On the Security Loss of Unique Signatures

Andrew Morgan[1(✉)] and Rafael Pass[2]

[1] Cornell University, Ithaca, USA
asmorgan@cs.cornell.edu
[2] Cornell Tech, New York City, USA
rafael@cornell.edu

Abstract. We consider the question of whether the security of unique digital signature schemes can be based on game-based cryptographic assumptions using *linear-preserving* black-box security reductions—that is, black-box reductions for which the security loss (i.e., the ratio between "work" of the adversary and the "work" of the reduction) is some *a priori* bounded polynomial. A seminal result by Coron (Eurocrypt'02) shows limitations of such reductions; however, his impossibility result and its subsequent extensions all suffer from two notable restrictions: (1) they only rule out so-called "simple" reductions, where the reduction is restricted to only *sequentially* invoke "straight-line" instances of the adversary; and (2) they only rule out reductions to non-interactive (two-round) assumptions. In this work, we present the first full impossibility result: our main result shows that the existence of *any* linear-preserving black-box reduction for basing the security of unique signatures on some *bounded-round* assumption implies that the assumption can be broken in polynomial time.

1 Introduction

Digital signature schemes, whereby a party can "sign" a message in a publicly verifiable yet still adversarially unforgeable way, are one of the most basic and important classes of cryptographic primitives; their security has been studied since the 1970s. While the earliest constructions of digital signatures [16,38,40,41] were heuristic in nature, modern constructions have tight proofs of security against all computationally bounded adversaries based on certain underlying assumptions.

Specifically, in a provably secure construction, we have a reduction $\mathcal{R}$, which, given any adversary $\mathcal{A}$ which breaks the security of a digital signature scheme Π, can break a certain underlying assumption $\mathcal{C}$; hence, if the assumption holds,

Supported in part by NSF Award CNS-1561209, NSF Award CNS-1217821, NSF Award CNS-1704788, AFOSR Award FA9550-18-1-0267, a Microsoft Faculty Fellowship, and a Google Faculty Research Award.

A. Beimel and S. Dziembowski (Eds.): TCC 2018, LNCS 11239, pp. 507–536, 2018.
https://doi.org/10.1007/978-3-030-03807-6_19

then the scheme must be secure. In this paper, we restrict our attention to *black-box* security reductions, where $\mathcal{R}$ only interacts with $\mathcal{A}$ as a "black box".[1] As far as we know, all security proofs for digital signatures rely on black-box security reductions.

We are interested in the *security loss* of such a reduction (a concept originally proposed as "security preservation" in [29]), or intuitively how "inefficient" it is in terms of running time and success probability compared to the adversary it runs. Informally, if, given a security parameter n, $\mathcal{R}$ (including the instances of $\mathcal{A}$ it runs) and $\mathcal{A}$ run in time $\mathsf{Time}_{\mathcal{R}^{\mathcal{A}}}(n)$ and $\mathsf{Time}_{\mathcal{A}}(n)$ respectively, and have success probabilities of $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$ and $\mathsf{Success}_{\mathcal{A}}(n)$, then the security loss is given by the maximum over all adversaries $\mathcal{A}$ of:

$$\lambda_{\mathcal{R}}(n) = \frac{\mathsf{Success}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)} \frac{\mathsf{Time}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Time}_{\mathcal{A}}(n)}$$

Intuitively, if we define the "work factor" of the adversary $\mathcal{A}$ to be the ratio of its running time to its success probability, or $\mathsf{Work}_{\mathcal{A}}(n) = \frac{\mathsf{Time}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{A}}(n)}$ (and respectively for $\mathcal{R}^{\mathcal{A}}$), then we can think of the security loss as

$$\lambda_{\mathcal{R}} = \frac{\mathsf{Work}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Work}_{\mathcal{A}}(n)}$$

or how much "work" the reduction $\mathcal{R}$ needs to do to break the underlying assumption compared to the amount of work that its adversary $\mathcal{A}$ does to break the primitive Π. So the higher the security loss, the easier the primitive is to break compared to the underlying assumption. As such, having security reductions with low security loss is essential for proving practical security of cryptographic primitives, since the security loss has a significant effect on the security parameter (i.e., the bit length of a key, size of a large prime for RSA, etc.) which must be used for the underlying assumption to achieve a particular level of security for the primitive.

The most efficient possible reductions are those which have *constant* security loss $\lambda_{\mathcal{R}}(n) \leq c$; these are commonly called *tight* reductions [5]. These reductions prove that $\mathsf{Work}_{\mathcal{A}}(n)$ is always directly proportional to $\mathsf{Work}_{\mathcal{R}^{\mathcal{A}}}(n)$, and so increasing the security parameter will always have the same effect on the security of the primitive as on the security of the underlying assumption.

A weaker notion of efficiency—introduced by Luby [29]—is that of a *linear-preserving* reduction, where the security loss is required to be bounded by some *a priori* fixed polynomial $p(\cdot)$ in just the security parameter; that is, $\lambda_{\mathcal{R}}(n) \leq p(n)$.

