Fiat-Shamir Transformation of Multi-Round Interactive Proofs

Thomas Attema[1,3,4,⋆], Serge Fehr[1,3,⋆⋆], and Michael Klooß[2,⋆⋆⋆]

[1] CWI, Cryptology Group, Amsterdam, The Netherlands
[2] Karlsruhe Institute of Technology, KASTEL, Karlsruhe, Germany
[3] Leiden University, Mathematical Institute, Leiden, The Netherlands
[4] TNO, Cyber Security and Robustness, The Hague, The Netherlands

Abstract. The celebrated Fiat-Shamir transformation turns any public-coin interactive proof into a non-interactive one, which inherits the main security properties (in the random oracle model) of the interactive version. While originally considered in the context of 3-move public-coin interactive proofs, i.e., so-called Σ-protocols, it is now applied to multi-round protocols as well. Unfortunately, the security loss for a $(2\mu+1)$-move protocol is, in general, approximately Q^μ, where Q is the number of oracle queries performed by the attacker. In general, this is the best one can hope for, as it is easy to see that this loss applies to the μ-fold sequential repetition of Σ-protocols, but it raises the question whether certain (natural) classes of interactive proofs feature a milder security loss.

In this work, we give positive and negative results on this question. On the positive side, we show that for $(k_1,\ldots,k_\mu)$-special-sound protocols (which cover a broad class of use cases), the knowledge error degrades linearly in Q, instead of Q^μ. On the negative side, we show that for t-fold *parallel repetitions* of typical $(k_1,\ldots,k_\mu)$-special-sound protocols with $t \geq \mu$ (and assuming for simplicity that t and Q are integer multiples of μ), there is an attack that results in a security loss of approximately $\frac{1}{2}Q^\mu/\mu^{\mu+t}$.

1 Introduction

1.1 Background and State of the Art

The celebrated and broadly used Fiat-Shamir transformation turns any public-coin interactive proof into a *non-interactive* proof, which inherits the main security properties (in the random oracle model) of the interactive version. The rough idea is to replace the random challenges, which are provided by the verifier in the interactive version, by the hash of the current message (concatenated with the messages from previous rounds). By a small adjustment, where also

⋆ thomas.attema@tno.nl
⋆⋆ serge.fehr@cwi.nl
⋆⋆⋆ michael.klooss@kit.edu

the to-be-signed message is included in the hashes, the transformation turns any public-coin interactive proof into a signature scheme. Indeed, the latter is a commonly used design principle for constructing very efficient signature schemes.

While originally considered in the context of 3-move public-coin interactive proofs, i.e., so-called Σ-protocols, the Fiat-Shamir transformation also applies to *multi-round* protocols. However, a major drawback in the case of multi-round protocols is that, in general, the security loss obtained by applying the Fiat-Shamir transformation grows exponentially with the number of rounds. Concretely, for any $(2\mu+1)$-move interactive proof Π (where we may assume that the prover speaks first and last, so that the number of communication rounds is indeed odd) that admits a cheating probability of at most ϵ, captured by the knowledge or soundness error, the Fiat-Shamir-transformed protocol $\mathsf{FS}[\Pi]$ admits a cheating probability of (approximately) at most $Q^\mu \cdot \epsilon$, where Q denotes the number of random-oracle queries admitted to the dishonest prover. A tight reduction is due to [11] with a security loss $\binom{Q}{\mu} \approx \frac{Q^\mu}{\mu^\mu}$, where the approximation holds whenever μ is much smaller than Q, which is the typical case. More concretely, [11] introduces the notions of *state-restoration soundness (SRS)* and *state-restoration knowledge (SRK)*, and it shows that any (knowledge) sound protocol Π satisfies these notions with the claimed security loss.[1] The security of $\mathsf{FS}[\Pi]$ (with the same loss) then follows from the fact that these soundness notions imply the security of the Fiat-Shamir transformation.

Furthermore, there are (contrived) examples of multi-round protocols Π for which this Q^μ security loss is almost tight. For instance, the μ-fold sequential repetition Π of a special-sound Σ-protocol with challenge space $\mathcal{C}$ is ϵ-sound with $\epsilon = \frac{1}{|\mathcal{C}|^\mu}$, while it is easy to see that, by attacking the sequential repetitions round by round, investing Q/μ queries per round to try to find a "good" challenge, and assuming $|\mathcal{C}|$ to be much larger than Q, its Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ can be broken with probability approximately $\left(\frac{Q}{\mu}\frac{1}{|\mathcal{C}|}\right)^\mu = \frac{Q^\mu}{\mu^\mu} \cdot \epsilon$.[2]

For μ beyond 1 or 2, let alone for non-constant μ (e.g., for IOP-based protocols [11,3,10] and also Bulletproofs-like protocols [13,15]), this is a very unfortunate situation when it comes to choosing concrete security parameters. If one wants to rely on the proven security reduction, one needs to choose a large security parameter for Π, in order to compensate for the order Q^μ security loss, effecting its efficiency; alternatively, one has to give up on proven security and simply *assume* that the security loss is much milder than what the general bound suggests. Often, the security loss is simply ignored.

This situation gives rise to the following question: *Do there exist natural classes of multi-round public-coin interactive proofs for which the security loss behaves more benign than what the general reduction suggests?* Ideally, the general Q^μ loss appears for contrived examples *only*.

[1] As a matter of fact, [11] considers arbitrary *interactive oracle proofs (IOPs)*, but these notions are well-defined for ordinary interactive proofs too.

[2] This is clearly a contrived example since the natural construction would be to apply the Fiat-Shamir transformation to the *parallel* repetition of the original Σ-protocol, where no such huge security loss would then occur.

So far, the only positive results, establishing a security loss linear in Q, were established in the context of *straight-line/online* extractors that do not require rewinding. These extractors either rely on the algebraic group model (AGM) [22], or are restricted to protocols using hash-based commitment schemes in the random oracle model [11]. To analyze the properties of straight-line extractors, new auxiliary soundness notions were introduced: *round-by-round (RBR) soundness* [17] and *RBR knowledge* [18]. However, it is unclear if and how these notions can be used in scenarios where straight-line extraction does not apply.

In this work, we address the above question (in the plain random-oracle model, and without restricting to schemes that involve hash-based commitments), and give both positive and negative answers, as explained in more detail below.

1.2 Our Results

Positive Result. We show that the Fiat-Shamir transformation of any $(k_1, \ldots, k_\mu)$-special-sound interactive proof has a security loss of at most $Q + 1$. More concretely, we consider the *knowledge error* κ as the figure of merit, i.e., informally, the maximal probability of the verifier accepting the proof when the prover does not have a witness for the claimed statement, and we prove the following result, also formalized in the theorem below. For any $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-move interactive proof Π with knowledge error κ (which is a known function of $(k_1, \ldots, k_\mu)$), the Fiat-Shamir transformed protocol $\mathsf{FS}[\Pi]$ has a knowledge error at most $(Q + 1) \cdot \kappa$. This result is directly applicable to a long list of recent zero-knowledge proof systems, e.g., [13,15,29,26,16,4,6,12,21]. While all these works consider the Fiat-Shamir transformation of special-sound protocols, most of them ignore the associated security loss.

Main Theorem (Theorem 2)**.** *Let Π be a $(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$ special-sound interactive proof with knowledge error κ. Then the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of Π is knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q + 1) \cdot \kappa\,.$$

Since in the Fiat-Shamir transformation of any $(2\mu + 1)$-move protocol Π, a dishonest prover can simulate any attack against Π, and can try Q/μ times when allowed to do Q queries in total, our new upper bound $(Q + 1) \cdot \kappa$ is close to the trivial lower bound $1 - (1 - \kappa)^{Q/\mu} \approx Q\kappa/\mu$. Another, less explicit, security measure in the context of knowledge soundness is the run time of the knowledge extractor. Our bound on the knowledge error holds by means of a knowledge extractor that makes an expected number of $K + Q \cdot (K - 1)$ queries, where $K = k_1 \cdots k_\mu$. This is a natural bound: K is the number of necessary distinct "good" transcripts (which form a certain tree-like structure). The loss of $Q \cdot (K - 1)$ captures the fact that a prover may finish different proofs, depending on the random oracle answers, and only one out of Q proofs may be useful for extraction, as explained below.

Our result on the *knowledge* soundness of $\mathsf{FS}[\Pi]$ for special-sound protocols Π immediately carries over to *ordinary* soundness of $\mathsf{FS}[\Pi]$, with the same security loss $Q+1$. However, proving knowledge soundness is more intricate; showing a linear-in-Q loss for ordinary soundness can be obtained via simpler arguments (e.g., there is no need to argue efficiency of the extractor).

The construction of our knowledge extractor is motivated by the extractor from [5] in the interactive case, but the analysis here in the context of a non-interactive proof is much more involved. We analyze the extractor in an inductive manner, and capture the induction step (and the base case) by means of an abstract experiment. The crucial idea for the analysis (and extractor) is how to deal with accepting transcripts which are not useful.

To see the core problem, consider a Σ-protocol, i.e., a 3-move k-special-sound interactive proof, and a semi-honest prover that knows a witness and behaves as follows. It prepares, independently, Q first messages $a^1, \ldots, a^Q$ and asks for all hashes $c^i = \mathsf{RO}(a^i)$, and then decides "randomly" (e.g., using a hash over all random oracle answers) which thread to complete, i.e., for which i^* to compute the response z and then output the valid proof (a^{i^*}, z). When the extractor then reprograms the random oracle at the point a^{i^*} to try to obtain another valid response but now for a different challenge, this affects i^*, and most likely the prover will then use a different thread j^* and output the proof (a^{j^*}, z') with $a^{j^*} \neq a^{i^*}$. More precisely, $\Pr(j^* = i^*) = 1/Q$. Hence, an overhead of Q appears in the run-time.

In case of an *arbitrary* dishonest prover with an unknown strategy for computing the a^i's above, and with an arbitrary (unknown) success probability ϵ, the intuition remains: after reprogramming, we still expect $\Pr(j^* = i^*) \geq 1/Q$ and thus a linear-in-Q overhead in the run-time of the extractor. However, providing a rigorous proof is complicated by the fact that the event $j^* = i^*$ is not necessarily independent of the prover producing a *valid* proof (again) after the reprogramming. Furthermore, conditioned on the prover having been successful in the first run and conditioned on the corresponding i^*, the success probability of the prover after the reprogramming may be skewed, i.e., may not be ϵ anymore. As a warm-up for our general multi-round result, we first give a rigorous analysis of the above case of a Σ-protocol. For that purpose, we introduce an abstract sampling game that mimics the behavior of the extractor in finding two valid proofs with $j^* = i^*$, and we bound the success probability and the "cost" (i.e., the number of samples needed) of the game, which directly translate to the success probability and the run-time of the extractor.

Perhaps surprisingly, when moving to *multi-round* protocols, dealing with the knowledge error is relatively simple by recursively composing the extractor for the Σ-protocol. However, controlling the run-time is intricate. If the extractor is recursively composed, i.e., it makes calls to a sub-extractor to obtain a sub-tree, then a naive construction and analysis gives a blow-up of Q^μ in the run-time. Intuitively, because only $1/Q$ of the sub-extractor runs produce useful sub-trees, i.e., sub-trees which extend the current a^{i^*}. The other trees belong to some a^{j^*}

with $j^* \neq i^*$ and are thus useless. This overhead of Q then accumulates per round.

The crucial observation that we exploit in order to overcome the above issue is that the very first (accepting) transcript sampled by a sub-extractor already determines whether a sub-tree will be (potentially) useful, or not. Thus, if this very first transcript already shows that the sub-tree will not be useful, there is no need to run the full-fledged sub-tree extractor, saving precious time.

To illustrate this more, we again consider the simple case of a dishonest prover that succeeds with certainty. Then, after the first run of the sub-extractor to produce the first sub-tree (which requires expected time linear in Q) and having reprogrammed the random oracle with the goal to find another sub-tree that extends the current a^{i^*}, it is cheaper to first do a single run of the prover to learn j^* and only run the full fledged sub-extractor if $j^* = i^*$, and otherwise reprogram and re-try again. With this strategy, we expect Q tries, followed by the run of the sub-extractor, to find a second fitting sub-tree. Altogether, this amounts to linear-in-Q runs of the prover, compared to the Q^2 using the naive approach.

