from (4) into the game **Extract** from (4) using a sequence of games. The following game encodes the game **Prove**: (We write $com\|resp$ instead of π to be able to explicitly refer to the two components of π.)

Game 1 (Real world) $S_A \leftarrow \rho.$ $x\|com\|resp \leftarrow A^{H,S_{FS}}(S_A).$ $ok_V \leftarrow V_{FS}^{H_{final}}(x, com\|resp).$ $ok_A \leftarrow \Pi^{H_{final}}_{x\|com\|resp}(S_A).$ $win := \big(ok_V = 1 \land ok_A = 1 \land (x, com\|resp) \notin \mathsf{S\text{-}queries}\big).$

Here we use H to refer to the initial value of the random oracle H, and H_{final} to the value of H after it has been reprogrammed by S_{FS}. (See Definition 16.)

We define a variant of the random variable $\mathsf{S\text{-}queries}$. Let $\mathsf{S\text{-}queries}^*$ be the list of all S_{FS}-queries $(x', com'\|resp', ch')$ where x' was the input to S_{FS}, $com'\|resp'$ was the response of S_{FS}, and ch' was the value of $H(x'\|com')$ right after the query to S_{FS}. (Note that $H(x'\|com')$ may change later due to reprogramming.) Notice that the only difference between $\mathsf{S\text{-}queries}$ and $\mathsf{S\text{-}queries}^*$ is that in the latter, we additionally track the values $ch' = H(x'\|com')$.

Let $\mathsf{RespConflict}$ denote the event that $V_\Sigma(x, com, H_{final}(x\|com), resp) = 1$ and that there is a query $(x', com'\|resp', ch') \in \mathsf{S\text{-}queries}$ with $x' = x$, $com' = com$, $ch' = H_{final}(x\|com)$, and $resp' \neq resp$ and $V_\Sigma(x, com, ch', resp') = 1$.

Since Σ has unique responses, it follows that

$$\Pr[\mathsf{RespConflict} : \text{Game 1}] \leq \mu'$$

for some negligible μ'. (Otherwise, we could construct an adversary that simulates Game 1, and then searches for $(x, com\|resp', ch) \in \mathsf{S\text{-}queries}$ with $V_\Sigma(x, com, ch, resp') = 1$ and $resp' \neq resp$.

Thus

$$\Big|\Pr[win : \text{Game 1}] - \Pr[win \land \neg\mathsf{RespConflict} : \text{Game 1}]\Big| \leq \mu'. \tag{17}$$

We now show that in Game 1, we have

$$V_{FS}^{H_{final}}(x, com\|resp) = 1 \;\land\; (x, com\|resp) \notin \mathsf{S\text{-}queries} \;\land\; \neg\mathsf{RespConflict} \implies V^H_{FS}(x, com\|resp) = 1. \tag{18}$$

Assume for contradiction that (18) does not hold, i.e., that $V_{FS}^{H_{\mathit{final}}}(x, com\|resp) = 1$ and $(x, com\|resp) \notin \mathsf{S\text{-}queries}$ and $\neg\mathsf{RespConflict}$, but $V_{FS}^{H}(x, com\|resp) = 0$ in some execution of Game 1. Since V_{FS}^{H} queries H only for input $x\|com$, this implies that $H_{\mathit{final}}(x\|com) \neq H(x\|com)$. Since H is only reprogrammed by invocations of S_{FS}, $H(x\|com)$ must have been reprogrammed by S_{FS}. Consider the last query to S_{FS} that programmed $H(x\|com)$ (in case there are several). By construction of S_{FS}, that query had input x, and returns $(com, resp')$ for some $resp'$. In particular, $(x, com\|resp') \in \mathsf{S\text{-}queries}$. Let ch be the challenge chosen by S_{FS} in that query. Then $(x, com\|resp', ch) \in \mathsf{S\text{-}queries}^*$. By construction of S_{FS}, we have $V_\Sigma(x, com, ch, resp') = 1$ (else H would not have been reprogrammed in that query) and $H_{\mathit{final}}(x\|com) = ch$ (because we are considering the last S_{FS}-query that programmed $H(x\|com)$). Since $(x, com\|resp) \notin \mathsf{S\text{-}queries}$ and $(x, com\|resp') \in \mathsf{S\text{-}queries}$, we have $resp \neq resp'$. Since $V_{FS}^{H_{\mathit{final}}}(x, com\|resp) = 1$ and $ch = H_{\mathit{final}}(x\|com)$, we have that $V_\Sigma(x, com, ch, resp) = 1$ by definition of V_{FS}. Summarizing, we have $V_\Sigma(x, com, ch, resp) = 1$ and $ch = H_{\mathit{final}}(x\|com)$ and $V_\Sigma(x, com, ch, resp') = 1$ and $(x, com\|resp', ch) \in \mathsf{S\text{-}queries}^*$ and $resp \neq resp'$. By definition of $\mathsf{RespConflict}$, this contradicts $\neg\mathsf{RespConflict}$. Thus our assumption that (18) does not hold was false. Thus (18) follows.

We now consider a variant of Game 1 where the verifier in the end gets access to H instead of H_{final}. (That is, we can think of H being reset to its original state without the simulator's changes.)

(In this and the following games, we will not need to refer to com and $resp$ individually any more, so we just write π instead of $com\|resp$.)

Game 2 (Unchanged H) $S_A \leftarrow \rho$. $x\|\pi \leftarrow A^{H,S_{FS}}(S_A)$. $ok_V \leftarrow V_{FS}^{H}(x, \pi)$. $ok_A \leftarrow \Pi_{x\|\pi}^{H_{\mathit{final}}}(S_A)$. $win := \bigl(ok_V = 1 \wedge ok_A = 1 \wedge (x,\pi) \notin \mathsf{S\text{-}queries}\bigr)$.

By (18), we get

$$\Pr[win : \text{Game 2}] \geq \Pr[win \wedge \neg\mathsf{RespConflict} : \text{Game 1}]. \tag{19}$$

Furthermore, we have

$$\Pr[ok_V = ok_A = 1 : \text{Game 2}] \geq \Pr[win : \text{Game 2}]. \tag{20}$$

Let ass_S denote the assignments performed by S_{FS} to H. (These can be obtained by keeping a log of the classical computations performed by S_{FS}.) Note that $H_{\mathit{final}} = H(ass_S)$.

Let ℓ_{ass} be a polynomial upper bound on the length of ass_S (encoded as a bitstring). Such ℓ_{ass} exists since A performs only polynomially many queries and S_{FS} programs the random oracle at only one location upon each query.

Let T be an ℓ_{ass} qubit register. For given H, let $\tilde\Pi^H$ be the projective measurement that does the following: If T contains $|ass\rangle$, then $\tilde\Pi_{x\|\pi}^{H}$ measures S_A using $\Pi_{x\|\pi}^{H(ass)}$. (Recall that $H(ass)$ denotes the function H, updated according to the assignment ass.) Formally: Let $\Pi_{x\|\pi}^{H,b}$ denote the projector corresponding to outcome $b \in \{0,1\}$ of the measurement $\Pi_{x\|\pi}^{H}$. Then $\tilde\Pi_{x\|\pi}^{H,b}$ is the projector on S_A, T defined by:

$$\tilde\Pi_{x\|\pi}^{H,b} = \sum_{ass \in \{0,1\}^{\ell_{ass}}} \Pi_{x\|\pi}^{H(ass),b} \otimes |ass\rangle\langle ass|_T.$$

And $\tilde\Pi_{x\|\pi}^{H}$ is the projective measurement on S_A, T defined by the projectors $\tilde\Pi_{x\|\pi}^{H,0}, \tilde\Pi_{x\|\pi}^{H,1}$. (If ass is not a valid encoding of an assignment, then we interpret it as the empty assignment. I.e., $H(ass) = H$ in that case.)

Note that if we initialize $T \leftarrow |ass_S\rangle$, then $ok_A \leftarrow \tilde\Pi^H_{x\|\pi}(S_A, T)$ is equivalent to $ok_A \leftarrow \Pi^{H(ass_S)}_{x\|\pi}(S_A)$. And $ok_A \leftarrow \Pi^{H(ass_S)}_{x\|\pi}(S_A)$ is the same as $ok_A \leftarrow \Pi^{H_{\mathit{final}}}_{x\|\pi}(S_A)$ by definition of H_{final} and ass_S.

Thus we can rewrite Game 2 as follows:

Game 3 $S_A \leftarrow \rho$. $x\|\pi \leftarrow A^{H,S_{FS}}(S_A)$. $T \leftarrow |ass_S\rangle$. $ok_V \leftarrow V^H_{FS}(x, \pi)$. $ok_A \leftarrow \tilde\Pi^H_{x\|\pi}(S_A, T)$. $win := \big(ok_V = 1 \land ok_A = 1$~~$\land (x,\pi) \notin$ S-queries~~$\big)$.