For instance, a security reduction that only runs the adversary $\mathcal{A}$ a fixed polynomial number of times (independent of A's running time and success probability) may not be tight, but is still linear-preserving.[2] While, with a linear-

[1] We note that Π need not be black-box itself in some underlying primitive; we only require the *reduction* $\mathcal{R}$ to be black-box.

[2] The name "linear-preserving" comes from the fact that $\mathsf{Work}_{\mathcal{R}^{\mathcal{A}}}$ is still linear in the quantity $\mathsf{Work}_{\mathcal{A}}(n)$, although it may depend polynomially on the security parameter n.

preserving reduction, the concrete security of the primitive Π is only comparable to that of the assumption if we use an increased security parameter for Π, Π still retains the same "asymptotic" security as the underlying assumption: for instance, if Π can be broken in time $poly(n) \cdot (2^{n/3})$, then so can the underlying assumption.

Unique Signatures. While the original provably secure construction of digital signatures in [21] was neither tight nor linear-preserving, more recent constructions [1,5,7,12,13,23] with tight reductions have been exhibited. However, while these modern constructions are quite efficient, they sacrifice some arguably important features of the original constructions in achieving this. Most notably, the earliest construction in [40] had the property that signatures were *unique*—that is, for every public key and every message, there exists at most one valid signature for that message. Whereas provably secure constructions of unique signatures exist [30,31], as well as constructions of verifiable random functions [9,26,31] (which [31] shows imply unique signatures), none of these have linear-preserving, let alone tight, security reductions. And unfortunately, for many recent applications of digital signatures (e.g., the recent applications to blockchains [19,34]), this uniqueness property is in fact necessary.

Can Unique Signatures Have Linear-Preserving Reductions? A natural question, given the fact that no linear-preserving reductions have been discovered, is whether a certain degree of security loss is *required* when proving the security of unique signatures. This question was first addressed in 2002 by Jean-Sébastien Coron in his seminal paper [14]. At a high level, Coron's goal was to demonstrate that any unique signature scheme with a black-box security reduction must have a security loss of $O(\ell(n))$, where $\ell(n)$ is the number of signing queries made by the adversary. This, in particular, would rule out all linear-preserving reductions for unique signature schemes because $\ell(n)$ depends on the specific adversary $\mathcal{A}$ and can be an arbitrarily large polynomial.

However, while Coron's proof rules out many "natural" reductions, it does not fully answer the question. In particular, it applies only to a quite restricted class of "simple" reductions which run the adversary in a "sequential straight-line" fashion—that is, they can run many instances of the adversary, but must run them sequentially (such that each must finish before the next starts) and cannot rewind the adversary. Furthermore, Coron's result applies only to reductions to the class of *non-interactive* (i.e., two-round) security assumptions (e.g., inverting a one-way function or breaking RSA). This latter restriction is necessary to some extent; if the security assumption may have arbitrarily many rounds, then it becomes trivial to base security on such an assumption (e.g., reducing the security of digital signatures to itself). However, there is still a large gap between non-interactive and "unbounded-round" security assumptions, leaving open the question of whether *bounded-round* assumptions [33] (that is, security assumptions modeled as security games with an *a priori* bounded number of communication rounds) can be used to prove the security of unique signatures.

Since Coron's seminal work, his result has been generalized to a number of related primitives [3,24], improved and simplified [3,27], and extended to rule out other notions of security tightness [42]. However, despite these extensions, improvements, and generalizations, the above restrictions—to simple reductions, and to non-interactive assumptions—have not yet been surmounted, leaving open the question:

Does there exist a linear-preserving security reduction for basing the security of unique digital signatures on some natural hardness assumption?

Main Theorem. In this work, we settle this question, ruling out *all* linear-preserving reductions from unique signatures to *any* bounded-round assumption.

Theorem 1 (Informal)**.** *There does not exist a linear-preserving black-box reduction from the security of some unique signature scheme Π to a bounded-round intractability assumption $\mathcal{C}$, unless $\mathcal{C}$ can be broken in polynomial time.*

More precisely, we show that, unless $\mathcal{C}$ can be broken in polynomial time, the security loss of any black-box reduction from the security of some unique signature scheme Π to any bounded-round intractability assumption $\mathcal{C}$ must be at least $O(\sqrt{\ell(n)})$, where $\ell(n)$ is the number of signature queries the adversary uses (and thus not a fixed polynomial independent of the adversary). Moreover, we observe (deferred to the full version) that our main theorem, with minor alterations, can also be applied to the related notion of *rerandomizable* signatures, or non-unique signatures with the property that signatures can be efficiently "rerandomized".

1.1 Proof Outline

In proving our main theorem, we follow the "meta-reduction" paradigm, originally pioneered in [8] (see also [2,4,6,10,18,22] for related work concerning meta-reductions), though we note that work on black-box separations using other frameworks dates back much farther, to [25]. The core idea behind this approach is, given an arbitrary black-box reduction $\mathcal{R}$ that breaks the assumption $\mathcal{C}$ by using black-box access to some "ideal adversary" $\mathcal{A}$ (which itself breaks the security of the primitive Π), to create an efficient adversary $\mathcal{B}$ which breaks $\mathcal{C}$ *without using* $\mathcal{A}$: $\mathcal{B}$ will internally run $\mathcal{R}$ and, roughly speaking, internally (and efficiently) "emulate" $\mathcal{A}$ for $\mathcal{R}$. The implication then is that if such a reduction $\mathcal{R}$ exists and breaks $\mathcal{C}$ using the inefficient adversary $\mathcal{A}$, then $\mathcal{B}$ would likewise break $\mathcal{C}$, but in polynomial time, proving that $\mathcal{C}$ is not a secure assumption. See Fig. 1 for an illustration of the mechanics of this paradigm.