Again, what complicates the rigorous analysis is that the prover may succeed with bounded probability ϵ only, and the event $j^* = i^*$ may depend on the prover/sub-extractor being successful (again) after the reprogramming. Furthermore, as an additional complication, conditioned on the sub-extractor having been successful in the first run and conditioned on the corresponding i^*, *both* the success probability of the prover *and* the run-time of the sub-extractor after the reprogramming may be skewed now. Again, we deal with this by considering an abstract sampling game that mimics the behavior of the extractor, but where the cost function is now more fine-grained in order to distinguish between a single run of the prover and a run of the sub-extractor. Because of this more fine-grained way of defining the "cost", the analysis of the game also becomes substantially more intricate.

Negative Result. We also show that the general exponential security loss of the Fiat-Shamir transformation, when applied to a multi-round protocol, is *not* an artefact of contrived examples, but there exist *natural* protocols that indeed have such an exponential loss. For instance, our negative result applies to the lattice-based protocols in [14,5]. Concretely, we show that the t-fold parallel repetition Π^t of a typical $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-move interactive proof Π features this behavior when $t \geq \mu$. For simplicity, let us assume that t and Q are multiples of μ. Then, in more detail, we show that for any typical $(k_1, \ldots, k_\mu)$-special-sound protocol Π there exists a poly-time Q-query prover $\mathcal{P}^*$ against $\mathsf{FS}[\Pi^t]$ that succeeds in making the verifier accept with probability $\approx \frac{1}{2} Q^\mu \kappa^t / \mu^{\mu+t}$ for *any* statement x, where κ is the knowledge error (as well as the soundness error) of Π. Thus, with the claimed probability, $\mathcal{P}^*$ succeeds in making the verifier accept for statements x that are not in the language and/or for which $\mathcal{P}^*$ does not know a witness. Given that κ^t is the soundness error of Π^t (i.e., the soundness error of Π^t as an interactive proof), this shows that

the *soundness error* of Π^t grows proportionally with Q^μ when applying the Fiat-Shamir transformation. Recent work on the knowledge error of the parallel repetition of special-sound multi-round interactive proofs [7] shows that κ^t is also the knowledge error of Π^t, and so the above shows that the same exponential loss holds in the *knowledge error* of the Fiat-Shamir transformation of a parallel repetition.

1.3 Related Work

Independent Concurrent Work. In independent and to a large extent concurrent work,[3] Wikström [31] achieves a similar positive result on the Fiat-Shamir transformation, using a different approach and different techniques: [31] reduces non-interactive extraction to a form of interactive extraction and then applies a generalized version of [30], while our construction adapts the interactive extractor from [5] and offers a direct analysis. One small difference in the results, which is mainly of theoretical interest, is that our result holds and is meaningful for *any* $Q < |\mathcal{C}|$, whereas [31] requires the challenge set $\mathcal{C}$ to be large.

The Forking Lemma. Security of the Fiat–Shamir transformation of k-special-sound 3-move protocols is widely used for construction of signatures. There, unforgeability is typically proven via a forking lemma [28,9], which extracts, with probability roughly ϵ^k/Q, a witness from a signature-forging adversary with success probability ϵ, where Q is the number of queries to the random oracle. The loss ϵ^k is due to *strict* polynomial time extraction (and can be decreased, but in general not down to ϵ). Such a k-th power loss in the success probability for a constant k is fine in certain settings, e.g., for proving the security of signature schemes; however, not for proofs of knowledge (which, on the other hand, consider *expected* polynomial time extraction [8]).

A previous version of [20] generalizes the original forking lemma [28,9] to accommodate Fiat-Shamir transformations of a larger class of (multi-round) interactive proofs. However, their forking lemma only targets a subclass of the $(k_1, \ldots, k_\mu)$-special-sound interactive proofs considered in this work. Moreover, in terms of (expected) runtime and success probability, our techniques significantly outperform their generalized forking lemma. For this reason, the latest version of [20] is based on our extraction techniques instead.

A forking lemma for *interactive* multi-round proofs was presented in [13] and its analysis was improved in a line of follow-up works [30,24,27,25,2]. This forking lemma shows that multi-round special-sound interactive proofs satisfy a notion of knowledge soundness called *witness extended emulation*. Eventually, it was shown that $(k_1, \ldots, k_\mu)$-special-soundness tightly implies knowledge soundness [5].

The aforementioned techniques for interactive proofs are not directly applicable to the Fiat-Shamir mode. First, incorporating the query complexity Q of a dishonest prover $\mathcal{P}^*$ attacking the non-interactive Fiat–Shamir transformation

[3] When finalizing our write-up, we were informed by Wikström that he derived similar results a few months earlier, subsequently made available online [31].

complicates the analysis. Second, a naive adaptation of the forking lemmas for interactive proofs gives a blow-up of Q^μ in the run-time.

1.4 Structure of the Paper

Section 2 recalls essential preliminaries. In Section 3, the abstract sampling game is defined and analyzed. It is used in Section 4 to handle the Fiat–Shamir transformation of Σ-protocols. Building on the intuition, Section 5 introduces the *refined* game, and Section 6 uses it to handle multi-round protocols. Lastly, our negative result on parallel repetitions is presented in Section 7.

2 Preliminaries

2.1 (Non-)Interactive Proofs

We assume the reader to be familiar with the basic concepts related to interactive proofs, and to non-interactive proofs in the random oracle model. We briefly recall here the notions that are important for us and fix the notation that we will be using. For formal definitions and more details, we refer to the full version [1].

Special-Sound Protocols. We consider a public-coin interactive proof Π for an NP relation R. If Π consists of 3 moves, it is called a Σ-protocol, and we then typically write a for the first message, c for the challenge, and z for the response. A Σ-protocol Π is called *k-special-sound* if there exists a polynomial-time algorithm that computes a witness w for the statement x from any k accepting transcripts $(a, c_1, z_1), \ldots, (a, c_k, z_k)$ for x with the same fist message a and pairwise distinct challenges $c_i \neq c_j$. We refer to Π as being *k-out-of-N special-sound* to emphasize that the challenge space $\mathcal{C}$ has cardinality N.

More generally, we consider $(2\mu + 1)$-move public-coin interactive proofs.[4] The communication transcript is then written as $(a_1, c_1, \ldots, a_\mu, c_\mu, a_{\mu+1})$ by default. Such a protocol is called *$(k_1, \ldots, k_\mu)$-special-sound*, or *$(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$ special-sound* when we want to be explicit about the sizes of the challenge sets, if there exists a polynomial-time algorithm that computes a witness w for the statement x from any accepting *$(k_1, \ldots, k_\mu)$-tree of transcripts* for x, defined in Definition 1 and illustrated in Fig. 1.

Definition 1 (Tree of Transcripts). *Let $k_1, \ldots, k_\mu \in \mathbb{N}$. A $(k_1, \ldots, k_\mu)$-tree of transcripts for a $(2\mu+1)$-move public-coin interactive proof $\Pi = (\mathcal{P}, \mathcal{V})$ is a set of $K = \prod_{i=1}^{\mu} k_i$ transcripts arranged in the following tree structure. The nodes in this tree correspond to the prover's messages and the edges to the verifier's challenges. Every node at depth i has precisely k_i children corresponding to k_i pairwise distinct challenges. Every transcript corresponds to exactly one path from the root node to a leaf node. See Figure 1 for a graphical illustration. We refer to the corresponding tree of challenges as a $(k_1, \ldots, k_\mu)$-tree of challenges.*

[4] We always assume that the prover sends the first and the last message.

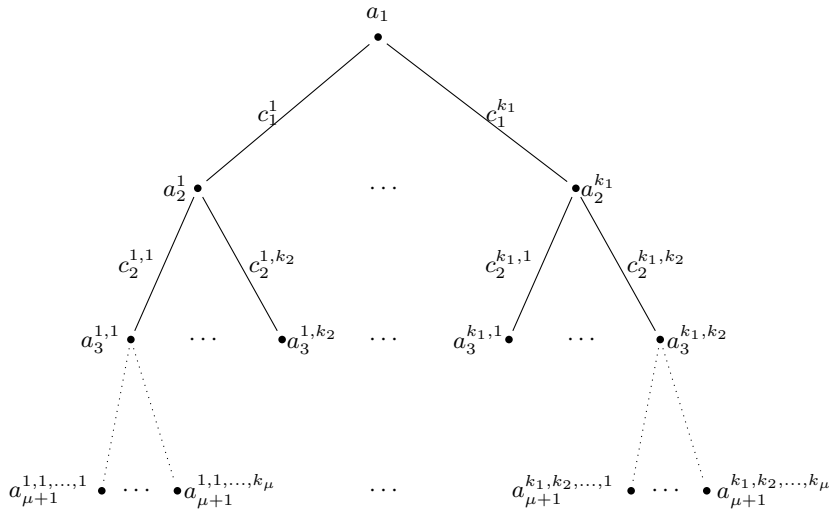

Fig. 1. $(k_1, \ldots, k_\mu)$-tree of transcripts of a $(2\mu+1)$-move interactive proof [5].

A $(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$ special-sound protocol is known to be (knowledge) sound with knowledge/soundness error

$$\mathrm{Er}(k_1, \ldots, k_\mu; N_1, \ldots, N_\mu) = 1 - \prod_{i=1}^{\mu} \frac{N_i - k_i + 1}{N_i} = 1 - \prod_{i=1}^{\mu} \left(1 - \frac{k_i - 1}{N_i}\right), \quad (1)$$

which is tight in general [5]. Note that $\mathrm{Er}(k; N) = (k-1)/N$ and, for all $1 \leq m \leq \mu$,

$$\begin{aligned} &\mathrm{Er}(k_m, \ldots, k_\mu; N_m, \ldots, N_\mu) \\ &\qquad = 1 - \frac{N_m - k_m + 1}{N_m} \big(1 - \mathrm{Er}(k_{m+1}, \ldots, k_\mu; N_{m+1}, \ldots, N_\mu)\big), \end{aligned} \quad (2)$$

where we define $\mathrm{Er}(\emptyset; \emptyset) = 1$. If $N_1 = \cdots = N_\mu = N$, we simply write $\mathrm{Er}(k_1, \ldots, k_\mu; N)$, or $\mathrm{Er}(\mathbf{k}; N)$ for $\mathbf{k} = (k_1, \ldots, k_\mu)$.

The Fiat-Shamir Transformation and NIROPs. By applying the Fiat-Shamir transformation [19] to a public-coin interactive proof, one obtains a *non-interactive* proof in the *random oracle model*, i.e., a so-called *non-interactive random oracle proof (NIROP)*. In the case of a Σ-protocol, the Fiat-Shamir transformation replaces the random choice of the challenge c by setting $c = \mathsf{RO}(a)$ (or $c = \mathsf{RO}(x, a)$ in case of adaptive security), where RO is a random oracle. In case of multi-round protocols, the idea is the same, but one has to be careful with "chaining" the challenges properly. For concreteness, we specify

that the i-th challenge is set to be

$$c_i = \mathsf{RO}_i(a_1, \ldots, a_{i-1}, a_i)\,.$$

Note that, for simplicity, we assume μ different random oracles RO_i then. Furthermore, we assume the range of RO_i to be the corresponding challenge set $\mathcal{C}_i$, and the domain to be $\{0,1\}^{\leq u}$ for large enough u.

The notion of knowledge soundness that we consider for NIROPs, and in particular for the Fiat-Shamir transformation of special-sound protocols, is the natural modification of the knowledge soundness definition of interactive proofs as introduced by Goldreich [23], to the setting of non-interactive proofs in the random oracle model. In more detail, a NIROP is *knowledge sound* with *knowledge error* $\kappa : \mathbb{N} \times \mathbb{N} \to [0,1]$, if there exists an expected polynomial time *knowledge extractor* $\mathcal{E}$ and a polynomial q such that for every Q-query dishonest prover $\mathcal{P}^*$ that succeeds to convince the verifier about a statement x with probability $\epsilon(\mathcal{P}^*, x)$, when $\mathcal{E}$ is given black-box access to $\mathcal{P}^*$ it holds that

$$\Pr\big((x;w) \in R : w \leftarrow \mathcal{E}^{\mathcal{P}^*}(x)\big) \geq \frac{\epsilon(\mathcal{P}^*, x) - \kappa(|x|, Q)}{q(|x|)}\,,$$

i.e., $\mathcal{E}$ succeeds to extract a witness w for x with the above probability. It is not too hard to see that it is sufficient to consider *deterministic* provers $\mathcal{P}^*$.

2.2 Negative Hypergeometric Distribution

An important tool in our analysis is the negative hypergeometric distribution. Consider a bucket containing ℓ green balls and $N - \ell$ red balls, i.e., a total of N balls. In the negative hypergeometric experiment, balls are drawn uniformly at random from this bucket, without replacement, until k green balls have been found or until the bucket is empty. The number of red balls X drawn in this experiment is said to have a *negative hypergeometric distribution* with parameters N, ℓ, k, which is denoted by $X \sim \mathsf{NHG}(N, \ell, k)$.