We then have

$$\Pr[ok_V = ok_A = 1 : \text{Game 2}] = \Pr[win = 1 : \text{Game 3}]. \tag{21}$$

We define a pure oracle circuit B. This oracle circuit takes an oracle H, and operates on quantum registers S_A, T, and performs the following steps: It maintains an assignment-list ass_S that tracks where the random oracle has been reprogrammed. It then executes the oracle circuit A on register S_A. Whenever A performs a query to S_{FS}, B simulates the behavior of S_{FS}, and when S_{FS} programs the random oracle as $H(x\|com) := ch$, then B appends $(x\|com := ch)$ to ass_S. When A performs a query to H, B instead invokes $H(ass_S)$. After executing A, B initializes $T \leftarrow |ass_S\rangle$. The final state of B is then in the registers S_A, T.

Note that all of this can be done using only unitary operations (i.e., B is a pure oracle circuit, using a sufficiently large number of auxiliary qubits). Furthermore, given $\mathbf{shape}_A$ and a description of $\mathcal{U}_A$ (the unitary implemented by A), one can compute $\mathbf{shape}_B$ and the description of $\mathcal{U}_B$. Thus $B = B_\eta$ is a polynomial-time family of pure oracle circuits.

We then have

$$\Pr[win = 1 : \text{Game 4}] = \Pr[win = 1 : \text{Game 3}] \tag{22}$$

using the following game:

Game 4 (Simulating S_{FS}) $S_A, T \leftarrow \rho \otimes |0\rangle\langle 0|$. $x\|\pi \leftarrow B^H(S_A, T)$. $ok_V \leftarrow V^H_{FS}(x, \pi)$. $ok_A \leftarrow \tilde\Pi^H_{x\|\pi}(S_A, T)$. $win := \big(ok_V = 1 \land ok_A = 1\big)$.

By assumption, (P_{FS}, V_{FS}) is extractable. Game 4 is the game **Prove** from Definition 15 (with P_{FS}, V_{FS}, B, H, $\tilde\Pi$, and (S_A, T) in Game 4 being P, V, A, H, Π, and S_A in **Prove**). Thus there is and extractor E_0, a constant $d > 0$, a negligible function μ, a polynomial ℓ_0, and a polynomial p (where E_0, d are independent of $B, \tilde\Pi$, while ℓ_0 depends on B, and μ, p depend on $B, \tilde\Pi$) such that

$$\Pr[win = 1 : \text{Game 5}] \geq \tfrac{1}{p} \Pr[win = 1 : \text{Game 4}]^d - \mu \tag{23}$$

for the following game:

Game 5 (Extraction for B) $S_A, T \leftarrow \rho \otimes |0\rangle\langle 0|$. $(x, w, \pi, ass_E) \leftarrow E_0^{B^{\mathbf{rew}}(S_A,T), \tilde\Pi^{\mathbf{oracle},H,\ell_0}(S_A,T), H}(\ell_0, \mathbf{shape}_B)$, $ok_A \leftarrow \tilde\Pi^{H(ass_E)}_{x\|\pi}(S_A, T)$, $win := \big((x, w) \in R \land ok_A = 1\big)$.

Without loss of generality, we can assume $d \geq 1$.

By definition of $\tilde\Pi$, we have that "$ok_A \leftarrow \tilde\Pi^f_{x\|\pi}(S_A, T)$" is equivalent to "$ass_S \leftarrow \mathcal{M}(T)$, $ok_A \leftarrow \Pi^{f(ass_S)}_{x\|\pi}(S_A)$" for any function f. (As long as the register T is not used afterwards.) Here $\mathcal{M}$ is a measurement in the computational basis. In particular, this holds for $f := H(ass_E)$, in which case we have $f(ass_S) = H(ass_E)(ass_S) = H(ass)$ for $ass := ass_E\|ass_S$.

Thus we can rewrite Game 5 as follows:

Game 6 $S_A \leftarrow \rho$. $T \leftarrow |0\rangle\langle 0|$. $(x, w, \pi, ass_E) \leftarrow E_0^{B^{\mathbf{rew}}(S_A,T), \tilde\Pi^{\mathbf{oracle},H,\ell_0}(S_A,T), H}(\ell_0, \mathbf{shape}_B)$, $ass_S \leftarrow \mathcal{M}(T)$. $ass := ass_E\|ass_S$. $ok_A \leftarrow \Pi^{H(ass)}_{x\|\pi}(S_A)$, $win := \big((x, w) \in R \land ok_A = 1\big)$.

And we have

$$\Pr[\mathit{win} = 1 : \text{Game 6}] = \Pr[\mathit{win} = 1 : \text{Game 5}] \tag{24}$$

Let ℓ_1 be a polynomial upper bounds the length of ass_S. (Such a bound exists, since E_0 is polynomial-time.) Let $\ell := \ell_0 + \ell_1$. (Then ℓ is independent of Π, and ℓ_1 depends only on E_0.)

Let E be a quantum algorithm that, given oracle access to $A^{\mathbf{rew}}(S_A)$ and $\Pi^{\mathbf{oracle},H,\ell}(S_A)$ and input ℓ, executes the steps "$T \leftarrow |0\rangle\langle 0|$. $(x, w, \pi, \mathit{ass}_E) \leftarrow E_0^{B^{\mathbf{rew}}(S_A,T),\tilde{\Pi}^{\mathbf{oracle},H,\ell_0}(S_A,T),H}(\ell_0, \mathbf{shape}_B)$, $\mathit{ass}_S \leftarrow \mathcal{M}(T)$. $\mathit{ass} := \mathit{ass}_E \| \mathit{ass}_S$." from the above game and returns $(x, w, \pi, \mathit{ass})$. (Here E can compute $\ell_0 = \ell - \ell_1$, so that E will only depend on ℓ_1, and not on ℓ_0. Thus E depends on E_0, but not on Π or A.)

Such an algorithm E exists because queries to $B^{\mathbf{rew}}(S_A, T)$ can be simulated using oracle access to $A^{\mathbf{rew}}(S_A)$ (by construction of B), and queries to $\tilde{\Pi}^{\mathbf{oracle},H,\ell_0}(S_A, T)$ can be simulated using oracle access to $\Pi^{\mathbf{oracle},H,\ell}(S_A)$ (by construction of $\tilde{\Pi}$ and since $\ell = \ell_0 + \ell_1$ and since T contains at most ℓ_1 assignments). Note also that E is polynomial-time since E_0 is.

Then we have

$$\Pr[\mathit{win} : \text{Game 6}] = \Pr[\mathit{win} : \text{Game 7}] \tag{25}$$

for the following game:

Game 7 (Extraction) *$S_A \leftarrow \rho$. $(x, w, \pi, \mathit{ass}) \leftarrow E^{A^{\mathbf{rew}}(S_A),\Pi^{\mathbf{oracle},H,\ell}(S_A),H}(\ell, \mathbf{shape}_A)$, $\mathit{ok}_A \leftarrow \Pi_{x\|\pi}^{H(\mathit{ass})}(S_A)$, $\mathit{win} := \big((x, w) \in R \wedge \mathit{ok}_A = 1\big)$.*

And we have

$$\begin{aligned}
\Pr[\mathit{win} : \text{Game 7}] &\overset{(25),(24)}{=} \Pr[\mathit{win} : \text{Game 5}] \overset{(23)}{\geq} \tfrac{1}{p} \Pr[\mathit{win} = 1 : \text{Game 4}]^d - \mu \\
&\overset{(22)-(19)}{\geq} \tfrac{1}{p} \Pr[\mathit{win} = 1 \wedge \neg \mathsf{RespConflict} : \text{Game 1}]^d - \mu \\
&\overset{(17)}{\geq} \tfrac{1}{p} \big(\Pr[\mathit{win} = 1 : \text{Game 1}] - \mu'\big)^d - \mu \\
&\overset{(*)}{\geq} \tfrac{1}{p} \big(\Pr[\mathit{win} = 1 : \text{Game 1}]^d - \mu'\big) - \mu \\
&= \tfrac{1}{p} \Pr[\mathit{win} = 1 : \text{Game 1}]^d - \mu'' \qquad \text{for } \mu'' := \mu'/p + \mu.
\end{aligned}$$

Here $(*)$ holds whenever $\mu \leq \Pr[\mathit{win} = 1 : \text{Game 1}]$, because x^d has a derivative smaller than 1 for $x \in [0, 1]$. (And whenever $\mu > \Pr[\mathit{win} = 1 : \text{Game 1}]$, we trivially have $\Pr[\mathit{win} : \text{Game 7}] \geq \frac{1}{p} \Pr[\mathit{win} = 1 : \text{Game 1}]^d - \mu''$.)