Of course, we cannot prove complete non-existence (since, indeed, provably secure unique signature schemes exist); instead, we show that, unless $\mathcal{R}$ makes many queries to $\mathcal{A}$—which already implies that the security loss is high—then we can efficiently emulate $\mathcal{A}$'s responses with "high" (but not overwhelming) probability, which in turn will imply that $\mathcal{B}$ breaks $\mathcal{C}$ unless $\mathcal{R}$'s success probability is relatively small (again implying that its security loss is high).

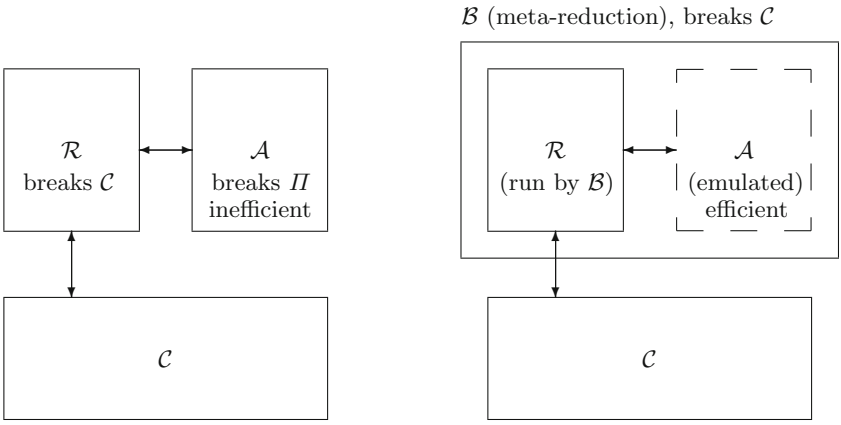

Fig. 1. The meta-reduction paradigm. *Left:* $\mathcal{R}$ breaks the assumption $\mathcal{C}$ by using the "ideal" but inefficient adversary $\mathcal{A}$ (against the signature scheme Π) as an oracle. *Right:* the meta-reduction $\mathcal{B}$ runs $\mathcal{R}$ (forwarding its communication with $\mathcal{C}$) and efficiently emulates $\mathcal{A}$ to break $\mathcal{C}$ with slightly less probability than in the left experiment.

Coron's Meta-reduction. As already mentioned, Coron, in [14], demonstrates how to employ this technique for a *restricted* class of reductions from the security of unique signatures. In particular, imagine an "ideal" inefficient adversary $\mathcal{A}$ which requests signatures for $\ell(n)$ randomly chosen messages, next uses a *brute-force search* to find a signature (i.e., a forgery) on a new random message, and finally returns the forgery. (Note that this adversary $\mathcal{A}$ is inefficient as it requires a brute-force search to recover the forgery.) In order to simulate $\mathcal{R}$'s interaction with $\mathcal{A}$ while running in polynomial time, $\mathcal{B}$ will run $\mathcal{R}$ normally and simulate $\mathcal{A}$ by first requesting signatures for $\ell(n)$ random messages. However, in order to extract a forgery without using brute force, $\mathcal{B}$ will pick a random message m^*, rewind the execution of $\mathcal{R}$ to before a randomly selected query, and try querying $\mathcal{R}$ for m^* instead of what it sent to $\mathcal{R}$ during the "main" (i.e., non-rewound) thread. If $\mathcal{R}$ returns a correct signature for m^* during this "rewinding", then $\mathcal{B}$ has succeeded in efficiently extracting a forgery and can return it to $\mathcal{R}$. In this case, $\mathcal{B}$ has succeeded in *perfectly* emulating $\mathcal{A}$; note that this relies on the fact that the signature scheme is unique and thus there exists at most one valid signature on m^*.

Of course, $\mathcal{R}$ may not always return a correct response to $\mathcal{A}$'s (or $\mathcal{B}$'s) queries; however, if $\mathcal{A}$ receives any incorrect responses in the "main" thread, it may simply return $\perp$ to $\mathcal{R}$ (as the security game for unique signatures only dictates that $\mathcal{A}$ must return a valid forgery when its queries are correctly answered), and so, in that case, $\mathcal{B}$ may also do so when emulating $\mathcal{A}$. The only time that $\mathcal{B}$ will fail to emulate $\mathcal{A}$ is, in fact, when $\mathcal{R}$ responds correctly to the original $\ell(n)$ queries by $\mathcal{B}$ during the "main" execution, but fails to respond to the rewound query of m^* in *any* rewinding (and so $\mathcal{B}$ can neither return $\perp$ or a forgery). Coron, through an elegant (yet complex) probabilistic argument, shows that the probability of

this "bad event" is bounded above by $O(1)/\ell(n)$. Intuitively, the reason this holds is that, unless $\mathcal{R}$ provides signatures to a fraction $O(1)/\ell(n)$ of random messages m^* (and thus the rewinding succeeds with probability $O(1)/\ell(n)$), it is unlikely that $\mathcal{R}$ provides correct signatures to all the $\ell(n)$ signature requests on the "main" thread, and in this case $\mathcal{B}$ does not need to provide a forgery to succeed in emulating $\mathcal{A}$. Of course, formalizing this argument is quite non-trivial, and Coron presents a sophisticated analysis which does so.