Lemma 1 (Negative Hypergeometric Distribution). *Let $N, \ell, k \in \mathbb{N}$ with $\ell, k \leq N$, and let $X \sim \mathsf{NHG}(N, \ell, k)$. Then $\mathbb{E}[X] \leq k\frac{N-\ell}{\ell+1}$.*

Remark 1. Typically, negative hypergeometric experiments are restricted to the non-trivial case $\ell \geq k$. For reasons to become clear later, we also allow parameter choices with $\ell < k$ resulting in a trivial negative hypergeometric experiment in which all balls are always drawn.

Remark 2. The above has a straightforward generalization to buckets with balls of more than 2 colors: say ℓ green balls and m_i balls of color i for $1 \leq i \leq M$. The experiment proceeds as before, i.e., drawing until either k green balls have been found or the bucket is empty. Let X_i be the number of balls of color i that are drawn in this experiment. Then $X_i \sim \mathsf{NHG}(\ell + m_i, \ell, k)$ for all i. To see this, simply run the generalized negative hypergeometric experiment without counting the balls that are neither green nor of color i.

3 An Abstract Sampling Game

Towards the goal of constructing and analyzing a knowledge extractor for the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of special-sound interactive proofs Π, we define and analyze an abstract sampling game. Given access to a deterministic Q-query prover $\mathcal{P}^*$, attacking the non-interactive random oracle proof $\mathsf{FS}[\Pi]$, our extractor will essentially play this abstract game in the case Π is a Σ-protocol, and it will play this game recursively in the general case of a multi-round protocol. The abstraction allows us to focus on the crucial properties of the extraction algorithm, without unnecessarily complicating the notation.

The game considers an arbitrary but fixed U-dimensional array M, where, for all $1 \leq j_1, \ldots, j_U \leq N$, the entry $M(j_1, \ldots, j_U) = (v, i)$ contains a bit $v \in \{0, 1\}$ and an index $i \in \{1, \ldots, U\}$. Think of the bit v indicating whether this entry is "good" or "bad", and the index i points to one of the U dimensions. The goal will be to find k "good" entries with the same index i, and with all of them lying in the 1-dimensional array $M(j_1, \ldots, j_{i-1}, \cdot, j_{i+1}, \ldots, j_U)$ for some $1 \leq j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_U \leq N$.

Looking ahead, considering the case of a Σ-protocol first, this game captures the task of our extractor to find k proofs that are valid and feature the same first message but have different hash values assigned to the first message. Thus, in our application, the sequence $j_1, \ldots, j_U$ specifies the function table of the random oracle $\mathsf{RO}\colon \{1, \ldots, U\} \to \{1, \ldots, N\}, i \mapsto j_i$, while the entry $M(j_1, \ldots, j_U) = (v, i)$ captures the relevant properties of the proof produced by the considered prover when interacting with that particular specification of the random oracle. Concretely, the bit v indicates whether the proof is valid, and the index i is the first message a of the proof. Replacing j_i by j_i' then means to reprogram the random oracle at the point $i = a$. Note that after the reprogramming, we want to obtain another valid proof with the *same* first message, i.e., with the same index i (but now a different challenge, due to the reprogramming).

The game is formally defined in Figure 2 and its core properties are summarized in Lemma 2 below. Looking ahead, we note that for efficiency reasons, the extractor will not sample the entire sequence $j_1, \ldots, j_U$ (i.e., function table), but will sample its components on the fly using lazy sampling.

It will be useful to define, for all $1 \leq i \leq U$, the function

$$\begin{aligned} a_i\colon \{1, \ldots, N\}^U &\to \mathbb{N}_{\geq 0}, \\ (j_1, \ldots, j_U) &\mapsto \left|\left\{ j : M(j_1, \ldots, j_{i-1}, j, j_{i+1}, \ldots, j_U) = (1, i) \right\}\right| . \end{aligned} \tag{3}$$

The value $a_i(j_1, \ldots, j_U)$ counts the number of entries that are "good" and have index i in the 1-dimensional array $M(j_1, \ldots, j_{i-1}, \cdot, j_{i+1}, \ldots, j_U)$. Note that a_i does not depend on the i-th entry of the input vector $(j_1, \ldots, j_U)$, and so, by a slight abuse of notation, we sometimes also write $a_i(j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_U)$.

Lemma 2 (Abstract Sampling Game). *Consider the game in Figure 2. Let $J = (J_1, \ldots, J_U)$ be uniformly distributed in $\{1, \ldots, N\}^U$, indicating the first entry sampled, and let $(V, I) = M(J_1, \ldots, J_U)$. Further, for all $1 \leq i \leq U$,*

Fig. 2. Abstract Sampling Game.

Parameters: $k, N, U \in \mathbb{N}$, and M a U-dimensional array with entries in $M(j_1, \ldots, j_U) \in \{0,1\} \times \{1, \ldots, U\}$ for all $1 \leq j_1, \ldots, j_U \leq N$.

- Sample $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$ at random and set $(v, i) = M(j_1, \ldots, j_U)$.
- If $v = 0$, abort.
- Else, repeat
  - sample $j' \in \{1, \ldots, N\} \setminus \{j_i\}$ (without replacement),
  - compute $(v', i') = M(j_1, \ldots, j_{i-1}, j', j_{i+1}, \ldots, j_U)$,

  until either $k-1$ additional entries equal to $(1, i)$ have been found or until all indices j' have been tried.

let $A_i = a_i(J)$. Moreover, let X be the number of entries of the form $(1, i)$ with $i = I$ sampled (including the first one), and let Λ be the total number of entries sampled in this game. Then

$$\mathbb{E}[\Lambda] \leq 1 + (k-1)P \quad \textit{and}$$
$$\Pr(X = k) \geq \frac{N}{N-k+1}\left(\Pr(V = 1) - P \cdot \frac{k-1}{N}\right),$$

where $P = \sum_{i=1}^{U} \Pr(A_i > 0)$.

Remark 3. Note the abstractly defined parameter P. In our application, where the index i of $(v, i) = M(j_1, \ldots, j_U)$ is determined by the output of a prover making no more than Q queries to the random oracle with function table $j_1, \ldots, j_U$, the parameter P will be bounded by $Q + 1$. We show this formally (yet again somewhat abstractly) in Lemma 3. Intuitively, the reason is that the events $A_i > 0$ are *disjoint* for all but Q indices i (those that the considered prover does *not* query), and so their probabilities add up to at most 1. Indeed, if $a_i(j_1, \ldots, j_U) > 0$ for an index i that the algorithm did *not* query then $M(j_1, \ldots, j_U) \in \{(0, i), (1, i)\}$; namely, since i has not been queried, the index i output by the algorithm is oblivious to the value of j_i. Therefore, given $j_1, \ldots, j_U$, there is at most one *unqueried* index i with $a_i(j_1, \ldots, j_U) > 0$.

Proof (of Lemma 2). **Expected Number of Samples.** Let us first derive an upper bound on the expected value of Λ. To this end, let X' denote the number of sampled entries of the form $(1, i)$ with $i = I$, but, in contrast to X, *without* counting the first one. Similarly, let Y' denote the number of sampled entries of the form (v, i) with $v = 0$ or $i \neq I$, again without counting the first one. Then $\Lambda = 1 + X' + Y'$ and

$$\Pr(X' = 0 \mid V = 0) = \Pr(Y' = 0 \mid V = 0) = 1\,.$$

Hence, $\mathbb{E}[X' \mid V = 0] = \mathbb{E}[Y' \mid V = 0] = 0$.

Let us now consider the expected value $\mathbb{E}[Y' \mid V = 1]$. To this end, we observe that, conditioned on the event $V = 1 \wedge I = i \wedge A_i = a$ with $a > 0$, Y' follows a

negative hypergeometric distribution with parameters $N-1$, $a-1$ and $k-1$. Hence, by Lemma 1,

$$\mathbb{E}[Y' \mid V = 1 \wedge I = i \wedge A_i = a] \leq (k-1)\frac{N-a}{a}\,,$$

and thus, using that $\Pr(X' \leq k-1 \mid V = 1) = 1$,

$$\mathbb{E}[X' + Y' \mid V = 1 \wedge I = i \wedge A_i = a] \leq (k-1) + (k-1)\frac{N-a}{a} = (k-1)\frac{N}{a}\,.$$

On the other hand

$$\Pr(V = 1 \wedge I = i \mid A_i = a) = \frac{a}{N}$$

and thus

$$\Pr(V = 1 \wedge I = i \wedge A_i = a) = \Pr(A_i = a)\frac{a}{N}\,. \tag{4}$$

Therefore, and since $\Pr(V = 1 \wedge I = i \wedge A_i = 0) = 0$,

$$\begin{aligned}
\Pr(V = 1) \cdot \mathbb{E}[X' + Y' \mid V = 1] &= \sum_{i=1}^{U}\sum_{a=1}^{N} \Pr(V = 1 \wedge I = i \wedge A_i = a) \\
&\qquad \cdot \mathbb{E}[X' + Y' \mid V = 1 \wedge I = i \wedge A_i = a] \\
&\leq \sum_{i=1}^{U}\sum_{a=1}^{N} \Pr(A_i = a)(k-1) \\
&= (k-1)\sum_{i=1}^{U} \Pr(A_i > 0) \\
&= (k-1)P\,,
\end{aligned}$$

where $P = \sum_{i=1}^{U} \Pr(A_i > 0)$. Hence,

$$\begin{aligned}
\mathbb{E}[\Lambda] &= \mathbb{E}[1 + X' + Y'] \\
&= 1 + \Pr(V = 0) \cdot \mathbb{E}[X' + Y' \mid V = 0] + \Pr(V = 1) \cdot \mathbb{E}[X' + Y' \mid V = 1] \\
&\leq 1 + (k-1)P\,,
\end{aligned}$$

which proves the claimed upper bound on $\mathbb{E}[\Lambda]$.

Success Probability. Let us now find a lower bound for the "success probability" $\Pr(X = k)$ of this game. Using (4) again, we can write

$$\Pr(X = k) = \sum_{i=1}^{U} \Pr(V = 1 \wedge I = i \wedge A_i \geq k) = \sum_{i=1}^{U}\sum_{a=k}^{N} \Pr(A_i = a)\frac{a}{N}\,.$$

Now, using $a \leq N$, note that

$$\frac{a}{N} = 1 - \left(1 - \frac{a}{N}\right) \geq 1 - \frac{N}{N-k+1}\left(1 - \frac{a}{N}\right)$$

$$= \frac{N}{N-k+1}\left(\frac{N-k+1}{N} - 1 + \frac{a}{N}\right) = \frac{N}{N-k+1}\left(\frac{a}{N} - \frac{k-1}{N}\right).$$

Therefore, combining the two, and using that the summand becomes negative for $a < k$ to argue the second inequality, and using (4) once more, we obtain

$$\begin{aligned}
\Pr(X = k) &\geq \sum_{i=1}^{U}\sum_{a=k}^{N} \Pr(A_i = a)\frac{N}{N-k+1}\left(\frac{a}{N} - \frac{k-1}{N}\right) \\
&\geq \sum_{i=1}^{U}\sum_{a=1}^{N} \Pr(A_i = a)\frac{N}{N-k+1}\left(\frac{a}{N} - \frac{k-1}{N}\right) \\
&= \frac{N}{N-k+1}\sum_{i=1}^{U}\sum_{a=1}^{N}\left(\Pr(V = 1 \wedge I = i \wedge A_i = a) - \Pr(A_i = a)\cdot\frac{k-1}{N}\right) \\
&= \frac{N}{N-k+1}\left(\Pr(V = 1) - \frac{k-1}{N}\sum_{i=1}^{U}\Pr(A_i > 0)\right) \\
&= \frac{N}{N-k+1}\left(\Pr(V = 1) - P\cdot\frac{k-1}{N}\right),
\end{aligned}$$

where, as before, we have used that $\Pr(V = 1 \wedge I = i \wedge A_i = 0) = 0$ for all $1 \leq i \leq U$ to conclude the second equality, and finally that $P = \sum_{i=1}^{U}\Pr(A_i > 0)$. This completes the proof of the lemma. □

Our knowledge extractor will instantiate the abstract sampling game via a deterministic Q-query prover $\mathcal{P}^*$ attacking the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$. The index i of $M(v, i) = (j_1, \ldots, j_U)$ is then determined by the output of $\mathcal{P}^*$, with the random oracle being given by the function table $j_1, \ldots, j_U$. Since the index i is thus determined by Q queries to the random oracle, the following shows that the parameter P will in this case be bounded by $Q + 1$.