Since Game 7 is the game **Extract** from Definition 16, and Game 1 is the game **Prove** from Definition 16, and since p is polynomially-bounded and μ'' is negligible, this shows that (P_{FS}, V_{FS}) is simulation-sound extractable. □

8 Signatures

Originally, Fiat-Shamir was constructed as a signature scheme [FS87]. Only later, [BR93] used the same idea to construct a non-interactive zero-knowledge proof. The fact that Fiat-Shamir gives rise to a secure signature scheme can be seen as a special case of its properties as a proof system. Namely, any non-interactive zero-knowledge proof system with simulation-sound extractability can be used as a signature scheme. In the quantum setting, [Unr15] showed that their construction of simulation-sound extractable non-interactive proofs gives rise to a signature scheme in the same way. In this section, we show that this is also the case for our definition of simulation-sound extractability. (This is a non-trivial generalization, since their definition assumed online-extractors and is thus much stronger than ours.) This proof can be seen as a sanity check for our definition of simulation-sound extractability (i.e., that our definition is not accidentally too weak). (Sometimes a definition is perfectly reasonable in the classical setting while its natural quantum counterpart is almost useless. An example is the classical definition of "computationally binding commitments" which was shown to imply almost no security in the quantum setting [ARU14].)

However, this result does not imply that Fiat-Shamir gives rise to a secure signature scheme because we are not able to prove that Fiat-Shamir is extractable. For analyzing Fiat-Shamir, we present a second result that shows under which conditions a simulation-sound zero-knowledge non-interactive proof system gives rise to a signature scheme. Combined with our results from Section 7, this implies security for Fiat-Shamir based signatures.

The basic idea of the construction of signatures from non-interactive proof systems (e.g., Fiat-Shamir) is the following: To sign a message m, one needs to show the knowledge of one's secret key. Thus, we need a relation R_η between public and secret keys, and we need an algorithm G to generate public/secret key pairs such that it is hard to guess the secret key (a "hard instance generator"). We will formalize two different variants of hard instance generators below (Definitions 25 and 26).

An example of a hard instance generator would be: $R_\eta := \{(x,w) : |w| = \eta \wedge x = f(w)\}$ for some quantum-one-way function f, and G picks w uniformly from $\{0,1\}^\eta$, sets $x := f(w)$, and returns (x,w).

Now a signature is just a proof of knowledge of the secret key. That is, the statement is the public key, and the witness is the secret key. However, a signature should be bound to a particular message. For this, we include the message m in the statement that is proven. That is, the statement that is proven consists of a public key and a message, but the message is ignored when determining whether a given statement has a witness or not. (In the definition below, this is formalized by considering an extended relation R'.) The simulation-soundness of the proof system will then guarantee that a proof/signature with respect to one message cannot be transformed into a proof/signature with respect to another message because this would mean changing the statement.

A signature scheme consists of three oracle algorithms: Keys are generated with $(\mathit{pk}, \mathit{sk}) \leftarrow \mathit{KeyGen}^H(1^\eta)$. The secret key sk is used to sign a message m using the signing algorithm $\sigma \leftarrow \mathit{Sign}^H(1^\eta, \mathit{sk}, m)$ to get a signature σ. And the signature is considered valid iff $\mathit{Verify}^H(1^\eta, \mathit{pk}, \sigma, m) = 1$.

An instance generator for a relation R_η is an algorithm G such that $G(1^\eta)$ outputs $(x,w) \in R_\eta$ with overwhelming probability.

We now describe how to use a simulation-sound zero-knowledge protocol (e.g., Fiat-Shamir) to construct a signature scheme:

Definition 23 (Signatures from non-interactive proofs) *Let G be an instance generator for a relation R_η. Fix a length ℓ_η^m. Let $R'_\eta := \{(x\|m, w) : |m| = \ell_\eta^m \wedge (x,w) \in R_\eta\}$. Let (P,V) be a non-interactive proof system for R'_η (in the random oracle model). Then we construct the signature scheme $(\mathit{KeyGen}, \mathit{Sign}, \mathit{Verify})$ with message space $\{0,1\}^{\ell_\eta^m}$ as follows:*

- *$\mathit{KeyGen}^H(1^\eta)$: Pick $(x,w) \leftarrow G(1^\eta)$. Let $\mathit{pk} := x$, $\mathit{sk} := (x,w)$. Return $(\mathit{pk}, \mathit{sk})$.*
- *$\mathit{Sign}^H(1^\eta, \mathit{sk}, m)$ with $\mathit{sk} = (x,w)$: Run $\sigma \leftarrow P^H(1^\eta, x\|m, w)$. Return σ.*
- *$\mathit{Verify}^H(1^\eta, \mathit{pk}, \sigma, m)$ with $\mathit{pk} = x$: Run $\mathit{ok} \leftarrow V^H(1^\eta, x\|m, \sigma)$. Return ok.*

Note that we use a proof system for the relation R'_η instead of R_η. However, in most cases (including Fiat-Shamir) it is trivial to construct a proof system for R'_η given one for R_η. This is because any sigma-protocol for R_η is also a sigma-protocol for R'_η.[19] The only reason why we

[19] This is made formal by the construction of Σ' in the proof of Corollary 30.

need to use R'_η is that we want to include the message m inside the statement (without logical significance), and R'_η allows us to do precisely that. (In the case of Fiat-Shamir, the overall effect will simply be to include m in the hash, see Definition 29.)

The security property we will prove is strong unforgeability. This definition is standard, we include it here for completeness:

Definition 24 (Strong unforgeability) *A signature scheme* $(\mathit{KeyGen}, \mathit{Sign}, \mathit{Verify})$ *is* strongly unforgeable *iff for all polynomial-time oracle algorithms A there exists a negligible μ such that for all η, we have*

$$\Pr[ok = 1 \ \wedge \ (m^*, \sigma^*) \notin \mathbf{Sig}\text{-}\mathit{queries} : H \leftarrow \mathrm{Fun}(\ell_\eta^{in}, \ell_\eta^{out}), (pk, sk) \leftarrow \mathit{KeyGen}^H(1^\eta), \\ (\sigma^*, m^*) \leftarrow A^{H,\mathbf{Sig}}(1^\eta, pk), ok \leftarrow \mathit{Verify}^H(1^\eta, pk, \sigma^*, m^*)] \leq \mu(\eta).$$

Here **Sig** *is a* classical[20] *oracle that upon classical input m returns $\mathit{Sign}^H(1^\eta, sk, m)$. (But queries to H are quantum.) And* **Sig**-*queries is the list of all queries made to* **Sig**. *(I.e., when* **Sig** *is queried with m and σ, (m, σ) is added to the list* **Sig**-*queries.) And $\ell_\eta^{in}, \ell_\eta^{out}$ denote the input/output length of the random oracle used by the signature scheme.*

In the discussion above, we deferred the definition of hard instance generators. In fact, the definition we have to use depends on the properties of the non-interactive proof system we have. If the proof system has simulation-sound extractability, we can make use of a weaker property, "hard instance generator". But if the proof system has only simulation-soundness, we need a stronger property, "dual-mode hard instance generator". A hard instance generator outputs $(x, w) \in R$ such that it is hard to find a witness for x:

Definition 25 (Hard instance generators) *We call an algorithm G a* hard instance generator *for a fixed-length relation R_η iff*

- *G is quantum-polynomial-time, and*
- *there is a negligible μ such that for every η,* $\Pr[(x, w) \in R_\eta : (x, w) \leftarrow G(1^\eta)] \geq 1 - \mu(\eta)$, *and*
- *for any polynomial-time A, there is a negligible μ such that for every η,* $\Pr[(x, w') \in R_\eta : (x, w) \leftarrow G(1^\eta), w' \leftarrow A(1^\eta, x)] \leq \mu(\eta)$.

As mentioned above, an example of a hard instance generator is: Let $R_\eta := \{(x, w) : |w| = \eta \wedge x = f(w)\}$ for some quantum-one-way function f, and G picks w uniformly from $\{0, 1\}^\eta$, sets $x := f(w)$, and returns (x, w).

In contrast, a dual-mode hard instance generator requires more. While a hard instance generator requires that is it hard to find a witness for x, a dual-mode hard instance generator requires that it is hard to distinguish whether x even has a witness. In other words, we should not be able to distinguish x as returned by G from x^* as returned by an algorithm G^* that returns statements that do not have a witness (except with negligible probability). Formally:

Definition 26 (Dual-mode hard instance generators) *We call an algorithm G a* dual-mode hard instance generator *for a fixed-length relation R_η iff*

- *G is quantum-polynomial-time, and*
- *there is a negligible μ such that for every η,* $\Pr[(x, w) \in R_\eta : (x, w) \leftarrow G(1^\eta)] \geq 1 - \mu(\eta)$, *and*

[20] Formally, this means that **Sig** measures its input at the beginning of the each query.

- *for all quantum-polynomial-time algorithm A, there is a quantum-polynomial-time algorithm G^* and negligible μ_1, μ_2 such that for all η,*

$$\Big|\Pr[b = 1 : (x, w) \leftarrow G(1^\eta),\ b \leftarrow A(1^\eta, x)] - \Pr[b = 1 : x \leftarrow G^*(1^\eta),\ b \leftarrow A(1^\eta, x)]\Big| \le \mu_1(\eta).$$

and

$$\Pr[x \in L_R : x \leftarrow G^*(1^\eta)] \le \mu_2(\eta).$$

Note that we allow G^* to depend on A. This is a slightly weaker requirement than requiring a universal G^*. We chose the weaker variant because it is sufficient for our proof below.