This argument rules out all reductions $\mathcal{R}$ from the security of unique signatures to a non-interactive security assumption which break the assumption with probability greater than the failure probability of $\mathcal{B}$—that is, $O(1)/\ell(n)$—assuming $\mathcal{R}$ runs a single instance of its adversary. If $\mathcal{R}$ runs multiple, $M(n)$, instances of its adversary in a sequential (i.e., non-interleaved) manner, then by the union bound over all instances, the failure probability bound becomes $O(M(n))/\ell(n)$. Furthermore, in this case $\mathsf{Time}_{\mathcal{R}^{\mathcal{A}}}(n) \geq M(n)\mathsf{Time}_{\mathcal{A}}(n)$, and so (given $\mathsf{Success}_{\mathcal{A}}(n) = 1$) Coron's argument achieves a bound of

$$\lambda_{\mathcal{R}}(n) \geq O(\ell(n))$$

which thus rules out all linear-preserving reductions as there is no *a priori* polynomial bound on $\ell(n)$.

We note that while Coron's proof relies on a subtle and non-trivial analysis, a very recent and elegant work by Bader et al. [3] presents a much simpler proof of Coron's theorem. In their approach, however, they consider a quite different ideal adversary $\mathcal{A}'$, which is even more tailored to simple reductions and non-interactive assumptions.[3] Consequently, we focus on Coron's original approach, which we shall see can be generalized to deal with all reductions and all bounded-round assumptions.

The Problem with Interactive Assumptions and "Nesting". Note that, in the above argument, it is crucial that the security assumption is non-interactive. Otherwise, when we rewind $\mathcal{R}$, $\mathcal{R}$ may send a new message to $\mathcal{C}$ and may require a response before proceeding; if this happens, we can no longer perform the emulation (as we cannot rewind the communication with $\mathcal{C}$).

Additionally, the argument crucially relies on the fact that $\mathcal{R}$ talks to $\mathcal{A}$ in a "straight-line" fashion, and only considers sequential interactions with (multiple instances) of $\mathcal{A}$. If we did not have that restriction, $\mathcal{R}$ might simultaneously run multiple instances of $\mathcal{A}$ and "nest" these different executions. For instance, it might be the case that $\mathcal{R}$ receives a query from a particular instance of $\mathcal{A}$, begins an entirely different instance of $\mathcal{A}$ (or perhaps even multiple instances), makes queries, and potentially requests a forgery, all before returning a response to the first query. Rewinding this will be troublesome because, depending on the query, $\mathcal{R}$ could respond differently to the nested queries or even follow an entirely different execution pattern. Even more worrying is the fact that, if there

[3] In fact, whereas it is not clear whether Coron's meta-reduction $\mathcal{B}$ fails under these more general conditions, the meta-reduction from [3] trivially breaks down under them.

are enough levels of nesting, rewinding every query for every instance may take super-polynomial time, which would invalidate the construction of $\mathcal{B}$ (since an inefficient adversary would not contradict the assumption $\mathcal{C}$). See Fig. 2 for an example illustrating this.

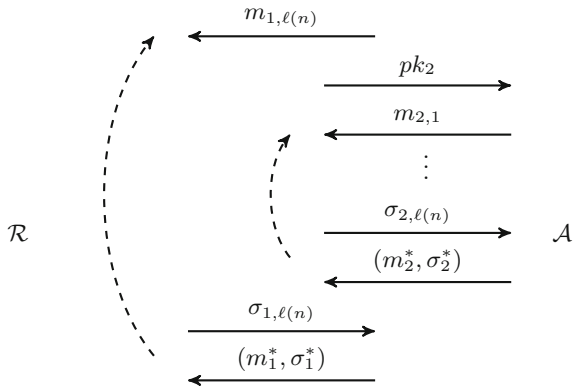

Fig. 2. A simple example of nested rewinding by $\mathcal{B}$ that might occur during interaction between $\mathcal{R}$ and two concurrent simulated instances of $\mathcal{A}$. Note that the inner rewinding must occur twice, once before the outer rewinding and once after, as the public key pk_2 might change based on the message $m_{1,\ell(n)}$. In fact, with m concurrent instances, up to 2^m rewindings may occur in this fashion.