Lemma 3. *Consider the game in Figure 2. Let v and idx be functions such that $M(j) = \big(v(j), \mathsf{idx}(j)\big)$ for all $j \in \{1, \ldots, N\}^U$. Furthermore, let $J = (J_1, \ldots, J_U)$ be uniformly distributed in $\{1, \ldots, N\}^U$, and set $A_i = a_i(J)$ for all $1 \leq i \leq U$. Let us additionally assume that for all $j \in \{1, \ldots, N\}^U$ there exists a subset $S(j) \subseteq \{1, \ldots, U\}$ of cardinality at most Q such that $\mathsf{idx}(j) = \mathsf{idx}(j')$ for all j' with $j'_\ell = j_\ell$ for all $\ell \in S(j)$. Then*

$$P = \sum_{i=1}^{U}\Pr(A_i > 0) \leq Q + 1\,.$$

Proof. By basic probability theory, it follows that[5]

$$P = \sum_{i=1}^{U} \Pr(A_i > 0) = \sum_{j \in \{1,\ldots,N\}^U} \Pr(J = j) \sum_{i=1}^{U} \Pr(A_i > 0 \mid J = j)$$
$$= \sum_{j} \Pr(J = j) \Big(\sum_{i \in S(j)} \Pr(A_i > 0 \mid J = j) + \sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j) \Big)$$

Since $|S(j)| \leq Q$ for all j, it follows that

$$P \leq \sum_{j} \Pr(J = j) \Big(Q + \sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j) \Big)$$
$$\leq Q + \sum_{j} \Pr(J = j) \sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j)$$

Now note that, by definition of the sets $S(j)$, for all $j \in \{1, \ldots, N\}^U$, $i \notin S(j)$ and $j^* \in \{1, \ldots, N\}$, it holds that

$$\Pr\big(\mathsf{idx}(J_1, \ldots, J_{i-1}, j^*, J_{i+1}, \ldots, J_U) = \mathsf{idx}(j) \mid J = j\big) = 1\,.$$

Therefore, for all $i \notin S(j) \cup \{\mathsf{idx}(j)\}$,

$$\Pr(A_i > 0 \mid J = j) = 0\,.$$

Hence,

$$\sum_{i \notin S(j)} \Pr(A_i > 0 \mid J = j) \leq \Pr(A_{\mathsf{idx}(j)} > 0 \mid J = j) \leq 1.$$

Altogether, it follows that

$$P \leq Q + \sum_{j} \Pr(J = j) = Q + 1\,,$$

which completes the proof. □

4 Fiat-Shamir Transformation of Σ-Protocols

Let us first consider the Fiat-Shamir transformation of a k-special-sound Σ-protocol Π, i.e., a 3-move interactive proof, with challenge set $\mathcal{C}$; subsequently, in Section 6, we move to general *multi-round* interactive proofs.

Let $\mathcal{P}^*$ be a deterministic dishonest Q-query random-oracle prover, attacking the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of Π on input x. Given a statement x as input, after making Q queries to the random oracle $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$, $\mathcal{P}^*$

[5] The probabilities $\Pr(A_i > 0 \mid J = j)$ are all 0 or 1; however, it's still convenient to use probability notation here.

outputs a proof $\pi = (a, z)$. For reasons to become clear later, we re-format (and partly rename) the output and consider $I := a$ and π as $\mathcal{P}^*$'s output. We refer to the output I as the *index*. Furthermore, we extend $\mathcal{P}^*$ to an algorithm $\mathcal{A}$ that additionally checks the correctness of the proof π. Formally, $\mathcal{A}$ runs $\mathcal{P}^*$ to obtain I and π, queries RO to obtain $c := \mathsf{RO}(I)$, and then outputs

$$I = a\,, \quad y := (a, c, z) \qquad \text{and} \qquad v := V(y)\,,$$

where $V(y) = 1$ if y is an accepting transcript for the interactive proof Π on input x and $V(y) = 0$ otherwise. Hence, $\mathcal{A}$ is a random-oracle algorithm making at most $Q+1$ queries; indeed, it relays the oracle queries done by $\mathcal{P}^*$ and makes the one needed to do the verification. We may write $\mathcal{A}^{\mathsf{RO}}$ to make the dependency of $\mathcal{A}$'s output on the choice of the random oracle RO explicit. $\mathcal{A}$ has a naturally defined success probability

$$\epsilon(\mathcal{A}) := \Pr\bigl(v = 1 : (I, y, v) \leftarrow \mathcal{A}^{\mathsf{RO}}\bigr)\,,$$

where $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$ is chosen uniformly at random. The probability $\epsilon(\mathcal{A})$ equals the success probability $\epsilon(\mathcal{P}^*, x)$ of the random-oracle prover $\mathcal{P}^*$ on input x.

Our goal is now to construct an extraction algorithm that, when given black-box access to $\mathcal{A}$, aims to output k accepting transcripts $y_1, \ldots, y_k$ with common first message a and distinct challenges. By the k-special-soundness property of Π, a witness for statement x can be computed efficiently from these transcripts.

The extractor $\mathcal{E}$ is defined in Figure 3. We note that, after a successful first run of $\mathcal{A}$, having produced a first accepting transcript (a, c, z), we rerun $\mathcal{A}$ from the very beginning and answer all oracle queries consistently, except the query to a; i.e., we *only* reprogram the oracle at the point $I = a$. Note that since $\mathcal{P}^*$ and thus $\mathcal{A}$ is deterministic, and we only reprogram the oracle at the point $I = a$, in each iteration of the repeat loop $\mathcal{A}$ is ensured to make the query to I again.[6]

A crucial observation is the following. Within a run of $\mathcal{E}$, all the queries that are made by the different invocations of $\mathcal{A}$ are answered *consistently* using lazy sampling, except for the queries to the index I, where different responses $c, c', \ldots$ are given. This is indistinguishable from having them answered by a full-fledged random oracle, i.e., by means of a pre-chosen function $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$, but then replacing the output $\mathsf{RO}(I)$ at I by fresh challenges c' for the runs of $\mathcal{A}$ in the repeat loop. By enumerating the elements in the domain and codomain of RO, it is easily seen that the extractor is actually running the abstract game from Figure 2. Thus, bounds on the success probability and the expected run time (in terms of queries to $\mathcal{A}$) follow from Lemma 2 and Lemma 3. Altogether we obtain the following result.

Lemma 4 (Extractor). *The extractor $\mathcal{E}$ of Figure 3 makes an expected number of at most $k+Q\cdot(k-1)$ queries to $\mathcal{A}$ and succeeds in outputting k transcripts*

[6] Of course, it would be sufficient to rewind $\mathcal{A}$ to the point where it makes the (first) query to a, but this would make the description more clumsy.

Fig. 3. Extractor $\mathcal{E}$.

Parameters: $k, Q \in \mathbb{N}$
Black-box access to: $\mathcal{A}$ as above

- Run $\mathcal{A}$ as follows to obtain (I, y_1, v): answer all (distinct) oracle queries with uniformly random values in $\mathcal{C}$. Let c be the response to query I.
- If $v = 0$, abort.
- Else, repeat
  - sample $c' \in \mathcal{C} \setminus \{c\}$ (without replacement);
  - run $\mathcal{A}$ as follows to obtain (I', y', v'): answer the query to I with c', while answering all other queries consistently if the query was performed by $\mathcal{A}$ already on a previous run and with a fresh random value in $\mathcal{C}$ otherwise;

  until either $k-1$ additional challenges c' with $v' = 1$ and $I' = I$ have been found or until all challenges $c' \in \mathcal{C} \setminus \{c\}$ have been tried.
- In the former case, output the k accepting transcripts $y_1, \ldots, y_k$.

$y_1, \ldots, y_k$ with common first message a and distinct challenges with probability at least

$$\frac{N}{N-k+1}\left(\epsilon(\mathcal{A}) - (Q+1)\cdot\frac{k-1}{N}\right).$$

Proof. By enumerating all the elements in the domain and codomain of the random oracle RO, we may assume that $\mathsf{RO}\colon \{1,...,U\} \to \{1,...,N\}$, and thus RO can be represented by the function table $(j_1, ..., j_U) \in \{1, \ldots, N\}^U$ for which $\mathsf{RO}(i) = j_i$. Further, since $\mathcal{P}^*$ is deterministic, the outputs I, y and v of the algorithm $\mathcal{A}$ can be viewed as functions taking as input the function table $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$ of RO, and so we can consider the array $M(j_1, \ldots, j_U) = \big(I(j_1, \ldots, j_U), v(j_1, \ldots, j_U)\big)$.

Then, a run of the extractor perfectly matches up with the abstract sampling game of Figure 2 instantiated with array M. The only difference is that, in this sampling game, we consider full-fledged random oracles encoded by vectors $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$, while the actual extractor implements these random oracles by lazy sampling. Thus, we can apply Lemma 2 to obtain bounds on the success probability and the expected run time. However, in order to control the parameter P, which occurs in the bound of Lemma 2, we make the following observation, so that we can apply Lemma 3 to bound $P \leq Q + 1$.

For every $(j_1, \ldots, j_U)$, let $S(j_1, \ldots, j_U) \subseteq \{1, \ldots, U\}$ be the set of points that $\mathcal{P}^*$ queries to the random oracle when $(j_1, \ldots, j_U)$ corresponds to the entire function table of the random oracle. Then, $\mathcal{P}^*$ will produce the same output when the random oracle is reprogrammed at an index $i \notin S(j_1, \ldots, j_U)$. In particular, $I(j_1, \ldots, j_{i-1}, j, j_{i+1}, \ldots, j_U) = I(j_1, \ldots, j_{i-1}, j', j_{i+1}, \ldots, j_U)$ for all j, j' and for all $i \notin S(j_1, \ldots, j_U)$. Furthermore, $|S(j_1, \ldots, j_U)| \leq Q$. Hence, the conditions of Lemma 3 are satisfied and $P \leq Q + 1$. The bounds on the success probability and the expected run time now follow, completing the proof. □

The existence of the above extractor, combined with the k-special-soundness property, implies the following theorem.

Theorem 1 (Fiat-Shamir Transformation of a Σ-Protocol). *The Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of a k-out-of-N special-sound Σ-protocol Π is knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q+1) \cdot \kappa\,,$$

where $\kappa := \mathrm{Er}(k; N) = (k-1)/N$ is the knowledge error of Π.

5 Refined Analysis of the Abstract Sampling Game

Before we prove knowledge soundness of the Fiat-Shamir transformation of *multi-round* interactive protocols, we reconsider the abstract game of Section 3, and consider a refined analysis of the cost of playing the game. The multi-round knowledge extractor will essentially play a recursive composition of this game; however, the analysis of Section 3 is insufficient for our purposes (resulting in a super-polynomial bound on the run-time of the knowledge extractor). Fortunately, it turns out that a refinement allows us to prove the required (polynomial) upper bound.

In Section 3, the considered cost measure is the number of entries visited during the game. For Σ-protocols, every entry corresponds to a single invocation of the dishonest prover $\mathcal{P}^*$. For multi-round protocols, every entry will correspond to a single invocation of a *sub-tree extractor*. The key observation is that some invocations of the sub-tree extractor are *expensive* while others are *cheap*. For this reason, we introduce a cost function Γ and a constant cost γ to our abstract game, allowing us to differentiate between these two cases. Γ and γ assign a cost to every entry of the array M; Γ corresponds to the cost of an expensive invocation of the sub-tree extractor and γ corresponds to the cost of a cheap invocation. While this refinement presents a natural generalization of the abstract game of Section 3, its analysis becomes significantly more involved.

The following lemma provides an upper bound for the total cost of playing the abstract game in terms of these two cost functions.

Lemma 5 (Abstract Sampling Game - Weighted Version). *Consider again the game of Figure 2, as well a cost function $\Gamma\colon \{1,\dots,N\}^U \to \mathbb{R}_{\geq 0}$ and a constant cost $\gamma \in \mathbb{R}_{\geq 0}$. Let $J = (J_1,\dots,J_U)$ be uniformly distributed in $\{1,\dots,N\}^U$, indicating the first entry sampled, and let $(V,I) = M(J_1,\dots,J_U)$. Further, for all $1 \leq i \leq U$, let $A_i = a_i(J)$, where the function a_i is as defined in Equation 3.*

We define the cost of sampling an entry $M(j_1,\dots,j_U) = (v,i)$ with index $i = I$ to be $\Gamma(j_1,\dots,j_U)$ and the cost of sampling an entry $M(j_1,\dots,j_U) = (v,i)$ with index $i \neq I$ to be γ. Let Δ be the total cost of playing this game. Then

$$\mathbb{E}[\Delta] \leq k \cdot \mathbb{E}[\Gamma(J)] + (k-1) \cdot T \cdot \gamma$$

where $T = \sum_{i=1}^{U} \Pr(I \neq i \wedge A_i > 0) \leq P$.