An example of a dual-mode hard instance generator is: Let $R_\eta := \{(x, w) : |w| = \eta \wedge x = F(w)\}$ for some quantum pseudorandom generator $F : \{0,1\}^\eta \to \{0,1\}^{2\eta}$, and G picks w uniformly from $\{0,1\}^\eta$, sets $x := F(w)$, and returns (x, w). The conditions from Definition 26 are satisfied for G^* which returns $x \overset{\$}{\leftarrow} \{0,1\}^{2\eta}$.

With this definition, we can state the two main results of this section, namely the strong unforgeability of signatures constructed from non-interactive zero-knowledge proof systems that are simulation-sound and simulation-sound extractable, respectively:

Theorem 27 (Strong unforgeability from simulation-soundness) *Fix a relation R_η. Let R'_η be defined as in Definition 23. If (P, V) is zero-knowledge and simulation-sound for R'_η, and G is a dual-mode hard instance generator for R_η, then the signature scheme ($KeyGen$, $Sign$, $Verify$) from Definition 23 is strongly unforgeable.*

Theorem 28 (Strong unforgeability from simulation-sound extractability) *If (P, V) is zero-knowledge and has simulation-sound extractability for R'_η, and G is a hard instance generator for R_η, then the signature scheme ($KeyGen$, $Sign$, $Verify$) from Definition 23 is strongly unforgeable.*

The theorems are proven in Section 8.1 and Section 8.2, respectively.

Fiat-Shamir. The two preceding theorems are formulated for generic simulation-sound (extractable) zero-knowledge proof systems. By specializing Theorem 27 to the case that (P, V) is the Fiat-Shamir proof system, we get a signature scheme based on a dual-mode hard instance generator and a zero-knowledge sigma-protocol with special soundness. The resulting signature scheme is the following:

Definition 29 (Fiat-Shamir signatures) *Let G be an instance generator for a relation R_η. Fix a length ℓ^m_η. Then we construct the signature scheme ($KeyGen$, $Sign$, $Verify$) with message space $\{0,1\}^{\ell^m_\eta}$ as follows:*

- *$KeyGen^H(1^\eta)$: Pick $(x, w) \leftarrow G(1^\eta)$. Let $pk := x$, $sk := (x, w)$. Return (pk, sk).*
- *$Sign^H(1^\eta, sk, m)$ with $sk = (x, w)$: $com \leftarrow P^1_\Sigma(1^\eta, x, w)$. $resp \leftarrow P^2_\Sigma(1^\eta, x, w, H(x\|m\|com))$. Return $\sigma := com\|resp$.*
- *$Verify^H(1^\eta, pk, \sigma, m)$ with $pk = x$ and $\sigma = com\|resp$: Run $ok \leftarrow V_\Sigma(1^\eta, x, com, H(x\|m\|com), resp)$. Return ok.*

Corollary 30 (Fiat-Shamir signatures) *Assume that Σ is honest-verifier zero-knowledge, has completeness, has unique responses, has unpredictable commitments, and has perfect special soundness for R_η, and that ℓ^{ch}_η is superlogarithmic. Assume that G is a dual-mode hard instance generator for R_η.*

Then the signature scheme ($KeyGen_{FS}$, $Sign_{FS}$, $Verify_{FS}$) from Definition 29 is strongly unforgeable.

Proof. Let Σ' be the following sigma-protocol for R': The message lengths $\ell_\eta^{com}, \ell_\eta^{ch}, \ell_\eta^{resp}$ are the same as for Σ. For $x \in \{0,1\}^{\ell_\eta^x}$, $m \in \{0,1\}^{\ell_\eta^m}$, the prover $P_{\Sigma'}^1(1^\eta, (x\|m), w)$ runs $P_\Sigma^1(1^\eta, x, w)$, and $P_{\Sigma'}^2(1^\eta, (x\|m), w, ch)$ runs $P_\Sigma^2(1^\eta, x, w, ch)$. And $V_{\Sigma'}(1^\eta, x\|m, com, ch, resp)$ runs $V_\Sigma(1^\eta, x, com, ch, resp)$.

It is easy to check that Σ' is honest-verifier zero-knowledge, has completeness, has unpredictable commitments, has unique responses, and has perfect special soundness for R'_η. (Using the fact that Σ has these properties for R_η.) And ℓ^{ch} is superlogarithmic.

We apply the Fiat-Shamir construction (Definition 17) to Σ'. The resulting proof system (P_{FS}, V_{FS}) is zero-knowledge and simulation-sound for R'_η by Theorems 19 and 21. Then we apply the construction of signatures (Definition 23) to (P_{FS}, V_{FS}) and G. By Theorem 27, the resulting signature scheme S is strongly unforgeable.

Finally, notice that this signature scheme S is the signature scheme from Definition 29. (By explicitly instantiating the constructions from Definition 17 and Definition 23 and the definition of Σ'.) □

If we can prove that Fiat-Shamir is extractable (under suitable conditions on the underlying sigma-protocol), then we get a similar result by combining Theorems 19, 22 and 28.

8.1 Security proof using simulation-soundness

Proof of Theorem 27. In this proof, we will in many places omit the security parameter η for readability. (E.g., we write ℓ^m instead of ℓ_η^m and $Sign(sk, m)$ instead of $Sign(1^\eta, sk, m)$.) It is to be understood that this is merely a syntactic omission, the variables and algorithms still depend on η.

In the following, H will always denote a uniformly random function from $\mathrm{Fun}(\ell^{in}, \ell^{out})$. That is, every game written below implicitly starts with $H \xleftarrow{\$} \mathrm{Fun}(\ell^{in}, \ell^{out})$.

Fix a polynomial-time oracle algorithm A. By definition of strong unforgeability (Definition 24), we need to show

$$\Pr[win = 1 : \text{Game 1}] \le \mu(\eta)$$

for some negligible μ and the following game:

Game 1 (Unforgeability) *$(pk, sk) \leftarrow KeyGen^H()$, $(\sigma^*, m^*) \leftarrow A^{H,\mathbf{Sig}}(pk)$, $ok \leftarrow Verify^H(pk, \sigma^*, m^*)$. $win := (ok = 1 \wedge (m^*, \sigma^*) \notin \mathbf{Sig}$-queries).*

We will transform this game in several steps. First, we inline the definitions of **Sig** (Definition 24) and *KeyGen*, *Sign*, and *Verify* (Definition 23). This leads to the following game:

Game 2 *$(x, w) \leftarrow G(1^\eta)$. $(x^*, \pi^*) \leftarrow B^{H,P^H}(x, w)$. $ok \leftarrow V^H(x^*, \pi^*)$. $win := (ok = 1 \wedge (x^*, \pi^*) \notin$ S-queries).*

Here B is a polynomial-time oracle algorithm that runs A with input $pk := x$, and that, whenever A queries **Sig** with input m, invokes P^H with input $(x\|m, w)$ instead. And when A returns some (m^*, σ^*), then B returns (x^*, π^*) with $x^* := x\|m^*$ and $\pi^* := \sigma^*$. And S-queries is the list of queries made to P^H. More precisely, when P^H is invoked with (x', w') and responds with π', then (x', π') is appended to S-queries.

We then have:

$$\Pr[win = 1 : \text{Game 1}] = \Pr[win = 1 : \text{Game 2}]$$

We now use the zero-knowledge property of (P, V). Let S be the simulator whose existence is guaranteed by Definition 8. Let S' be the oracle that on input $(x, w) \in R'$ runs $S(x)$ and returns the latter's output (as in Definition 8).

Then

$$\Big|\Pr[\mathit{win} = 1 : \text{Game 2}] - \Pr[\mathit{win} = 1 : \text{Game 3}]\Big| \leq \mu_1$$

for some negligible μ_1, and with the following game:

Game 3 $(x, w) \leftarrow G(1^\eta)$. $(x^*, \pi^*) \leftarrow B^{H,S'^H}(x, w)$. $\mathit{ok} \leftarrow V^{H_{\mathit{final}}}(x^*, \pi^*)$. $\mathit{win} := (\mathit{ok} = 1 \land (x^*, \pi^*) \notin \mathsf{S\text{-}queries})$.

Here H_{final} is as in Definition 16, i.e., the value of the random oracle H after it has been reprogrammed by S.

By $x \leq x^*$, we mean that x consists of the first ℓ^x bits of x^*. (I.e., $x^* = x\|m$ for some m.)

Game 4 $(x, w) \leftarrow G(1^\eta)$. $(x^*, \pi^*) \leftarrow B^{H,S'^H}(x, w)$. $\mathit{ok} \leftarrow V^{H_{\mathit{final}}}(x^*, \pi^*)$. $\mathit{win} := (\mathit{ok} = 1 \land x \leq x^* \land (x^*, \pi^*) \notin \mathsf{S\text{-}queries})$.