Interestingly, this problem is also prevalent in research concerning concurrent zero-knowledge [17]. This connection was already noted in the earlier impossibility result for black-box reductions of [33], where a "recursive rewinding strategy" (similar to [11,15,36,39]) is used to overcome this problem. The core idea behind this technique is to rewind *every* relevant query, but to "abort" the rewinding when "too many" nested queries take place during the rewinding. The limit on nested queries furthermore decreases by a factor of n with each recursive level of nesting, so, since the total number of messages is polynomial, the number of levels will always be bounded by a constant, providing the polynomial bound on running time.

One might be tempted to simply apply this technique directly to the problem at hand; unfortunately, due to a fundamental difference between the two results, this is not possible. Specifically, in [33], the result proven is a complete impossibility (and not just a bound on the security loss): more precisely, emulation of $\mathcal{A}$ can be shown to succeed with overwhelming probability. In our context, however, the probability that the emulation succeeds is some inverse polynomial (and inherently must be so, or else we would have shown a complete impossibility of black-box security reductions). The problem then with a recursive rewinding strategy is that the failure probabilities may "cascade". Additionally, we cannot rely on the technique from [33] of repeatedly rewinding until we obtain a correct

response, since that would bias the distribution of the message m^* on which we output a forgery!

A Simple Rewinding Technique. We deal with the problem by using a different (and actually much simpler) rewinding strategy, inspired by a technique for "bounded-concurrent" zero-knowledge arguments originally introduced by Lindell [28]. The key observation is that rewinding is only necessary when $\mathcal{B}$ encounters an "end message" (i.e., $\mathcal{R}$ requesting a forgery from an instance of $\mathcal{A}$). If, during the rewinding to extract a forgery for some instance of $\mathcal{A}$, $\mathcal{B}$ avoids queries that contain an end message for a separate instance of $\mathcal{A}$ (i.e., those that would cause recursive rewinding), then it becomes straightforward to bound the number of rewindings and show that $\mathcal{B}$ will run in polynomial time; as an added advantage, this allows us to treat end messages very similarly to external communication in the analysis of our meta-reduction.

However, while this simulation strategy at first glance may seem straightforward (and, indeed, in the context of bounded-concurrent zero-knowledge, the analysis is simple), our scenario presents multiple major differences that make it quite non-trivial. In particular (as already mentioned above), unlike for zero-knowledge, we can no longer rewind queries with arbitrary messages; instead, in order to generate a forgery of a uniformly random message, we must choose a single forgery target m^* and rewind every query *only once* using this *same* message m^* (otherwise, as mentioned above, we would bias the distribution of the message m^* on which we output a forgery). Thus, to not bias the distribution of m^*, we must rewind each query with the same message m^* and consequently, we no longer have any independence between the rewindings, which severely complicates the analysis. (Indeed, recall that even in the simplified setting of Coron, his argument is already quite non-trivial).

Towards dealing with this, we present a new way of analyzing an ideal adversary which is quite similar to Coron's ideal adversary. Our analysis relies on a "randomness-switching" argument similar in spirit to that of [35,37], where we demonstrate that any "bad" sequence of randomness which causes the meta-reduction to fail can be permuted into many "good" sequences for which the meta-reduction succeeds. In particular, recall from our above discussion of Coron's meta-reduction that, if a sequence of messages is such that $\mathcal{B}$ fails to emulate $\mathcal{A}$, then *all* of its rewindings with the forgery target m^* must fail to extract a signature for the query to m^*. Hence, every rewound sequence beginning with a prefix $(m_1, \ldots, m_{i-1}, m^*)$ will either contain nesting or external communication during the query for m^*, or is such that $\mathcal{R}$ provides an incorrect signature for m^*. In the latter case, we can conclude as above that $\mathcal{A}$ and $\mathcal{B}$, having received an invalid signature, will both accordingly return $\perp$ (meaning that in fact $\mathcal{B}$ will succeed) if any sequence with that prefix is given in the (non-rewound) execution; thus, any such sequence cannot itself be a "bad" sequence. This, combined with the fact that the amount of nesting and external communication (and hence the possible number of rewound sequences which do *not* fall into the latter category) is bounded, will allow us to derive an upper bound for the possible number of "bad" sequences of randomness and hence for the prob-

ability of one of these sequences occurring (and causing the meta-reduction $\mathcal{B}$ to fail). This failure probability bound for $\mathcal{B}$ will ultimately allow us to upper-bound the success probability, and hence lower-bound the security loss, of the reduction $\mathcal{R}$. (In the full version, as an independent contribution and a warm-up, we sketch how an even simpler randomness-switching approach can be used to provide a simplified proof of a generalization of Coron's theorem to arbitrary reductions with *static* scheduling—where the order in which the reductions sends its sends messages is a-priori fixed).

Overview. In Sect. 2 we present key notation and definitions to be employed in our proof. We present and discuss our main result in Sect. 3, construct our "ideal" adversary $\mathcal{A}$ in Sect. 3.1, construct the meta-reduction $\mathcal{B}$ in Sect. 3.2, and complete our analysis and proof of the main theorem in Sect. 3.3. Lastly, we defer a synopsis of related work, as well as the details of an extension of our theorem to rerandomizable signatures, to the full version of this paper.