Remark 4. Note that the parameter T in the statement here differs slightly from its counterpart $P = \sum_i \Pr(A_i > 0)$ in Lemma 2. Recall the informal discussion of P in the context of our application (Remark 3), where the array M is instantiated via a Q-query prover $\mathcal{P}^*$ attacking the Fiat-Shamir transformation of an interactive proof. We immediately see that now the defining events $I \neq i \wedge A_i > 0$ are *empty* for all $U - Q$ indices that the prover does not query, giving the bound $T \leq Q$ here, compared to the bound $P \leq Q + 1$ on P. The formal (and more abstract) statement and proof is given in Lemma 6.

Proof. Let us split up Δ into the cost measures Δ_1, Δ_2 and Δ_3, defined as follows. Δ_1 denotes the total costs of the elements $M(j_1, \ldots, j_U) = (1, i)$ with $i = I$ sampled in the game, i.e., the elements with bit $v = 1$ and index $i = I$; correspondingly, X denotes the number of entries of the form $(1, i)$ with $i = I$ sampled (including the first one if $V = 1$). Second, Δ_2 denotes the total costs of the elements $M(j_1, \ldots, j_U) = (0, i)$ with $i = I$ sampled, i.e., the elements with bit $v = 0$ and index $i = I$; correspondingly, Y denotes the number of entries of the form $(0, i)$ with $i = I$ sampled (including the first one if $V = 0$). Finally, Δ_3 denotes the total costs of the elements $M(j_1, \ldots, j_U) = (v, i)$ with $i \neq I$ sampled; correspondingly, Z denotes the number of entries of this form sampled.

Clearly $\Delta = \Delta_1 + \Delta_2 + \Delta_3$. Moreover, since the cost γ is constant, it follows that $\mathbb{E}[\Delta_3] = \gamma \cdot \mathbb{E}[Z]$. In a similar manner, we now aim to relate $\mathbb{E}[\Delta_1]$ and $\mathbb{E}[\Delta_2]$ to $\mathbb{E}[Y]$ and $\mathbb{E}[Z]$, respectively. However, since the cost function $\Gamma\colon \{1, \ldots, N\}^U \to \mathbb{R}_{\geq 0}$ is not necessarily constant, this is more involved.

For $1 \leq i \leq U$ let us write $J_i^* = (J_1, \ldots, J_{i-1}, J_{i+1}, \ldots, J_U)$, which is uniformly random with support $\{1, \ldots, N\}^{U-1}$. Moreover, for all $1 \leq i \leq U$ and $j^* = (j_1^*, \ldots, j_{i-1}^*, j_{i+1}^*, \cdots, j_U) \in \{1, \ldots, N\}^{U-1}$, let $\Lambda(i, j^*)$ denote the event

$$\Lambda(i, j^*) = [I = i \wedge J_i^* = j^*] .$$

We note that conditioned on the event $\Lambda(i, j^*)$, all samples are picked from the subarray $M(j_1^*, \ldots, j_{i-1}^*, \cdot, j_{i+1}^*, \cdots, j_U^*)$; the first one uniformly at random subject to the index I being i, and the remaining ones (if $V = 1$) uniformly at random (without replacement).

We first analyze and bound $\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*)]$. We observe that, for all i and j^* with $\Pr\big(\Lambda(i, j^*)\big) > 0$,

$$\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*)] = \sum_{\ell=0}^{N} \Pr\big(X = \ell \mid \Lambda(i, j^*)\big) \cdot \mathbb{E}[\Delta_1 \mid \Lambda(i, j^*) \wedge X = \ell] .$$

Since, conditioned on $\Lambda(i, j^*) \wedge X = \ell$ for $\ell \in \{0, \ldots, N\}$, any size-$\ell$ subset of elements with $v = 1$ and index i is equally likely to be sampled, it follows that

$$\mathbb{E}[\Delta_1 \mid \Lambda(i, j^*) \wedge X = \ell] = \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i, j^*)] \cdot \ell .$$

Hence,

$$\begin{aligned} \mathbb{E}[\Delta_1 \mid \Lambda(i, j^*)] &= \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i, j^*)] \cdot \sum_{\ell} \Pr\big(X = \ell \mid \Lambda(i, j^*)\big) \cdot \ell \\ &= \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i, j^*)] \cdot \mathbb{E}[X \mid \Lambda(i, j^*)] . \end{aligned}$$

Similarly,

$$\mathbb{E}[\Delta_2 \mid \Lambda(i,j^*)] = \mathbb{E}[\Gamma(J) \mid V = 0 \wedge \Lambda(i,j^*)] \cdot \mathbb{E}[Y \mid \Lambda(i,j^*)]\,.$$

Next, we bound the expected values of X and Y conditioned on $\Lambda(i,j^*)$. The analysis is a more fine-grained version of the proof of Lemma 2. Bounding $\mathbb{E}[X \mid \Lambda(i,j^*)]$ is quite easy: since $V = 0$ implies $X = 0$ and $V = 1$ implies $X \leq k$, it immediately follows that

$$\begin{aligned}\mathbb{E}[X \mid \Lambda(i,j^*)] &= \Pr(V = 0 \mid \Lambda(i,j^*)) \cdot \mathbb{E}[X \mid V = 0 \wedge \Lambda(i,j^*)] \\ &\quad + \Pr(V = 1 \mid \Lambda(i,j^*)) \cdot \mathbb{E}[X \mid V = 1 \wedge \Lambda(i,j^*)] \\ &\leq \Pr(V = 1 \mid \Lambda(i,j^*)) \cdot k\,.\end{aligned}$$

Hence,

$$\mathbb{E}[\Delta_1 \mid \Lambda(i,j^*)] \leq k \cdot \Pr(V = 1 \mid \Lambda(i,j^*)) \cdot \mathbb{E}[\Gamma(J) \mid V = 1 \wedge \Lambda(i,j^*)]\,. \tag{5}$$

Suitably bounding the expectation $\mathbb{E}[Y \mid \Lambda(i,j^*)]$, and thus $\mathbb{E}[\Delta_2 \mid \Lambda(i,j^*)]$, is more involved. For that purpose, we introduce the following parameters. For the considered fixed choice of the index $1 \leq i \leq U$ and of $j^* = (j_1^*, \ldots, j_{i-1}^*, j_{i+1}^*, \cdots, j_U^*)$, we let[7]

$$a := a_i(j^*) = \left|\left\{ j : (v_j, i_j) = M(j_1^*, \ldots, j_{i-1}^*, j, j_{i+1}^*, \ldots, j_U^*) = (1,i) \right\}\right| \quad \text{and}$$
$$b := b_i(j^*) := \left|\left\{ j : (v_j, i_j) = M(j_1^*, \ldots, j_{i-1}^*, j, j_{i+1}^*, \ldots, j_U^*) = (0,i) \right\}\right|\,.$$

Let us first note that

$$\Pr\big(V = 1 \mid \Lambda(i,j^*)\big) = \frac{a}{a+b} \quad \text{and} \quad \Pr\big(V = 0 \mid \Lambda(i,j^*)\big) = \frac{b}{a+b}$$

for all i and j^* with $\Pr\big(\Lambda(i,j^*)\big) > 0$. Therefore, if we condition on the event $V = 1 \wedge \Lambda(i,j^*)$ we implicitly assume that i and j^* are so that a is positive. Now, towards bounding $\mathbb{E}[Y \mid \Lambda(i,j^*)]$, we observe that conditioned on the event $V = 1 \wedge \Lambda(i,j^*)$, the random variable Y follows a negative hypergeometric distribution with parameters $a+b-1$, $a-1$ and $k-1$. Hence, by Lemma 1,

$$\mathbb{E}[Y \mid V = 1 \wedge \Lambda(i,j^*)] \leq (k-1)\frac{b}{a}\,,$$

and thus

$$\begin{aligned}\mathbb{E}[Y \mid \Lambda(i,j^*)] &= \Pr(V = 0 \mid \Lambda(i,j^*)) \cdot \mathbb{E}[Y \mid V = 0 \wedge \Lambda(i,j^*)] \\ &\quad + \Pr(V = 1 \mid \Lambda(i,j^*)) \cdot \mathbb{E}[Y \mid V = 1 \wedge \Lambda(i,j^*)] \\ &\leq \Pr\big(V = 0 \mid \Lambda(i,j^*)\big) + \Pr\big(V = 1 \mid \Lambda(i,j^*)\big) \cdot (k-1)\frac{b}{a} \\ &= \frac{b}{a+b} + \frac{a}{a+b} \cdot (k-1)\frac{b}{a} = k\frac{b}{a+b} \\ &= k \cdot \Pr(V = 0 \mid \Lambda(i,j^*))\,,\end{aligned}$$

[7] Recall that we use $a_i(j_1, \ldots, j_U)$ and $a_i(j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_U)$ interchangeably, exploiting that $a_i(j_1, \ldots, j_U)$ does not depend on the i-th input j_i.

where we use that $\mathbb{E}[Y \mid V = 0 \wedge \Lambda(i, j^*)] = 1$. Hence,

$$\mathbb{E}[\Delta_2 \mid \Lambda(i, j^*)] \leq k \cdot \Pr(V = 0 \mid \Lambda(i, j^*)) \cdot \mathbb{E}[\Gamma(J) \mid V = 0 \wedge \Lambda(i, j^*)]\,,$$

and thus, combined with Equation 5,

$$\mathbb{E}[\Delta_1 + \Delta_2 \mid \Lambda(i, j^*)] \leq k \cdot \mathbb{E}[\Gamma(J) \mid \Lambda(i, j^*)]\,.$$

Since this inequality holds for all i and j^* with $\Pr\big(\Lambda(i, j^*)\big) > 0$, it follows that

$$\mathbb{E}[\Delta_1 + \Delta_2] \leq k \cdot \mathbb{E}[\Gamma(J)]\,.$$

What remains is to show that $\mathbb{E}[Z] \leq (k-1)T$. The slightly weaker bound $\mathbb{E}[Z] \leq (k-1)P$ follows immediately from observing that $Z \leq Y'$ for Y' as in the proof of Lemma 2 (the number of entries counted by Z is a subset of those counted by Y'), and using that $\mathbb{E}[Y'] \leq \mathbb{E}[X' + Y'] \leq (k-1)P$ as derived in the proof of Lemma 2. This then implies $\mathbb{E}[\Delta_3] \leq (k-1) \cdot P \cdot \gamma$, and so, altogether, we obtain the weaker version of the claimed bound:

$$\mathbb{E}[\Delta] = \mathbb{E}[\Delta_1 + \Delta_2 + \Delta_3] \leq k \cdot \mathbb{E}[\Gamma(J)] + (k-1) \cdot P \cdot \gamma\,.$$

For the stronger version in terms of T, we refer to the full version [1]. □

Lemma 6. *Consider the game in Figure 2. Let v and idx be functions such that $M(j) = \big(v(j), \mathsf{idx}(j)\big)$ for all $j \in \{1, \ldots, N\}^U$. Furthermore, let $J = (J_1, \ldots, J_U)$ be uniformly distributed in $\{1, \ldots, N\}^U$ and set $A_i = a_i(J)$ for all $1 \leq i \leq U$ as in Equation 3. Let us additionally assume that for all $j \in \{1, \ldots, N\}^U$ there exists a subset $S(j) \subseteq \{1, \ldots, U\}$ of cardinality at most Q such that $\mathsf{idx}(j) = \mathsf{idx}(j')$ for all j, j' with $j_\ell = j'_\ell$ for all $\ell \in S(j)$. Then*

$$T = \sum_{i=1}^{U} \Pr\big(\mathsf{idx}(J) \neq i \wedge A_i > 0\big) \leq Q\,.$$

See the full version [1] for a proof of lemma 6.

6 Fiat-Shamir Transformation of Multi-Round Protocols

Let us now move to multi-round interactive proofs. More precisely, we consider the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of a $\mathbf{k}$-special-sound $(2\mu+1)$-move interactive proof Π, with $\mathbf{k} = (k_1, \ldots, k_\mu)$. While the multi-round extractor has a natural recursive construction, it requires a more fine-grained analysis to show that it indeed implies knowledge soundness.

To avoid a cumbersome notation, below we first handle $(2\mu+1)$-move interactive proofs in which the verifier samples all μ challenges uniformly at random from the *same* set $\mathcal{C}$. In the full version [1], we consider a generalization for varying challenges sets and extend our results to *adaptive* security.