Since B by construction always outputs $x^* = x\|m^*$, we have

$$\Pr[\mathit{win} = 1 : \text{Game 3}] = \Pr[\mathit{win} = 1 : \text{Game 4}]$$

Let $C^{H,S^H}(x)$ be a polynomial-time oracle algorithm that runs A with input $pk := x$, and that, whenever A queries **Sig** with input m, instead invokes S^H with input $x\|m$. And when A returns some (m^*, σ^*), then C returns (x^*, π^*) with $x^* := x\|m^*$ and $\pi^* := \sigma^*$.

Note that there are two differences between B^{H,S'^H} and C^{H,S^H}: First, C does not take w as input. Second, C invokes S^H instead of S'^H. Since $S'(x\|m, w)$ invokes $S(x\|m)$ whenever $(x\|m, w) \in R'$, B and C will differ only when $(x\|m, w) \notin R'$. By definition of R', this happens only when $(x, w) \notin R$. And this, in turn, happens with negligible probability since (x, w) are chosen by G, and G is a dual-mode hard instance generator. Thus there exists a negligible μ_2 such that

$$\Big|\Pr[\mathit{win} = 1 : \text{Game 4}] - \Pr[\mathit{win} = 1 : \text{Game 5}]\Big| \leq \mu_2$$

with

Game 5 $(x, w) \leftarrow G(1^\eta)$. $(x^*, \pi^*) \leftarrow C^{H,S^H}(x\cancel{, w})$. $\mathit{ok} \leftarrow V^{H_{\mathit{final}}}(x^*, \pi^*)$. $\mathit{win} := (\mathit{ok} = 1 \land x \leq x^* \land (x^*, \pi^*) \notin \mathsf{S\text{-}queries})$.

Since G is a dual-mode hard instance generator, and since the computation in Game 5 after $(x, w) \leftarrow G(1^\eta)$ is quantum-polynomial-time[21] and does not use w, we have (by Definition 26) that there exists a quantum-polynomial-time G^* and a negligible μ_3 such that:

$$\Big|\Pr[\mathit{win} = 1 : \text{Game 5}] - \Pr[\mathit{win} = 1 : \text{Game 6}]\Big| \leq \mu_3$$

with

Game 6 $x \leftarrow G^*(1^\eta)$. $(x^*, \pi^*) \leftarrow C^{H,S^H}(x)$. $\mathit{ok} \leftarrow V^{H_{\mathit{final}}}(x^*, \pi^*)$. $\mathit{win} := (\mathit{ok} = 1 \land x \leq x^* \land (x^*, \pi^*) \notin \mathsf{S\text{-}queries})$.

Since G^* was chosen as in Definition 26, we have that $x \in L_R$ with some negligible probability μ_4 in Game 6. Thus

$$\Big|\Pr[\mathit{win} = 1 : \text{Game 6}] - \Pr[\mathit{win} = 1 : \text{Game 7}]\Big| \leq \mu_4$$

with

[21] Note: to simulate the oracle H (which is a random function and thus has an exponentially large value-table), we use the fact from [Zha12] that a $2q$-wise hash function cannot be distinguished from random by a q-query adversary. This allows us to simulate H using a $2q$-wise hash function for suitable polynomially-bounded q.

Game 7 $x \leftarrow G^*(1^\eta)$. $(x^*,\pi^*) \leftarrow C^{H,S^H}(x)$. $ok \leftarrow V^{H_{final}}(x^*,\pi^*)$. $win := (ok = 1 \ \wedge x \notin L_R \ \wedge \ x \leq x^* \ \wedge \ (x^*,\pi^*) \notin \mathsf{S\text{-}queries})$.

By definition of R', we have that

$$x \notin L_R \wedge x \leq x^* \implies x^* \notin L_R.$$

Thus

$$\Pr[win : \text{Game } 7] \leq \Pr[ok = 1 \wedge x^* \notin L_R \wedge (x^*,\pi^*) \notin \mathsf{S\text{-}queries} : \text{Game } 7].$$

Since (P,V) is simulation-sound (Definition 10), and "$x \leftarrow G^*(1^\eta)$. $(x^*,\pi^*) \leftarrow C^{H,S^H}(x)$" can be executed by a quantum-polynomial-time oracle algorithm with oracle access to H and S^H, we have that there is a negligible μ_5 such that

$$\Pr[ok = 1 \wedge x^* \notin L_R \wedge (x^*,\pi^*) \notin \mathsf{S\text{-}queries} : \text{Game } 7] \leq \mu_5.$$

Combining all inequalities from this proof, we get that

$$\Pr[win : \text{Game } 1] \leq \mu_1 + \cdots + \mu_5 =: \mu.$$

The function μ is negligible since $\mu_1,\dots,\mu_5$ are. Since A was arbitrary and quantum-polynomial-time, and Game 1 is the game from Definition 24, it follows that $(\mathit{KeyGen}, \mathit{Sign}, \mathit{Verify})$ is strongly unforgeable. □

8.2 Security proof using simulation-sound extractability

Proof of Theorem 28. In this proof, we will in many places omit the security parameter η for readability. (E.g., we write ℓ^m instead of ℓ^m_η and $\mathit{Sign}(sk,m)$ instead of $\mathit{Sign}(1^\eta,sk,m)$.) It is to be understood that this is merely a syntactic omission, the variables and algorithms still depend on η.

In the following, H will always denote a uniformly random function from $\mathrm{Fun}(\ell^{in},\ell^{out})$. That is, every game written below implicitly starts with $H \overset{\$}{\leftarrow} \mathrm{Fun}(\ell^{in},\ell^{out})$.

Fix a polynomial-time oracle algorithm A. By definition of strong unforgeability (Definition 24), we need to show

$$\Pr[win = 1 : \text{Game } 1] \leq \mu(\eta)$$

for some negligible μ and the following game:

Game 1 (Unforgeability) $(pk,sk) \leftarrow \mathit{KeyGen}^H()$, $(\sigma^*,m^*) \leftarrow A^{H,\mathbf{Sig}}(pk)$, $ok \leftarrow \mathit{Verify}^H(pk,\sigma^*,m^*)$. $win := (ok = 1 \ \wedge \ (m^*,\sigma^*) \notin \mathbf{Sig}\text{-}\mathit{queries})$.

We will transform this game in several steps. First, we inline the definitions of **Sig** (Definition 24) and KeyGen, Sign, and Verify (Definition 23). This leads to the following game:

Game 2 $(x,w) \leftarrow G(1^\eta)$. $(x^*,\pi^*) \leftarrow B^{H,P^H}(x,w)$. $ok \leftarrow V^H(x^*,\pi^*)$. $win := (ok = 1 \ \wedge \ (x^*,\pi^*) \notin \mathsf{S\text{-}queries})$.

Here B is a polynomial-time oracle algorithm that runs A with input $pk := x$, and that, whenever A queries **Sig** with input m, invokes P^H with input $(x\|m,w)$ instead. And when A returns some (m^*,σ^*), then B returns (x^*,π^*) with $x^* := x\|m^*$ and $\pi^* := \sigma^*$. And $\mathsf{S\text{-}queries}$ is the list of queries made to P^H. More precisely, when P^H is invoked with (x',w') and responds with π', then (x',π') is appended to $\mathsf{S\text{-}queries}$.

We then have:

$$\Pr[win = 1 : \text{Game } 1] = \Pr[win = 1 : \text{Game } 2]$$

We now use the zero-knowledge property of (P, V). Let S be the simulator whose existence is guaranteed by Definition 8. Let S' be the oracle that on input $(x, w) \in R'$ runs $S(x)$ and returns the latter's output (as in Definition 8).

Then

$$\Big|\Pr[win = 1 : \text{Game } 2] - \Pr[win = 1 : \text{Game } 3]\Big| \leq \mu_1$$

for some negligible μ_1, and with the following game:

Game 3 $(x, w) \leftarrow G(1^\eta)$. $(x^*, \pi^*) \leftarrow B^{H,S'^H}(x, w)$. $ok \leftarrow V^{H_{final}}(x^*, \pi^*)$. $win := (ok = 1 \ \wedge \ (x^*, \pi^*) \notin \textsf{S-queries})$.

Here H_{final} is as in Definition 16, i.e., the value of the random oracle H after it has been reprogrammed by S.

By $x \leq x^*$, we mean that x consists of the first ℓ^x bits of x^*. (I.e., $x^* = x\|m$ for some m.)

Game 4 $(x, w) \leftarrow G(1^\eta)$. $(x^*, \pi^*) \leftarrow B^{H,S'^H}(x, w)$. $ok \leftarrow V^{H_{final}}(x^*, \pi^*)$. $win := (ok = 1 \ \wedge \ x \leq x^* \ \wedge \ (x^*, \pi^*) \notin \textsf{S-queries})$.

Since B by construction always outputs $x^* = x\|m^*$, we have

$$\Pr[win = 1 : \text{Game } 3] = \Pr[win = 1 : \text{Game } 4]$$

Let $C^{H,S^H}(x)$ be a polynomial-time oracle algorithm that runs A with input $pk := x$, and that, whenever A queries **Sig** with input m, instead invokes S^H with input $x\|m$. And when A returns some (m^*, σ^*), then C returns (x^*, π^*) with $x^* := x\|m^*$ and $\pi^* := \sigma^*$.