2 Preliminaries and Definitions

2.1 Notation

Let $\mathbb{N}$ denote the set of natural numbers (positive integers), and let $[n]$ denote the set of natural numbers at most n, or $\{1, 2, \ldots, n\}$. For $n \in \mathbb{N}$, we denote by 1^n the string of n ones, which will be used to provide a security parameter as input to an algorithm (this is by convention, so that the input length is bounded below by the security parameter). Given a set $S = \{s_1, \ldots, s_n\}$ of distinct elements, we shall let $|S|$ denote the number of elements n in S, and we refer to the set $\mathit{\Pi}_n(S)$ as the set of *permutations* of S, which contains any sequence which itself contains, in any order, each element of S exactly once.

When we say that a statement holds "for all sufficiently large $n \in \mathbb{N}$", by this we indicate that there exists an $N \in \mathbb{N}$ such that, for any integer $n \geq N$, the statement holds for n.

We recall that a function $\epsilon(\cdot)$ is *negligible* if, for any polynomial $p(\cdot)$, $\epsilon(n) < 1/p(n)$ for all sufficiently large $n \in \mathbb{N}$—that is, if $\epsilon(\cdot)$ is asymptotically smaller than any inverse polynomial. (For instance, an inverse exponential such as e^{-cn} is negligible in n for any constant $c > 0$).

Lastly, we assume a basic level of familiarity with the concepts of probabilistic algorithms and interactive Turing machines [20]. We will let $\mathcal{R}^{\mathcal{A}}(x)$ denote the probability distribution over the output of an oracle algorithm $\mathcal{R}$ given oracle access to a probabilistic $\mathcal{A}$. If $\mathcal{A}$ is a (deterministic) interactive algorithm, we instead assume $\mathcal{R}$ has oracle access to the function that, given the current partial transcript of (i.e., all messages sent up to a certain point in) interaction between $\mathcal{R}$ and $\mathcal{A}$, returns $\mathcal{A}$'s next message to $\mathcal{R}$.

Furthermore, we shall refer by $\langle \mathcal{A}, \mathcal{C} \rangle(x)$ to the probability distribution over the output of $\mathcal{C}$ after interaction between probabilistic interactive Turing machines $\mathcal{A}$ and $\mathcal{C}$, both given common input x (where the common input is also provided to any oracles, e.g., to $\mathcal{O}$ if $\mathcal{A}$ is an oracle machine given by $\mathcal{A}^{\mathcal{O}}$); the

view of the respective experiment, or the transcript of all messages sent and all randomness consumed, shall be denoted as $[\mathcal{A} \leftrightarrow \mathcal{C}](x)$.

2.2 Unique Signatures

First, we define unique signature schemes. Recall that a signature scheme is a means by which a message can be signed with the signer's secret key and the signature can be verified using a public key. A unique signature scheme, then, is simply a signature scheme for which each message can only have one possible signature:

Definition 1. *A* ***unique signature scheme*** *is a triple* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *of probabilistic polynomial-time algorithms such that, for every* $n \in \mathbb{N}$*:*

- Gen*, on input* 1^n*, produces a pair* (pk, sk)*.*
- Sign*, on input* (sk, m) *for any* $m \in \{0,1\}^n$*, produces a signature* σ*. (We write* $\sigma \leftarrow \mathsf{Sign}_{sk}(m)$*).*
- Ver*, on input* (pk, m, σ)*, produces either* Accept *or* Reject*. (We write out* $\leftarrow \mathsf{Ver}_{pk}(m, \sigma)$*).*

and, in addition, the following properties hold:

- *Correctness: For every* $n \in \mathbb{N}$ *and* $m \in \{0,1\}^n$*:*

$$Pr[(pk, sk) \leftarrow \mathsf{Gen}(1^n) : \mathsf{Ver}_{pk}(m, \mathsf{Sign}_{sk}(m)) = \mathsf{Accept}] = 1$$

- *Uniqueness: For every* $m \in \{0,1\}^*$*, and* $pk \in \{0,1\}^*$*, there exists at most one* $\sigma \in \{0,1\}^*$ *for which* $\mathsf{Ver}_{pk}(m, \sigma) = \mathsf{Accept}$*.*

We next turn to discussing what it means for such a scheme to be secure. A natural definition of security is the notion of *existential unforgeability against adaptive chosen-message attacks* [21], which requires that an adversary knowing the public key, even if allowed to adaptively choose a bounded number of messages and observe their signatures, is unable to forge any signature for a message they have not yet queried. We formalize this by allowing the adversary access to an oracle for Sign, as follows:

Definition 2. *We say that a signature scheme is* ***unforgeable*** *if, for every non-uniform probabilistic polynomial-time oracle-aided algorithm* $\mathcal{A}$*, there is some negligible function* $\epsilon(\cdot)$ *such that for all* $n \in \mathbb{N}$*:*

$$Pr\left[(pk, sk) \leftarrow \mathsf{Gen}(1^n); (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}_{sk}(\cdot)}(1^n, pk) : \mathsf{Ver}_{pk}(m, \sigma) = \mathsf{Accept} \wedge \mathsf{Valid}\right] \leq \epsilon(n)$$

where Valid *is the event that none of* $\mathcal{A}$*'s queries were for the signature of the output message* m*.*