Consider a deterministic dishonest Q-query random-oracle prover $\mathcal{P}^*$, attacking the Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of a $\mathbf{k}$-special-sound interactive proof Π on input x. We assume all challenges to be elements in the same set $\mathcal{C}$. After making at most Q queries to the random oracle, $\mathcal{P}^*$ outputs a proof $\pi = (a_1, \ldots, a_{\mu+1})$. We re-format the output and consider

$$I_1 := a_1\,,\; I_2 := (a_1, a_2)\,, \ldots,\; I_\mu := (a_1, \ldots, a_\mu) \quad \text{and} \quad \pi$$

as $\mathcal{P}^*$'s output. Sometimes it will be convenient to also consider $I_{\mu+1} := (a_1, \ldots, a_{\mu+1})$. Furthermore, we extend $\mathcal{P}^*$ to a random-oracle algorithm $\mathcal{A}$ that additionally checks the correctness of the proof π. Formally, relaying all the random oracle queries that $\mathcal{P}^*$ is making, $\mathcal{A}$ runs $\mathcal{P}^*$ to obtain $\mathbf{I} = (I_1, \ldots, I_\mu)$ and π, additionally queries the random oracle to obtain $c_1 := \mathsf{RO}(I_1), \ldots, c_\mu := \mathsf{RO}(I_\mu)$, and then outputs

$$\mathbf{I}\,, \quad y := (a_1, c_1, \ldots, a_\mu, c_\mu, a_{\mu+1}) \qquad \text{and} \qquad v := V(x, y)\,,$$

where $V(x, y) = 1$ if y is an accepting transcript for the interactive proof Π on input x and $V(x, y) = 0$ otherwise. Hence, $\mathcal{A}$ makes at most $Q + \mu$ queries (the queries done by $\mathcal{P}^*$, and the queries to $I_1, \ldots, I_\mu$). Moreover, $\mathcal{A}$ has a naturally defined success probability

$$\epsilon(\mathcal{A}) := \Pr\big(v = 1 : (I, y, v) \leftarrow \mathcal{A}^{\mathsf{RO}}\big)\,,$$

where $\mathsf{RO}\colon \{0,1\}^{\leq u} \to \mathcal{C}$ is distributed uniformly. As before, $\epsilon(\mathcal{A}) = \epsilon(\mathcal{P}^*, x)$.

Our goal is now to construct an extraction algorithm that, when given black-box access to $\mathcal{A}$, and thus to $\mathcal{P}^*$, aims to output a $\mathbf{k}$-tree of accepting transcripts. By the $\mathbf{k}$-special-soundness property of Π, a witness for statement x can then be computed efficiently from these transcripts.

To this end, we recursively introduce a sequence of "sub-extractors" $\mathcal{E}_1, \ldots, \mathcal{E}_\mu$, where $\mathcal{E}_m$ aims to find a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts. The main idea behind this recursion is that such a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts is the composition of k_m appropriate $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-trees.

For technical reasons, we define the sub-extractors $\mathcal{E}_m$ as *random-oracle* algorithms, each one making $Q + \mu$ queries to a random oracle. As we will see, the recursive definition of $\mathcal{E}_m$ is very much like the extractor from the 3-move case, but with $\mathcal{A}$ replaced by the sub-extractor $\mathcal{E}_{m+1}$; however, for this to work we need the sub-extractor to be the same kind of object as $\mathcal{A}$, thus a random-oracle algorithm making the same number of queries. As base for the recursion, we consider the algorithm $\mathcal{A}$ (which outputs a single transcript, i.e., a $(1, \ldots, 1)$-tree); thus, the sub-extractor $\mathcal{E}_\mu$ (which outputs a $(1, \ldots, 1, k_\mu)$-tree) is essentially the extractor of the 3-move case, but with $\mathcal{A}$ now outputting an index *vector* $\mathbf{I} = (I_1, \ldots, I_\mu)$, and with $\mathcal{E}_\mu$ being a *random-oracle* algorithm, so that we can recursively replace the random-oracle algorithm $\mathcal{A}$ by $\mathcal{E}_\mu$ to obtain $\mathcal{E}_{\mu-1}$, etc.

Formally, the recursive definition of $\mathcal{E}_m$ from $\mathcal{E}_{m+1}$ is given in Figure 4, where $\mathcal{E}_{\mu+1}$ (the base case) is set to $\mathcal{E}_{\mu+1} := \mathcal{A}$, and where $\mathcal{E}_m$ exploits the following

early abort feature of $\mathcal{E}_{m+1}$: like $\mathcal{A}$, the sub-extractor $\mathcal{E}_{m+1}$ computes the index vector it eventually outputs by running $\mathcal{P}^*$ *as its first step* (see Lemma 7 below). This allows the executions of $\mathcal{E}_{m+1}$ in the repeat loop in Fig. 4 to abort after a single run of $\mathcal{P}^*$ if the requirement $I'_m = I_m$ on its index vector $\mathbf{I}$ is not satisfied, without proceeding to produce the remaining parts y', v' of the output (which would invoke more calls to $\mathcal{P}^*$).

The actual extractor $\mathcal{E}$ is then given by a run of $\mathcal{E}_1$, with the $Q + \mu$ random-oracle queries made by $\mathcal{E}_1$ being answered using lazy-sampling.

Fig. 4. Sub-extractor $\mathcal{E}_m$, as a $(Q + \mu)$-query random-oracle algorithm.

Parameters: $k_m, Q \in \mathbb{N}$
Black-box access to: $\mathcal{E}_{m+1}$
Random oracle queries: $Q + \mu$

- Run $\mathcal{E}_{m+1}$ as follows to obtain $(\mathbf{I}, y_1, v)$: relay the $Q+\mu$ queries to the random oracle and record all query-response pairs. Let c be the response to query I_m.
- If $v = 0$, abort with output $v = 0$.
- Else, repeat
  - sample $c' \in \mathcal{C} \setminus \{c\}$ (without replacement);
  - run $\mathcal{E}_{m+1}$ as follows to obtain $(\mathbf{I}', y', v')$, aborting right after the initial run of $\mathcal{P}^*$ if $I'_m \neq I_m$: answer the query to I_m with c', while answering all other queries consistently if the query was performed by $\mathcal{E}_{m+1}$ already on a previous run and with a fresh random value in $\mathcal{C}$ otherwise;

  until either $k_m - 1$ additional challenges c' with $v' = 1$ and $I'_m = I_m$ have been found or until all challenges $c' \in \mathcal{C} \setminus \{c\}$ have been tried.
- In the former case, output $\mathbf{I}$, the k_m accepting $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-trees $y_1, \ldots, y_{k_m}$, and $v := 1$; in the latter case, output $v := 0$.

Remark 5. Let us emphasize that within *one* run of $\mathcal{E}_m$, except for the query to I_m for which the response is "reprogrammed", all the queries made by the multiple runs of the sub-extractor $\mathcal{E}_{m+1}$ in the repeat loop are answered *consistently*, both with the run of $\mathcal{E}_{m+1}$ in the first step and among the runs in the repeat loop. This means, a query to a value ξ that has been answered by η in a previous run on $\mathcal{E}_{m+1}$ (within the considered run of $\mathcal{E}_m$) is again answered by η, and a query to a value ξ' that has not been queried yet in a previous run on $\mathcal{E}_{m+1}$ (within the considered run of $\mathcal{E}_m$) is answered with a freshly chosen uniformly random $\eta' \in \mathcal{C}$. In *multiple* runs of $\mathcal{E}_m$, very naturally the random tape of $\mathcal{E}_m$ will be refreshed, and thus there is no guaranteed consistency among the answers to the query calls of $\mathcal{E}_{m+1}$ across multiple runs of $\mathcal{E}_m$.

The following lemma captures some technical property of the sub-extractors $\mathcal{E}_m$. Subsequently, Proposition 1 shows that $\mathcal{E}_m$, if successful, indeed outputs a $(1, \ldots, 1, k_m \ldots, k_\mu)$-tree of accepting transcripts. Proposition 2 bounds the success probability and expected run time of $\mathcal{E}_m$. All statements are understood to hold for any statement x and any $m \in \{1, \ldots, \mu + 1\}$.

Lemma 7 (Consistency of $\mathcal{P}^*$ and $\mathcal{E}_m$). *$\mathcal{E}_m$ obtains the index vector $\mathbf{I}$, which it eventually outputs, by running $(\mathbf{I}, \pi) \leftarrow \mathcal{P}^*$ as its first step. In particular, for any fixed choice of the random oracle RO, the index vector $\mathbf{I}$ output by $\mathcal{E}_m^{\mathsf{RO}}$ matches the one output by $\mathcal{P}^{*,\mathsf{RO}}$.*

Proof. The first claim holds for $\mathcal{E}_{\mu+1} = \mathcal{A}$ by definition of $\mathcal{A}$, and it holds for $\mathcal{E}_m$ with $m \leq \mu$ by induction, given that $\mathcal{E}_m$ runs $\mathcal{E}_{m+1}$ as a first step. The claim on the matching index vectors then follows trivially. □

Proposition 1 (Correctness). *For any fixed choice of the random oracle let $(\mathbf{I}, y_1, \ldots, y_{k_m}, v) \leftarrow \mathcal{E}_m^{\mathsf{RO}}(x)$. If $v = 1$ then $(y_1, \ldots, y_{k_m})$ forms a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts.*

Proof. All $k_{m+1} \cdots k_\mu$ transcripts in a $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-tree contain the same partial transcript $(a_1, c_1, \ldots, c_m, a_{m+1})$, i.e., the first $2m-1$ messages in all these transcripts coincide. Hence, any $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-tree of transcripts has a well-defined *trunk* $(a_1, c_1, \ldots, c_m, a_{m+1})$.

By induction on m, we will prove that if $v = 1$ then $(y_1, \ldots, y_{k_m})$ forms a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts with trunk $(a_1, \mathsf{RO}(I_1), \ldots, \mathsf{RO}(I_{m-1}), a_m)$, where $I_{m+1} = (a_1, \ldots, a_{m+1})$. This obviously implies the correctness claim.

For the base case $m = \mu + 1$, recall that $\mathcal{E}_{\mu+1} = \mathcal{A}$, and that by definition of $\mathcal{A}$ and its output $(\mathbf{I}, y, v)$, if $v = 1$ then y is an accepting transcript, and thus a $(1, \ldots, 1)$-tree of accepting transcripts with $(a_1, \mathsf{RO}(I_1), \ldots, \mathsf{RO}(I_\mu), a_{\mu+1})$ as trunk where $I_{\mu+1} = (a_1, \ldots, a_{\mu+1})$, by definition of $\mathbf{I} = (I_1, \ldots, I_\mu)$.

For the induction step, by the induction hypothesis on $\mathcal{E}_{m+1}$ and its output $(\mathbf{I}, y, v)$, if $v = 1$ then y is a $(1, \ldots, 1, k_{m+1}, \ldots, k_\mu)$-tree of accepting transcripts with trunk $(a_1, \mathsf{RO}(I_1), \ldots, a_m, \mathsf{RO}(I_m), a_{m+1})$, where $I_{m+1} = (a_1, \ldots, a_{m+1})$. This holds for $(\mathbf{I}, y_1, v)$ output by $\mathcal{E}_{m+1}$ in the first step of $\mathcal{E}_m$, but also for any invocation of $\mathcal{E}_{m+1}$ in the repeat loop with output $(\mathbf{I}', y', v')$, here with trunk $(a'_1, \mathsf{RO}'(I'_1), \ldots, a'_m, \mathsf{RO}'(I'_m), a'_{m+1})$, where $I'_{m+1} = (a'_1, \ldots, a'_{m+1})$ and RO' is such that $\mathsf{RO}'(I_j) = \mathsf{RO}(I_j)$ for all $j \neq m$, while $\mathsf{RO}(I_m) = c_i$ and $\mathsf{RO}'(I_m) = c'_i$. By definition of the output of $\mathcal{E}_m$, for y_1 and y' occurring in the output of $\mathcal{E}_m$, it is ensured that $I_m = I'_m$.

Now note that, by Lemma 7, for the purpose of the argument, $\mathcal{E}_m$ could have run $\mathcal{P}^*$ instead of $\mathcal{E}_{m+1}$ to obtain $\mathbf{I}$ and $\mathbf{I}'$. Therefore, by definition of the index vectors output by $\mathcal{P}^*$, which is such that I_j is a (fixed-size) prefix of I_m for $j < m$, it follows that also $I_j = I'_j$ for all $j < m$.