Note that there are two differences between B^{H,S'^H} and C^{H,S^H}: First, C does not take w as input. Second, C invokes S^H instead of S'^H. Since $S'(x\|m, w)$ invokes $S(x\|m)$ whenever $(x\|m, w) \in R'$, B and C will differ only when $(x\|m, w) \notin R'$. By definition of R', this happens only when $(x, w) \notin R$. And this, in turn, happens with negligible probability since (x, w) are chosen by G, and G is a hard instance generator. Thus there exists a negligible μ_2 such that

$$\Big|\Pr[win = 1 : \text{Game } 4] - \Pr[win = 1 : \text{Game } 5]\Big| \leq \mu_2$$

with

Game 5 $(x, w) \leftarrow G(1^\eta)$. $(x^*, \pi^*) \leftarrow C^{H,S^H}(x\cancel{, w})$. $ok \leftarrow V^{H_{final}}(x^*, \pi^*)$. $win := (ok = 1 \ \wedge \ x \leq x^* \ \wedge \ (x^*, \pi^*) \notin \textsf{S-queries})$.

Let X, X' be quantum registers of lengths ℓ^x. Let

$$\rho := \sum_{x'} |x'\rangle\langle x'| \otimes |x'\rangle\langle x'| \cdot \Pr[x = x' : (x, w) \leftarrow G()].$$

be a density operator on X, X'. (Intuitively, ρ represents the state where we picked x using $G()$ and stored the same x in both X and X'.) Let D denote the oracle algorithm that behaves like C, except that it takes two quantum registers X, X' as input, and invokes C with the classical value stored in X as input. (I.e., X is measured and X' is not used.) We have:

$$\Pr[win = 1 : \text{Game } 5] = \Pr[win = 1 : \text{Game } 6]$$

with

Game 6 $X, X' \leftarrow \rho$. $(x^*, \pi^*) \leftarrow D^{H,S^H}(X, X')$. $ok \leftarrow V^{H_{\mathit{final}}}(x^*, \pi^*)$. $\mathit{win} := (ok = 1 \ \wedge \ x \leq x^* \ \wedge \ (x^*, \pi^*) \notin \mathsf{S\text{-}queries})$.

Let F be a purification of D. That is, let F be a pure oracle circuit, and let Z be a quantum register (containing the ancillae for F), such that "$(x^*, \pi^*) \leftarrow D^{H,S^H}(X, X')$" and "$Z \leftarrow |0\rangle$, $x^* \| \pi^* \leftarrow F^{H,S^H}(Z, X, X')$" perform the same operation. Then

$$\Pr[\mathit{win} = 1 : \text{Game 6}] = \Pr[\mathit{win} = 1 : \text{Game 7}]$$

with

Game 7 $X, X', Z \leftarrow \rho \otimes |0\rangle\langle 0|$. $x^* \| \pi^* \leftarrow F^{H,S^H}(Z, X, X')$. $ok \leftarrow V^{H_{\mathit{final}}}(x^*, \pi^*)$. $x \leftarrow \mathcal{M}(X')$. $\mathit{win} := (ok = 1 \ \wedge \ x \leq x^* \ \wedge \ (x^*, \pi^*) \notin \mathsf{S\text{-}queries})$.

Here $x \leftarrow \mathcal{M}(X')$ represents a measurement of X' in the computational basis, with outcome assigned to x. Note that F does not use the register X'.

Let Π^H be a projective measurement circuit on Z, X, X' such that $\Pi^H_{x^* \| \pi^*}$ measures whether X' contains a prefix of x^*. That is, the projector corresponding to outcome 1 of $\Pi^H_{x^* \| \pi^*}$ is $I_Z \otimes I_X \otimes |x\rangle\langle x|_{X'}$ for the unique $x \in \{0,1\}^{\ell^x}$ with $x \leq x^*$. ($\Pi^H_{x^* \| \pi^*}$ does not use query H.)

Then we can rewrite Game 7 as:

Game 8 $X, X', Z \leftarrow \rho \otimes |0\rangle\langle 0|$. $x^* \| \pi^* \leftarrow F^{H,S^H}(Z, X, X')$. $ok \leftarrow V^{H_{\mathit{final}}}(x^*, \pi^*)$. ~~$x \leftarrow \mathcal{M}(X')$.~~ $ok_A \leftarrow \Pi^{H_{\mathit{final}}}_{x^* \| \pi^*}(Z, X, X')$. $\mathit{win} := (ok = 1 \ \wedge \ ok_A = 1 \ \wedge \ (x^*, \pi^*) \notin \mathsf{S\text{-}queries})$.

And we have

$$\Pr[\mathit{win} = 1 : \text{Game 7}] = \Pr[\mathit{win} = 1 : \text{Game 8}].$$

Now we can use the simulation-sound extractability of (P, V) for the relation R'. By Definition 16, we have

$$\Pr[\mathit{win} = 1 : \text{Game 9}] \geq \tfrac{1}{p} \Pr[\mathit{win} = 1 : \text{Game 8}]^d - \mu$$

for some polynomial-time oracle algorithm E, some polynomials $p > 0, \ell \leq 0$, some negligible μ, and some constant $d > 0$ and with the following game:

Game 9 $X, X', Z \leftarrow \rho \otimes |0\rangle\langle 0|$. $(x^*, w^*, \pi^*, \mathit{ass}) \leftarrow E^{F^{\mathbf{rew}}(X,X',Z), \Pi^{\mathbf{oracle}, H, \ell}(X,X',Z), H}(\ell, \mathbf{shape}_F)$. $ok_A \leftarrow \Pi^{H(\mathit{ass})}_{x^* \| \pi^*}(X, X', Z)$. $\mathit{win} := \big((x^*, w^*) \in R' \wedge ok_A = 1\big)$.

From this we get

$$\Pr[\mathit{win} = 1 : \text{Game 8}] \leq \sqrt[d]{p \cdot \Pr[\mathit{win} = 1 : \text{Game 9}] + \mu_3}.$$

By definition of ρ, we can rewrite $X, X', Z \leftarrow \rho \otimes |0\rangle$ as follows:

Game 10 $(x, w) \leftarrow G()$, $X \leftarrow |x\rangle$, $X' \leftarrow |x\rangle$, $Z \leftarrow |0\rangle$. $(x^*, w^*, \pi^*, \mathit{ass}) \leftarrow E^{F^{\mathbf{rew}}(X,X',Z), \Pi^{\mathbf{oracle}, H, \ell}(X,X',Z), H}(\ell, \mathbf{shape}_{F_\eta})$. $ok_A \leftarrow \Pi^{H(\mathit{ass})}_{x^* \| \pi^*}(X, X', Z)$. $\mathit{win} := \big((x^*, w^*) \in R' \wedge ok_A = 1\big)$.

We have

$$\Pr[\mathit{win} = 1 : \text{Game 9}] = \Pr[\mathit{win} = 1 : \text{Game 10}].$$

Note that E cannot directly access the quantum register X' (it is part of the state of $F^{\mathbf{rew}}$ and $\Pi^{\mathbf{rew}}$). $F^{\mathbf{rew}}$ does not access X'. So X' is only accessed by applications of $\Pi^{\mathbf{rew}}$. However, Π

is a measurement in the computational basis. Thus, if X' is in state $|x\rangle$ before the invocation of E, it will also be in state $|x\rangle$ after the invocation of E. Thus the measurement $\Pi^{H(ass)}_{x^*\|\pi^*}(X, X', Z)$ will return 1 iff $x \leq x^*$. Thus

$$\Pr[\mathit{win} = 1 : \text{Game 10}] = \Pr[\mathit{win} = 1 : \text{Game 11}]$$

with

Game 11 $(x, w) \leftarrow G()$, $X \leftarrow |x\rangle$, $X' \leftarrow |x\rangle$, $Z \leftarrow |0\rangle$. $(x^*, w^*, \pi^*, \mathit{ass}) \leftarrow E^{F^{\mathbf{rew}}(X,X',Z), \Pi^{\mathbf{oracle},H,\ell}(X,X',Z), H}(\ell, \mathbf{shape}_F)$. $\mathit{ok}_A := (x \leq x^*)$. $\mathit{win} := \big((x^*, w^*) \in R' \wedge \mathit{ok}_A = 1\big)$.

If $\mathit{win} = 1$, then $(x^*, w^*) \in R'$ and $x \leq x^*$. By definition of R', this implies $(x, w^*) \in R$. Thus we have

$$\Pr[\mathit{win} = 1 : \text{Game 11}] \leq \Pr[(x, w^*) \in R : \text{Game 11}].$$

And finally, since G is a hard instance generator for R, and all steps in the game $(x, w) \leftarrow G()$ can be simulated by a quantum-polynomial-time algorithm,[22] we have that

$$\Pr[(x, w^*) \in R : \text{Game 11}] \leq \mu_4$$

for some negligible μ_4.