We will define a weaker notion of a signature scheme being $\ell(\cdot)$*-****unforgeable*** *identically to the above, with the exception that* Valid *is the event that the following* two *conditions on* $\mathcal{A}$ *are true:*

– *$\mathcal{A}$ has queried its oracle at most $\ell(n)$ times.*
– *None of $\mathcal{A}$'s queries were for the signature of the output message m.*

The bounded notion of $\ell(\cdot)$-unforgeability is primarily useful to prove our concrete security loss bound, whereas our main result applies to the general notion of unforgeability. Furthermore, for the purposes of the impossibility result, we weaken the definition of unforgeability to a worst-case definition, as this will strengthen our main theorem (by showing that basing even this weak notion of security on standard assumptions will incur a security loss):

Definition 3. *We say that a signature scheme is **weakly unforgeable** (respectively, weakly $\ell(\cdot)$-unforgeable) if, for every non-uniform probabilistic polynomial-time oracle-aided algorithm $\mathcal{A}$ and every $n \in \mathbb{N}$:*

$$Pr\left[(pk, sk) \leftarrow \mathsf{Gen}(1^n); (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}_{sk}(\cdot)}(1^n, pk) : \mathsf{Ver}_{pk}(m, \sigma) = \mathsf{Accept} \wedge \mathsf{Valid}\right] < 1$$

where Valid *is defined as above (and respectively for $\ell(\cdot)$-unforgeability). In particular, we say that a non-uniform probabilistic polynomial-time algorithm $\mathcal{A}$* breaks weak unforgeability *of a signature scheme* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *if the probability above is equal to 1.*

2.3 Intractability Assumptions

Next, we define intractability assumptions in a manner originally proposed in [32]. Formally, we can model an assumption as a "security game" involving an interaction between a probabilistic *challenger* $\mathcal{C}$ and *adversary* $\mathcal{A}$, after which $\mathcal{C}$ will output either Accept or Reject. We say that an adversary $\mathcal{A}$ breaks the assumption if $\mathcal{C}$ accepts with probability non-negligibly greater than a certain threshold.

For instance, an assumption that a function f is one-way could be modeled by a two-round interaction where $\mathcal{C}$ sends $\mathcal{A}$ the image $y = f(x)$ on a uniformly random input x, $\mathcal{A}$ sends a message x' to $\mathcal{C}$, and $\mathcal{C}$ accepts if and only if $f(x') = y$. In this case, $\mathcal{A}$ breaks the assumption if it inverts f (i.e., $\mathcal{C}$ accepts) with probability non-negligibly greater than zero.

As an example of an assumption that would have a non-zero threshold, an assumption that two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ are indistinguishable could be modeled by the two-round interaction where $\mathcal{C}$ picks a random $b \in \{0, 1\}$, sends $\mathcal{A}$ a sample from $\mathcal{D}_b$, receives b' from $\mathcal{A}$, and accepts if $b = b'$. Then $\mathcal{A}$ would only break the assumption if $\mathcal{C}$ accepts with probability non-negligibly greater than a threshold of $1/2$. Formally, we model these assumptions following [33]:

Definition 4. *For polynomial $r(\cdot)$, we denote an $r(\cdot)$-**round intractability assumption** by a pair $(\mathcal{C}, t(\cdot))$, where $t(\cdot)$ is a function and $\mathcal{C}$ is a probabilistic interactive algorithm with input 1^n and an* a priori *bound of $r(n)$ rounds of communication. We say that $(\mathcal{C}, t(\cdot))$ is **secure** if the following is true:*

For any non-uniform probabilistic polynomial-time interactive algorithm $\mathcal{A}$, there exists a negligible function $\epsilon(\cdot)$ such that, for all $n \in \mathbb{N}$:

$$Pr[\langle \mathcal{A}, \mathcal{C} \rangle(1^n) = \mathsf{Accept}] \leq t(n) + \epsilon(n)$$

Furthermore, we say that a specific $\mathcal{A}$ breaks *the assumption if the above inequality is not true with respect to that $\mathcal{A}$; in particular, for some polynomial $p(\cdot)$, we say that $\mathcal{A}$ breaks $(\mathcal{C}, t(\cdot))$ with probability $1/p(\cdot)$ if, for infinitely many $n \in \mathbb{N}$,*

$$Pr[\langle \mathcal{A}, \mathcal{C} \rangle(1^n) = \mathsf{Accept}] \geq t(n) + \frac{1}{p(n)}$$

We also call a pair $(\mathcal{C}, t(\cdot))$ a ***bounded-round intractability assumption*** *if there exists some polynomial $r(\cdot)$ such that $(\mathcal{C}, t(\cdot))$ is an $r(\cdot)$-round intractability assumption.*

We note that any standard cryptographic security assumption can be modeled as a pair $(\mathcal{C}, t(\cdot))$ of this form, including our definitions above of the security of signature schemes. (In this case, the threshold $t(n)$ would be zero, and $\mathcal{C}$ would have $r(n) = 2\ell(n) + 2$ rounds of communication, first generating (pk, sk) and sending pk to $\mathcal{A}$, then signing $\ell(n)$ messages for $\mathcal{A}$, and finally receiving (m, σ) and outputting the result of Ver, or Reject if it had already signed m.) In particular, this is why we require an *a priori* bound on the number of rounds $r(\cdot)$ of the assumption; it will allow us to avoid such trivial reductions as reducing unforgeability to itself (for which we could obviously not prove the impossibility result).