Therefore, the output $y_1, \ldots, y_{k_m}$ of $\mathcal{E}_m$ forms a $(1, \ldots, 1, k_m, \ldots, k_\mu)$-tree of accepting transcripts with trunk $(a_1, \mathsf{RO}(I_1), \ldots, a_{m-1}, \mathsf{RO}(I_{m-1}), a_m)$, where $I_m = (a_1, \ldots, a_m)$. This completes the proof. □

Proposition 2 (Run Time and Success Probability). *Let $K_m = k_m \cdots k_\mu$. The extractor $\mathcal{E}_m$ makes an expected number of at most $K_m + Q \cdot (K_m - 1)$ queries to $\mathcal{A}$ (and thus to $\mathcal{P}^*$) and successfully outputs $v = 1$ with probability at least*

$$\frac{\epsilon(\mathcal{A}) - (Q+1) \cdot \kappa_m}{1 - \kappa_m}$$

where $\kappa_m := \mathrm{Er}(k_m, \ldots, k_\mu; N)$ *is as defined in Equation 1.*

Proof. The proof goes by induction on m. The base case $m = \mu+1$ holds trivially, understanding that $K_{\mu+1} = 1$ and $\mathrm{Er}(\emptyset, N) = 0$. Indeed, $\mathcal{E}_{\mu+1}$ makes 1 call to $\mathcal{A}$ and outputs $v = 1$ with probability $\epsilon(\mathcal{A})$. Alternatively, we can take $m = \mu$ as base case, which follows immediately from Lemma 4.

For the induction step, we assume now that the lemma is true for $m' = m+1$ and consider the extractor $\mathcal{E}_m$. As in the 3-move case, we observe that, within a run of $\mathcal{E}_m$, all the queries that are made by the different invocations of $\mathcal{E}_{m+1}$ are answered *consistently* using lazy sampling, except for the queries to the index I_m, which is answered with different responses c'. This is indistinguishable from having them answered by a full-fledged random oracle $\mathsf{RO}\colon \{1, \ldots, U\} \to \{1, \ldots, N\}$, where we have enumerated the domain and codomain of RO as before. This enumeration allows RO to be identified with its function table $(j_1, \ldots, j_U) \in \{1, \ldots, N\}^U$. Thus, the extractor is actually running the abstract sampling game from Figure 2.

However, in contrast to the instantiation of Section 4, the entries of the array M are now *probabilistic*. Namely, while $\mathcal{A}$ is deterministic, the extractor $\mathcal{E}_{m+1}$ is a probabilistic algorithm. Fortunately, this does not influence the key properties of the abstract sampling game. For the purpose of the analysis we may namely fix the randomness of the extractor $\mathcal{E}_{m+1}$. By linearity of the success probability and the expected run time, the bounds that hold for any fixed choice of randomness also hold when averaged over the randomness. Thus, we can apply Lemma 2 and Lemma 5 to bound the success probability and the expected run time.[8]

To control the parameters P and T, which occur in the bounds of these lemmas, we make the following observation. A similar observation was required in the proof of Lemma 4.

First, by Lemma 7, the index vector $\mathbf{I}$ output by $\mathcal{E}_{m+1}$ matches the index vector output by $\mathcal{P}^*$, when given the same random oracle RO. Second, since $\mathcal{P}^*$ is deterministic, its output can only change when the random oracle is reprogrammed at one of the indices $i \in \{1, \ldots, U\}$ queried by $\mathcal{P}^*$. Therefore, for every $(j_1, \ldots, j_U)$, let $S(j_1, \ldots, j_U) \subseteq \{1, \ldots, U\}$ be the set of points that $\mathcal{P}^*$ queries to the random oracle when $(j_1, \ldots, j_U)$ corresponds to the entire function table of the random oracle. Then, $\mathcal{P}^*$ will produce the same output when the random oracle is reprogrammed at an index $i \notin S(j_1, \ldots, j_U)$. In particular, $\mathbf{I}(j_1, \ldots, j_{i-1}, j, j_{i+1}, \ldots, j_U) = \mathbf{I}(j_1, \ldots, j_{i-1}, j', j_{i+1}, \ldots, j_U)$ for all j, j' and for all $i \notin S(j_1, \ldots, j_U)$. Furthermore, $|S(j_1, \ldots, j_U)| \leq Q$. Hence, the conditions of Lemma 3 and Lemma 6 are satisfied, and it follows that $P \leq Q + 1$ and $T \leq Q$. We are now ready to analyze the success probability and the expected number of $\mathcal{A}$ queries of $\mathcal{E}_m$.

[8] To be more precise, to allow for fresh randomness in the different runs of $\mathcal{E}_{m+1}$ within $\mathcal{E}_m$, we first replace the randomness of $\mathcal{E}_{m+1}$ by $F(j_1, \ldots, j_U)$ for a random function F, where $(j_1, \ldots, j_U)$ is the function table of the random oracle providing the answers to $\mathcal{E}_{m+1}$'s queries, and then we fix the choice of F and average over F after having applied Lemma 2 and Lemma 5.

Success Probability. By the induction hypothesis, the success probability p_{m+1} of $\mathcal{E}_{m+1}$ is bounded by

$$p_{m+1} \geq \frac{\epsilon(\mathcal{A}) - (Q+1) \cdot \kappa_{m+1}}{1 - \kappa_{m+1}} \,.$$

Then, by Lemma 2 and Lemma 3, the success probability of $\mathcal{E}_m$ is bounded by

$$\begin{aligned} &\frac{N}{N-k_m+1}\left(p_{m+1} - (Q+1)\frac{k_m-1}{N}\right) \\ &\qquad \geq \frac{N}{N-k_m+1}\left(\frac{\epsilon(\mathcal{A}) - (Q+1)\cdot\kappa_{m+1}}{1-\kappa_{m+1}} - (Q+1)\frac{k_m-1}{N}\right) . \end{aligned}$$

By the recursive property (2) of $\kappa_m = \mathrm{Er}\big(k_m, \ldots, k_\mu; N, \ldots, N\big)$, it follows that

$$\frac{N-k_m+1}{N}(1-\kappa_{m+1}) = 1 - \kappa_m \,.$$

Hence,

$$\begin{aligned} p_m &\geq \frac{\epsilon(\mathcal{A}) - (Q+1)\cdot\kappa_{m+1}}{1-\kappa_m} - (Q+1)\frac{k_m-1}{N-k_m+1} \\ &= \frac{1}{1-\kappa_m}\left(\epsilon(\mathcal{A}) - (Q+1)\cdot\left(\kappa_{m+1} + (1-\kappa_m)\frac{k_m-1}{N-k_m+1}\right)\right) \\ &= \frac{1}{1-\kappa_m}\left(\epsilon(\mathcal{A}) - (Q+1)\cdot\left(1 - (1-\kappa_m)\cdot\right.\right. \\ &\qquad\qquad \left.\left.\frac{N}{N-k_m+1} + (1-\kappa_m)\frac{k_m-1}{N-k_m+1}\right)\right) \\ &= \frac{\epsilon(\mathcal{A}) - (Q+1)\cdot\kappa_m}{1-\kappa_m} \,, \end{aligned}$$

which proves the claimed success probability.

Expected Number of $\mathcal{A}$-Queries. Let the random variable T_m denote the number of $\mathcal{A}$-queries made by extractor $\mathcal{E}_m$. By the induction hypothesis,

$$\mathbb{E}[T_{m+1}] \leq K_{m+1} + Q \cdot (K_{m+1} - 1) \,.$$

We make one crucial observation, allowing us to achieve the claimed query complexity, linear in Q. Namely, we can view the run of a (sub)extractor as a *two-stage* algorithm that allows an *early abort*. By Lemma 7, after only one $\mathcal{A}$-query $\mathcal{E}_{m+1}$ already returns the index I_m. At this stage, $\mathcal{E}_m$ can decide whether to continue the execution of $\mathcal{E}_{m+1}$ or to *early abort* this execution. If the index is incorrect, i.e., it does not match the one obtained in the first invocation of

$\mathcal{E}_{m+1}$, then $\mathcal{E}_m$ early aborts the execution of $\mathcal{E}_{m+1}$. Only if the index is correct, the $\mathcal{E}_{m+1}$ execution has to be finished.

For this reason, we define the function $(j_1, \ldots, j_U) \mapsto \Gamma(j_1, \ldots, j_U)$, where $\Gamma(j_1, \ldots, j_U)$ is the (expected) costs of running $\mathcal{E}_{m+1}$ (completely) with random oracle $(j_1, \ldots, j_U)$. Moreover, we set $\gamma = 1$ indicating the cost of an early abort invocation of $\mathcal{E}_{m+1}$. These cost functions measure the expected number of calls to $\mathcal{A}$.

Hence, by Lemma 5 and Lemma 6, the expected cost of running $\mathcal{E}_m$ is

$$\begin{aligned}\mathbb{E}[T_m] &\leq k_m \cdot \mathbb{E}[\Gamma(C)] + \gamma \cdot Q \cdot (k_m - 1) = k_m \cdot \mathbb{E}[T_{m+1}] + Q \cdot (k_m - 1) \\ &\leq K_m + Q \cdot (K_m - k_m) + Q \cdot (k_m - 1) = K_m + Q \cdot (K_m - 1)\,,\end{aligned}$$

where C is distributed uniformly at random in $\mathcal{C}^U$. This completes the proof. □

The existence of extractor $\mathcal{E}_1$, combined with the **k**-special-soundness property, implies the following. This theorem shows that the Fiat-Shamir security loss for **k**-out-of-**N** special-sound $(2\mu + 1)$-round interactive proofs is $Q + 1$, i.e., the security loss is linear in the query complexity Q of provers $\mathcal{P}^*$ attacking the considered non-interactive random oracle proof $\mathsf{FS}[\Pi]$. In particular, the Fiat-Shamir security loss is independent of the number of rounds $(2\mu+1)$ of Π.

Theorem 2 (FS Transformation of a $(k_1, \ldots, k_\mu)$-Special-Sound Protocol). *The Fiat-Shamir transformation $\mathsf{FS}[\Pi]$ of a $\mathbf{k} = (k_1, \ldots, k_\mu)$-special-sound interactive proof Π, in which all challenges are sampled from a set $\mathcal{C}$ of size N, is knowledge sound with knowledge error*

$$\kappa_{\mathrm{fs}}(Q) = (Q + 1)\kappa\,,$$

where $\kappa := \mathrm{Er}(\mathbf{k}; N)$ is the knowledge error of the interactive proof Π.

7 The Fiat-Shamir Transformation of Parallel Repetitions

In the previous sections we have established a positive result; for a broad class of interactive proofs the Fiat-Shamir security loss is only linear in the query complexity Q and independent of the number of rounds. One might therefore wonder whether the generic $(Q + 1)^\mu$ security loss, for $(2\mu + 1)$-move protocols, is only tight for contrived examples. In this section, we show that this is *not* the case. We demonstrate a non-trivial attack on the Fiat–Shamir transformation of the *parallel repetition* of **k**-special-sound protocols.

Let $\Pi = (\mathcal{P}, \mathcal{V})$ be a $(2\mu+1)$-move **k**-special-sound interactive proof. We write $\Pi^t = (\mathcal{P}^t, \mathcal{V}^t)$ for its t-fold parallel repetition. That is, the prover $\mathcal{P}^t(x; w)$ runs t instances of $\mathcal{P}(x; w)$, i.e., each message is a tuple $(a^1, \ldots, a^t)$ of messages, one for each parallel thread of execution. Likewise, the verifier $\mathcal{V}^t(x)$ runs t instances of $\mathcal{V}(x)$ in parallel, i.e., each challenge is a tuple $(c^1, \ldots, c^t)$ of challenges, one for each parallel thread of the execution. Finally, the verifier accepts if all parallel instances are accepting.

Assuming certain natural properties on Π, which are satisfied by typical examples, and assuming again for simplicity that the challenge spaces $\mathcal{C}_i$ all have the same cardinality N, we show that, when $t \geq \mu$, there exists a malicious Q-query prover $\mathcal{P}^*$, attacking $\mathsf{FS}[\Pi^t]$, that, for any statement x, succeeds in convincing the verifier with probability at least

$$\frac{1}{2}\frac{Q^\mu}{\mu^{t+\mu}}\,\mathrm{Er}(\mathbf{k};N)^t\,,$$

assuming some mild conditions on the parameters. Given that $\mathrm{Er}(\mathbf{k};N)^t$ equals the soundness as well as the knowledge error of Π^t,[9] our attack shows that the security loss of the Fiat-Shamir transformation, when applied to the t-fold parallel repetition of Π, is at least $\frac{1}{2}Q^\mu/\mu^{t+\mu}$. This stands in stark contrast to a single execution of a $\mathbf{k}$-special-sound protocol, where the loss is linear in Q and independent of μ.