Collecting all inequalities and equalities from this proof, we get:

$$\Pr[\mathit{win} = 1 : \text{Game 1}] \leq \sqrt[d]{p\,\mu_4 + \mu_3} + \mu_1 + \mu_2 =: \mu.$$

Since $d > 0$ is a constant, p is a polynomial, and $\mu_1, \dots, \mu_4$ are negligible, we have that μ is negligible. Thus we have showed that the probability of an attack against the strong unforgeability game is negligible, it follows that $(\mathit{KeyGen}, \mathit{Sign}, \mathit{Verify})$ is strongly unforgeable (Definition 24). □

Acknowledgments. I thank Andris Ambainis, and Ali El Kaafarani for valuable discussions, and Alexander Belov for breaking the Quantum Forking Conjecture upon which earlier versions of this work were based. This work was supported by institutional research funding IUT2-1 of the Estonian Ministry of Education and Research, the Estonian ICT program 2011-2015 (3.2.1201.13-0022), and by the Estonian Centre of Exellence in IT (EXCITE) funded by ERDF.

A Problems with concurrent executions of Fiat-Shamir

This section given an example of the intricacies of Fiat-Shamir when used concurrently in a protocol. Everything in this section applies both to the classical and the quantum case.

Construction of the sigma-protocol. Consider a fixed-length relation R_0. Assume that for any pk, there is at most one sk with $(pk, sk) \in R_0$.[23] Define the fixed-length relation R such that $(pk_{10}\|pk_{11}\|\dots\|pk_{\eta 0}\|pk_{\eta 1},\ sk_0\|sk_1) \in R$ iff $(pk_{i0}, sk_0) \in R \wedge (pk_{i1}, sk_1) \in R$ for some i.

[22] Note that a naive implementation of the game involves choosing H, which would take exponential space and time. However, [Zha12] showed that we can use a $2q$-wise independent function instead, where q is the number of H-queries performed by the rest of the game. This leads to a quantum-polynomial-time implementation.

[23] If we drop this requirement, the only change will be that the sigma-protocol Σ' constructed below does not have unique responses. And example of a suitable relation would be $(x, w) \in R_0$ iff $x = f(w)$ for some injective quantum-one-way permutation f

Let $\Sigma = (\ell^{com}_\eta, \ell^{ch}_\eta, \ell^{resp}_\eta, P^1_\Sigma, P^2_\Sigma, V_\Sigma)$ be a sigma protocol for R that satisfies all conditions from Definition 6. We assume for simplicity that $\ell^{ch} = \eta$ and $\ell^{resp} \geq \eta \cdot |sk_i|$. We construct a new sigma-protocol $\Sigma' := (\ell^{com}_\eta, \ell^{ch}_\eta, \ell^{resp}_\eta, P^1_\Sigma, P^2_\Sigma, V'_\Sigma)$ for R (i.e., everything is the same except the definition of the verifier) with the following $V'_\Sigma(x, com, ch, resp)$:

- If $com \neq 0^{\ell^{com}}$, invoke $ok \leftarrow V_\Sigma(x, com, ch, resp)$ and return ok.
- If $com = 0^{\ell^{com}}$:
  - Parse x as $x = pk_{10} \| pk_{11} \| \dots \| pk_{\eta 0} \| pk_{\eta 1}$, and parse $resp$ as $resp = sk_1 \| \dots \| sk_\eta \| 0^{\ell^{resp} - \eta |sk_1|}$.
  - Check whether $(pk_{ich_i}, sk_i) \in R_0$ for all $i = 1, \dots, \eta$.
  - If the parsing and the check succeeded, return 1.

Properties of the sigma-protocol. The sigma-protocol Σ' still has all properties from Definition 6:

- Completeness: the output of V'_Σ differs from V_Σ only when $com = 0^{\ell^{com}}$. This happens with negligible probability in the game defining completeness since Σ has unpredictable commitments. Thus the success probability in the definition of completeness for Σ and for Σ' differs only by a negligible amount.
- Perfect special soundness: We define the extractor $E'_\Sigma(x, com, ch, resp, ch', resp')$ as follows: If $com \neq 0^{\ell^{com}}$, E'_Σ invokes E_Σ. If $com = 0^{\ell^{com}}$: E'_Σ parses $resp =: sk_1 \| \dots \| sk_n \| 0^{\ell^{resp} - n|sk_1|}$ and $resp' =: sk'_1 \| \dots \| sk'_\eta \| 0^{\ell^{resp} - \eta |sk_1|}$. It finds an index i such that $ch_i = 0, ch'_i = 1$ or $ch_i = 1, ch'_i = 0$. Then it returns the witness $sk_i \| sk'_i$ or $sk'_i \| sk_i$, respectively. It is easy to verify that this extractor satisfies the definition of perfect special soundness (since Σ has perfect special soundness).
- Honest-verifier zero-knowledge and unpredictable commitments: These properties do not depend on the verifier, so they follow immediately from the corresponding properties of Σ.
- Perfectly unique responses: Perfectly unique responses for the case $com \neq 0^{\ell^{com}}$ follows from the perfectly unique responses of Σ. Assume that there are $x, com, ch, resp, resp'$ with $com = 0^{\ell^{com}}$ and $V'_\Sigma(\eta, x, com, ch, resp) = 1$ and $V'_\Sigma(\eta, x, com, ch, resp') = 1$. By definition of V'_Σ, this implies that $x = pk_{10} \| pk_{11} \| \dots \| pk_{\eta 0} \| pk_{\eta 1}$ and $resp = sk_1 \| \dots \| sk_n \| 0^{\ell^{resp} - n|sk_1|}$ and $resp' = sk'_1 \| \dots \| sk'_n \| 0^{\ell^{resp} - n|sk'_1|}$ and $resp \neq resp'$ and $(pk_{ich_i}, sk_i) \in R_0$ and $(pk_{ich_i}, sk'_i) \in R_0$. Since $resp \neq resp'$, there is an i such that $sk_i \neq sk'_i$. Thus with $pk := pk_{ich_i}$, $sk := sk_i$, $sk' := sk'_i$, we have $(pk, sk), (pk, sk') \in R_0$, in contradiction to the assumptions we made about R_0 in the beginning of this section.

Summarizing, Σ' is a reasonable (in terms of its security properties) if somewhat artificial sigma-protocol for the relation R'. We would thus assume that Fiat-Shamir based on Σ' is a good non-interactive proof system. Yet the following example illustrates that this is not always the case.

Toy protocol. Consider the following toy protocol Π that uses a non-interactive proof system (P, V) for the relation R:

- Let G be a hard instance generator for R_0 (see Definition 25). Bob generates 2η key pairs $(pk_{ib}, sk_{ib}) \leftarrow G(1^\eta)$ with $i = 1, \dots, \eta$ and $b = 0, 1$. Then Bob sends $x = pk_{10} \| pk_{11} \| \dots \| pk_{\eta 0} \| pk_{\eta 1}$ to Alice.
- Alice picks bits $b_1, \dots, b_\eta \in \{0, 1\}$ and sends them to Bob.
- Bob sends sk_{ib_i} for all $i = 1, \dots, \eta$ to Alice.
- Then Bob expects a proof π from Alice.[24]

[24] Honest Alice will not send such a proof. However, as we will see, dishonest Alice can send a valid proof here.

- Bob checks using P whether π is a valid proof for the statement x with respect to the relation R. (I.e., π is interpreted as a proof that Alice knows both secret keys (sk_{i0}, sk_{i1}) for at least one index i.) Formally, Bob runs $ok \leftarrow V(1^\eta, x, \pi)$.

Intuitively, we would expect that for honest Bob (and polynomial-time Alice), we will have $ok = 0$ with overwhelming probability, at least if (P, V) is extractable (an argument of knowledge). This is because an extractable proof of knowledge should (intuitively!) guarantee that Alice can produce a valid proof π only when she knows (sk_{i0}, sk_{i1}) for at least one index i. But she will never know such (sk_{i0}, sk_{i1}) because she is given only one sk_{ib} for each i.