We go on to discuss the kind of $\mathbf{k}$-special-sound protocols Π for which our attack applies. For simplicity, we restrict our attention here to $\mathbf{k} = (k, \ldots, k)$ and assume t and Q to be multiples of μ. In the full version [7], we consider the case of arbitrary $\mathbf{k}$, and the restrictions on t and Q can be easily avoided with some adjustments to the bound and the reasoning. Let $\boldsymbol{\ell} = (\ell, \ldots, \ell)$ where $\ell \leq k-1$. The attack on $\mathsf{FS}[\Pi^t]$ uses a property most $\mathbf{k}$-special-sound protocols Π satisfy, namely that there exists an efficient attack strategy $\mathcal{A}$ against Π which tries to guess challenges up front so that:

1. In any round, $\mathcal{A}$ can prepare and send a message so that if he is lucky and the next challenge falls in a certain set Γ of cardinality ℓ, $\mathcal{A}$ will be able to complete the protocol and have the verifier accept (no matter what challenges $\mathcal{A}$ encounters in the remaining rounds), and
2. until $\mathcal{A}$ is lucky in the above sense, in any round $\mathcal{A}$ can actually prepare B distinct messages as above, for a given parameter B.

We call protocols which admit such an attack strategy *$\boldsymbol{\ell}$-special-unsound with B potential responses per round* (see the full version [7] for a formal definition). The first point in particular implies an attack strategy for the interactive proof Π that succeeds with probability $\mathrm{Er}(\boldsymbol{\ell}+1, N)$. Since many $\mathbf{k}$-special-sound interactive proofs Π are $\boldsymbol{\ell}$-special-unsound with $\boldsymbol{\ell} = \mathbf{k}-1$, this confirms the tightness of the knowledge error $\mathrm{Er}(\mathbf{k}, N)$. The second point implies that in the context of the Fiat-Shamir transformation, an attacker can produce and try multiple message-challenge pairs in any round.

These requirements are very common (for non-trivial $\boldsymbol{\ell}$ and large B). For example, the folding technique of [13], when used to fold two parts into one, satisfies $(3, \ldots, 3)$-special-soundness and $(2, \ldots, 2)$-special-unsoundness with an exponential parameter B. Note that, while the *honest* prover is *deterministic*, a

[9] The soundness and knowledge error of a single invocation of Π are both equal to $\mathrm{Er}(\mathbf{k};N)$. Therefore, it immediately follows that the soundness error of the parallel repetition Π^t is $\mathrm{Er}(\mathbf{k};N)^t$. The fact that the knowledge error of Π^t also equals $\mathrm{Er}(\mathbf{k};N)^t$ follows from the recent work [7].

dishonest prover can produce different messages (and hope to be lucky with one of the corresponding challenges).

The following theorem gives a lower bound for the success probability of our attack on the Fiat-Shamir transformation $\mathsf{FS}[\Pi^t]$ of the t-fold parallel repetition Π^t of an interactive proof Π with certain common soundness and unsoundness properties.

Theorem 3. *Let Π be a $(2\mu+1)$-move $(k,\ldots,k)$-out-of-$(N,\ldots,N)$ special-sound interactive proof that is $(\ell,\ldots,\ell)$-special-unsound with B responses per round for $\ell = k-1$. Furthermore, let $t, Q \in \mathbb{N}$ be integer multiples of μ such that $Q \cdot \left(\frac{\ell}{N}\right)^{t/\mu} \leq 1/4$ and $B \geq Q$. Then there exists a Q-query dishonest prover $\mathcal{P}^*$ against $(\mathcal{P},\mathcal{V}) = \mathsf{FS}[\Pi^t]$ such that, for any statement $x \in \{0,1\}^*$,*

$$\epsilon(\mathcal{P}^*, x) = \Pr\left(\mathcal{V}^{\mathsf{RO}}(x, \mathcal{P}^{*,\mathsf{RO}}) = 1\right) \geq \frac{1}{2}\frac{Q^\mu}{\mu^{t+\mu}}\,\mathrm{Er}(\mathbf{k}; N)^t\,.$$

The run-time of $\mathcal{P}^*$ is at most tQ times the run-time of attack strategy $\mathcal{A}$.

Proof. The basic idea of the attack is that (groups of) parallel threads can be attacked individually and independently from each other over the different rounds of the protocol. Concretely, the attack is given by the adversary $\mathcal{P}^*$ against $\mathsf{FS}[\Pi^t]$, which makes up to $Q = \mu \cdot Q'$ queries, defined as follows: $\mathcal{P}^*$ runs attack strategy $\mathcal{A}$ in parallel against all $t = \mu \cdot t'$ threads. Let us call a thread *green* if strategy $\mathcal{A}$ succeeds in guessing the challenge for that thread (and hence, $\mathcal{V}$ will eventually accept for that thread). Otherwise, a thread is *red*. All threads start out red, and the goal of $\mathcal{P}^*$ is to turn all threads green. To do so, in every round $\mathcal{P}^*$ tries to turn at least $t' = t/\mu$ red threads into green threads (or all red threads into green threads if fewer than t/μ remain). For this, $\mathcal{P}^*$ uses $\mathcal{A}$ to get the messages which it feeds to the random oracle. If $\mathcal{P}^*$ was lucky with the received challenges for at least $t' = t/\mu$ threads, then enough red threads turn green. Else, $\mathcal{P}^*$ tries the considered round again, exploiting that $\mathcal{A}$ can produce up to B distinct messages that give him a chance, each one giving a fresh challenge from the random oracle. The dishonest prover $\mathcal{P}^*$ tries up to $Q' = Q/\mu$ times per round until it gives up (and fails).

The number of queries $\mathcal{P}^*$ makes to the random oracle is at most Q, hence $\mathcal{P}^*$ is a Q-query adversary. The probability that $\mathcal{P}^*$ succeeds for any try in any round to turn at least $t' = t/\mu$ red threads into green threads is at least $(\frac{\ell}{N})^{t'} = \lambda^{t'}$, where we introduce $\lambda = \frac{\ell}{N}$ to simplify the upcoming expressions. Therefore, since $\mathcal{P}^*$ makes at most $Q' = Q/\mu$ queries in every round, the success probability for any fixed round is at least

$$1 - \left(1-\lambda^{t'}\right)^{Q'} \;\geq\; Q'\lambda^{t'} - 2\,Q'^2\lambda^{2t'} \;=\; Q'\lambda^{t'}\left(1 - 2\,Q'\lambda^{t'}\right). \tag{6}$$

where the inequality follows from the fact that $1-(1-x)^n \geq nx - 2n^2x^2$, which can be shown to hold when $nx \leq 1/2$, which is (more than) satisfied for $x = \lambda^{t'}$ and $n = Q'$ by assumption. Hence, $\mathcal{P}^*$ succeeds (in all μ rounds) with probability at least

$$Q'^\mu\lambda^t\left(1 - 2\,Q'\lambda^{t'}\right)^\mu \;\geq\; Q'^\mu\lambda^t\left(1 - 2Q\lambda^{t'}\right) \;\geq\; \frac{1}{2}Q'^\mu\lambda^t\,,$$

where we use that $(1-z)^n \geq 1-nz$ for $n \in \mathbb{N}$ and $z \in [0,1]$ to argue the first inequality, and $Q \cdot \left(\frac{\ell}{N}\right)^{t'} \leq 1/4$ for the second. To complete the analysis of $\mathcal{P}^*$'s success probability, we observe that

$$\mathrm{Er}(\mathbf{k}; N) = 1 - \left(1 - \frac{k-1}{N}\right)^{\mu} \leq \mu \cdot \frac{k-1}{N} = \mu \cdot \frac{\ell}{N} = \mu \cdot \lambda\,.$$

Hence, the success probability of $\mathcal{P}^*$ is at least $\frac{1}{2}Q'^{\mu}\left(\frac{\mathrm{Er}(\mathbf{k};N)}{\mu}\right)^t$, as claimed. □

Acknowledgments

The first author was supported by EU H2020 project No. 780701 (PROMETHEUS) and the Vraaggestuurd Programma Cyber Security & Resilience, part of the Dutch Top Sector High Tech Systems and Materials program. The third author was supported by the topic Engineering Secure Systems (46.23.01) of the Helmholtz Association (HGF) and by KASTEL Security Research Labs.

References

1. Full version of this paper. IACR ePrint 2021/1377
2. Albrecht, M.R., Lai, R.W.F.: Subtractive sets over cyclotomic rings - limits of Schnorr-like arguments over lattices. In: CRYPTO. pp. 519–548 (2021)
3. Ames, S., Hazay, C., Ishai, Y., Venkitasubramaniam, M.: Ligero: Lightweight sublinear arguments without a trusted setup. In: CCS. pp. 2087–2104. ACM (2017)
4. Attema, T., Cramer, R.: Compressed Σ-protocol theory and practical application to plug & play secure algorithmics. In: CRYPTO. pp. 513–543 (2020)
5. Attema, T., Cramer, R., Kohl, L.: A compressed Σ-protocol theory for lattices. In: CRYPTO. pp. 549–579 (2021)
6. Attema, T., Cramer, R., Rambaud, M.: Compressed Σ-protocols for bilinear group arithmetic circuits and application to logarithmic transparent threshold signatures. In: ASIACRYPT. pp. 526–556 (2021)
7. Attema, T., Fehr, S.: Parallel repetition of $(k_1, \ldots, k_\mu)$-special-sound multi-round interactive proofs. In: CRYPTO (2022)
8. Barak, B., Lindell, Y.: Strict polynomial-time in simulation and extraction. In: STOC. pp. 484–493 (2002)
9. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: CCS. pp. 390–399 (2006)
10. Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for R1CS. In: EUROCRYPT. pp. 103–128 (2019)
11. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: TCC. pp. 31–60 (2016)
12. Block, A.R., Holmgren, J., Rosen, A., Rothblum, R.D., Soni, P.: Time- and space-efficient arguments from groups of unknown order. In: CRYPTO. pp. 123–152 (2021)

13. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In: EUROCRYPT. pp. 327–357 (2016)
14. Bootle, J., Lyubashevsky, V., Nguyen, N.K., Seiler, G.: A non-PCP approach to succinct quantum-safe zero-knowledge. In: CRYPTO. pp. 441–469 (2020)
15. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: S&P. pp. 315–334 (2018)
16. Bünz, B., Fisch, B., Szepieniec, A.: Transparent SNARKs from DARK compilers. In: EUROCRYPT. pp. 677–706 (2020)
17. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D., Wichs, D.: Fiat-Shamir: from practice to theory. In: STOC. pp. 1082–1090. ACM (2019)
18. Chiesa, A., Manohar, P., Spooner, N.: Succinct arguments in the quantum random oracle model. In: TCC. pp. 1–29 (2019)
19. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: CRYPTO. pp. 186–194 (1986)
20. Ganesh, C., Khoshakhlagh, H., Kohlweiss, M., Nitulescu, A., Zajac, M.: What makes Fiat-Shamir zkSNARKs (updatable SRS) simulation extractable? In: SCN. pp. 735–760 (2022)
21. Gentry, C., Halevi, S., Lyubashevsky, V.: Practical non-interactive publicly verifiable secret sharing with thousands of parties. In: EUROCRYPT. pp. 458–487 (2022)
22. Ghoshal, A., Tessaro, S.: Tight state-restoration soundness in the algebraic group model. In: CRYPTO. pp. 64–93 (2021)
23. Goldreich, O.: The Foundations of Cryptography - Volume 2: Basic Applications. Cambridge University Press (2004)
24. Hoffmann, M., Klooß, M., Rupp, A.: Efficient zero-knowledge arguments in the discrete log setting, revisited. In: CCS. pp. 2093–2110 (2019)
25. Jaeger, J., Tessaro, S.: Expected-time cryptography: Generic techniques and applications to concrete soundness. In: TCC. pp. 414–443 (2020)
26. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S.: Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In: CCS. pp. 2111–2128 (2019)
27. del Pino, R., Lyubashevsky, V., Seiler, G.: Short discrete log proofs for FHE and ring-lwe ciphertexts. In: PKC. pp. 344–373 (2019)
28. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: EUROCRYPT. pp. 387–398 (1996)
29. Wahby, R.S., Tzialla, I., Shelat, A., Thaler, J., Walfish, M.: Doubly-efficient zkSNARKs without trusted setup. In: S&P. pp. 926–943 (2018)
30. Wikström, D.: Special soundness revisited. IACR ePrint 2018/1157 (2018)
31. Wikström, D.: Special soundness in the random oracle model. IACR ePrint 2021/1264 (2021)