Unfortunately, this intuition is not correct (at least in the case of Fiat-Shamir):

Attack. Assume that (P, V) is constructed by applying the Fiat-Shamir construction to the sigma-protocol Σ'. Then a malicious Alice can perform the following attack against the toy protocol:

- Alice receives x.
- Alice sets $com := 0^{\ell^{com}}$ and computes $ch := H(x\|com)$.
- Alice sets $b_i := ch_i$ and sends the bits $b_1, \dots, b_\eta$ to Bob.
- Bob sends $sk_{ib_i} = sk_{ich_i}$ to Alice for all i.
- Alice computes $resp := sk_{1ch_1}\| \dots \|sk_{\eta ch_\eta}\|0^*$. (Here 0^* is a zero-string of the expected length.)
- Alice sends $\pi := com\|resp$.
- Bob runs $ok \leftarrow V(1^\eta, x, \pi)$. By definition of $V = V_{FS}$ (Figure 1), this leads to the following computation:
  - V computes $com\|resp := \pi$ and $ch := H(x\|com)$. That is, V has the same $com, ch, resp$ as Alice.
  - V invokes $V'_\Sigma(1^\eta, x, com, ch, resp)$. This leads to the following steps by definition of Σ':
    - $*$ x is parsed as $x = pk_{10}\|pk_{11}\| \dots \|pk_{\eta 0}\|pk_{\eta 1}$. These are the same pk_{ib} as chosen by Bob. $resp$ is parsed as $sk_1\| \dots \|sk_\eta\|0^*$. Then $sk_i = sk_{ich_i}$ by definition of $resp$.
    - $*$ V'_Σ checks whether $(pk_{ich_i}, sk_i) \in R_0$ for all $i = 1, \dots, \eta$. Since $sk_i = sk_{ich_i}$, and all (pk_{ib}, sk_{ib}) pairs were chosen by a hard instance generator for R_0 (and are thus valid key pairs with respect to R_0 with overwhelming probability), we have that $(pk_{ich_i}, sk_i) \in R_0$ with overwhelming probability for all i.
    - $*$ Thus $ok = 1$ with overwhelming probability.

Note that this attack makes specific use of the design of Fiat-Shamir. For example, it is easy to see that schemes with online-extractability such as the ones from Fischlin [Fis05] (in the classical case) and Unruh [Unr15] (in the quantum case) indeed satisfy the intuition that $ok = 0$ with overwhelming probability in the toy protocol.

One can easily imagine a more realistic protocol in which the proof π also serves an honest purpose (e.g., it may depend on Alice's authorization whether she can produce such a proof); in order to keep the example minimal we opted not to include such extensions.

Index

Symbol index

ℓ_η^w	Length of witnesses in the relation R_η	12
ℓ_η^x	Length of statements in the relation R_η	12
$\mathcal{E}_C^{\mathcal{O}_1,\dots,\mathcal{O}_n}$	Superoperator implemented by pure oracle circuit C	8
ℓ_η^{resp}	Length of responses $resp$	13
ℓ_η^{ch}	Length of challenges ch	13
ℓ_η^{com}	Length of commitments com	13
$\{0,1\}^n$	Bitstrings of length n	
P_{FS}	Fiat-Shamir prover	21
V_{FS}	Fiat-Shamir verifier	21
S_{FS}	Fiat-Shamir simulator	22
η	Security parameter	
$E_\Sigma(x, com, ch, resp, ch', resp')$	Special soundness extractor for sigma protocol Σ	13
S_Σ	Honest-verifier simulator extractor for sigma protocol Σ	13
$H(ass)$	H updated with assignment-list ass	7
$\mathbb{R}$	Real numbers	
$\mathcal{E}$	A quantum operation	
$\mathbb{C}$	Complex numbers	
$\mathcal{H}$	A Hilbert space	
$\mathcal{O}$	An oracle	
$\mathbf{op}_C$	Execution schedule of pure oracle circuit C	
$\mathcal{H}_{state}$	State space of a pure oracle circuit	
call_i	Instruction that oracle circuit invokes oracle i in this step	
$\mathsf{compute}$	Instruction that oracle circuit computes in this step	
$\mathbf{U}_H$	Unitary evaluating the function H, $\mathbf{U}_H\lvert x,y\rangle = \lvert x, y \oplus H(x)\rangle$	8
$\mathcal{U}_C$	Unitary performed by circuit C in each step	
$\lvert\Psi\rangle$	Vector in a Hilbert space (usually a quantum state)	
$\langle\Psi\rvert$	Conjugate transpose of $\lvert\Psi\rangle$	
$x \leftarrow A$	x is assigned the output of algorithm A	7
$x \xleftarrow{\$} S$	x chosen uniformly from set S/according to distribution S	7
$\Pr[P : G]$	Probability of P after G	7
I	Identity	7
$A^{\mathbf{rew}}$	Pure oracle circuit A as a rewindable oracle	
$M^{\mathbf{oracle},H,\ell}$	Oracle access to projective measurement circuit M^H, with $\leq \ell$ assignments to H	11
$\mathrm{Fun}(n,m)$	Set of functions $\{0,1\}^n \to \{0,1\}^m$	14
$U^{\mathbf{rewind}}$	Rewindable version of unitary U	10
$F(\sigma,\rho)$	Fidelity	7
$\mathbb{N}$	Natural numbers (including 0)	
$\operatorname{im} A$	Image of function/operator A	
$\mathit{Verify}(pk,\sigma,m)$	Verifies a signature σ on message m using public key pk	33
$\mathit{Sign}(sk,m)$	Produces a signature on m using signing key sk	33
$\mathit{KeyGen}()$	Produces a public/secret key pair	33
sk	Secret key	
pk	Public key	
P_Σ^2	Second round of prover of sigma protocol Σ	13
P_Σ^1	First round of prover of sigma protocol Σ	13
V_Σ	Verifier of sigma protocol Σ	13
$\mathit{Sign}_{FS}(sk,m)$	Signing algorithm of Fiat-Shamir signature scheme	
$\mathit{KeyGen}_{FS}()$	Key generation of Fiat-Shamir signature scheme	

References

[Adi08] Ben Adida. Helios: Web-based open-audit voting. In Paul C. van Oorschot, editor, *USENIX Security Symposium 08*, pages 335–348. USENIX, 2008. Online at `http://www.usenix.org/events/sec08/tech/full_papers/adida/adida.pdf`.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems (the hardness of quantum rewinding). In *FOCS 2014*, pages 474–483. IEEE, October 2014.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Crypto 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, 2004.

[BCC04] Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *ACM CCS '04*, pages 132–145, New York, NY, USA, 2004. ACM.

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *ASIACRYPT 2011*, pages 41–69, Berlin, Heidelberg, 2011. Springer-Verlag.

[BG93] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Crypto '92*, number 740 in LNCS, pages 390–420. Springer, 1993. Extended version online available at `http://www-cse.ucsd.edu/users/mihir/papers/pok.ps`.

[BK16] Rachid El Bansarkhani and Ali El Kaafarani. Post-quantum attribute-based signatures from lattice assumptions. IACR ePrint 2016/823, 2016.

[BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to Helios. In *Asiacrypt 2012*, volume 7658 of *LNCS*, pages 626–643. Springer, 2012.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73. ACM, 1993.

[BrOP16] Carsten Baum, Ivan Damgård, Sabine Oechsner, and Chris Peikert. Efficient commitments and zero-knowledge protocols from ring-SIS with applications to lattice-based threshold cryptosystems. IACR ePrint https://eprint.iacr.org/2016/997, 2016.

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *Eurocrypt 2001*, pages 93–118. Springer, 2001.

[Fis05] Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In *Crypto 2005*, volume 3621 of *LNCS*, pages 152–168. Springer, 2005.

[FKMV12] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the Fiat-Shamir transform. In *INDOCRYPT 2012*, volume 7668 of *LNCS*, pages 60–79. Springer, 2012. Preprint is IACR ePrint 2012/704.

[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, Proceedings of CRYPTO '86*, number 263 in Lecture Notes in Computer Science, pages 186–194. Springer-Verlag, 1987.

[GCZ16] Steven Goldfeder, Melissa Chase, and Greg Zaverucha. Efficient post-quantum zero-knowledge and signatures. IACR ePrint 2016/1110, 2016.

[GKV10] S. Dov Gordon, Jonathan Katz, and Vinod Vaikuntanathan. A group signature scheme from lattice assumptions. In *Asiacrypt 2010*, volume 6477, pages 395–412. Springer, 2010.

[HRS16] Andreas Hülsing, Joost Rijneveld, and Fang Song. *Mitigating Multi-target Attacks in Hash-Based Signatures*, pages 387–416. Springer, 2016.

[LLM$^+$16a] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In *Asiacrypt 2016*, volume 10032 of *LNCS*, pages 373–403. Springer, 2016. Full version IACR ePrint 2016/101.

[LLM$^+$16b] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In *Asiacrypt 2016*, volume 10032 of *LNCS*, pages 101–131. Springer, 2016.

[LNW15] San Ling, Khoa Nguyen, and Huaxiong Wang. Group signatures from lattices: Simpler, tighter, shorter, ring-based. In *PKC 2015*, volume 9020, pages 427–449. Springer, 2015.

[PS96a] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In *Asiacrypt 1996*, volume 1163 of *LNCS*, pages 252–265. Springer, 1996.

[PS96b] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *Eurocrypt 96*, volume 1070 of *LNCS*, pages 387–398. Springer, 1996.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS '99*. IEEE, 1999.

[Sch91] Claus-Peter Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[SG02] Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *J Cryptology*, 15(2):75–96, 2002.

[Unr12] Dominique Unruh. Quantum proofs of knowledge. In *Eurocrypt 2012*, volume 7237 of *LNCS*, pages 135–152. Springer, April 2012. Preprint on IACR ePrint 2010/212.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *Eurocrypt 2015*, volume 9057, pages 755–784. Springer, 2015. IACR ePrint 2014/587.

[Zha12] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In *Crypto 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, 2012.