2.4 Black-Box Reductions

Finally, we briefly discuss what it means for one assumption to be based on another assumption. In particular, given two assumptions $(\mathcal{C}_1, t_1(\cdot))$ and $(\mathcal{C}_2, t_2(\cdot))$, basing the hardness of $\mathcal{C}_1$ on that of $\mathcal{C}_2$ in a black-box way would classically entail, given an arbitrary adversary $\mathcal{A}_2$ which can break $(\mathcal{C}_2, t_2(\cdot))$ constructing a polynomial-time procedure $\mathcal{A}_1$ that breaks $(\mathcal{C}_1, t_1(\cdot))$ through standard interactions with $\mathcal{A}_2$ (i.e., using $\mathcal{A}_2$ in a *black-box* manner).

Notably, there is no guarantee that $\mathcal{A}_1$ invoke $\mathcal{A}_2$ only once; it could be the case that there are polynomially many invocations, or even "nested" invocations (e.g., multiple concurrent invocations such that the rounds of communication may be interleaved or dependent on one another), of $\mathcal{A}_2$ during the execution of $\mathcal{A}_1$. We can formalize this by imagining $\mathcal{A}_1$ as a polynomial-time *reduction* $\mathcal{R}$ that has oracle access to the interactive algorithm $\mathcal{A}$ (formerly $\mathcal{A}_2$):

Definition 5. *We refer to a probabilistic polynomial-time oracle-aided algorithm $\mathcal{R}$ as a* ***black-box reduction for basing the hardness of assumption*** *$(\mathcal{C}_1, t_1(\cdot))$* ***on that of*** *$(\mathcal{C}_2, t_2(\cdot))$ if, given any deterministic $\mathcal{A}$ that breaks $(\mathcal{C}_1, t_1(\cdot))$, $\mathcal{R}^{\mathcal{A}}$ breaks $(\mathcal{C}_2, t_2(\cdot))$. We refer to such a black-box reduction as* ***fixed-parameter*** *if, given common input 1^n, $\mathcal{R}^{\mathcal{A}}$ queries $\mathcal{A}$ only on input 1^n.*

We notably restrict our attention to oracles that are deterministic (or have some fixed randomness), as this allows us to consider cases where the reduction $\mathcal{R}$ can rewind or restart its oracle. We shall also restrict our attention, similarly to [33], to the case of *fixed-parameter* reductions where $\mathcal{R}$ invokes its adversary $\mathcal{A}$ only using a single security parameter (i.e., $\mathcal{A}$ must be the same algorithm in each instance); in particular, for security parameter n, we allow $\mathcal{R}$ to run up to $M(n)$ instances of some parameterized adversary $\mathcal{A}(1^n)$. Lastly, we can also apply the above concept to our definition of weak $\ell(\cdot)$-unforgeability (and define a fixed-parameter reduction identically for this case):

Definition 6. *We say that a probabilistic polynomial-time oracle-aided algorithm $\mathcal{R}$ is a* black-box reduction for basing weak unforgeability of a signature scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ on the hardness of an assumption $(\mathcal{C}, t(\cdot))$ *(resp. for weak $\ell(\cdot)$-unforgeability) if, for every deterministic algorithm $\mathcal{A}$ that breaks weak unforgeability of* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *(i.e., forges a signature with probability 1), there is a polynomial $p(\cdot)$ such that, for infinitely many $n \in \mathbb{N}$, $\mathcal{R}^{\mathcal{A}}$ breaks $(\mathcal{C}, t(\cdot))$ with probability $1/p(n)$.*

Finally, we wish to formalize the *security loss* of such a reduction $\mathcal{R}$, or the loss in the reduction's success probability proportionate to its time efficiency. We state this as follows:

Definition 7. *Let $\mathcal{R}$ be a black-box reduction for basing the hardness of assumption $(\mathcal{C}_1, t_1(\cdot))$ on that of $(\mathcal{C}_2, t_2(\cdot))$. Given any deterministic $\mathcal{A}$ and for each $n \in \mathbb{N}$:*

- *Let* $\mathsf{Success}_{\mathcal{A}}(n) = \Pr[\langle \mathcal{A}, \mathcal{C}_1 \rangle(1^n) = \mathsf{Accept}] - t_1(n)$ *(that is, the probability with which $\mathcal{A}$ breaks $(\mathcal{C}_1, t_1(\cdot))$, taken over all randomness of $\mathcal{A}$ and $\mathcal{C}_1$).*
- *Let* $\mathsf{Query}_{\mathcal{A}}(n)$ *denote the maximum, over all randomness of $\mathcal{A}$ and $\mathcal{C}_1$, of the possible number of messages sent from $\mathcal{C}_1$ to $\mathcal{A}$ during the experiment $[\mathcal{A} \leftrightarrow \mathcal{C}_1](1^n)$.*
- *Let* $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n) = \Pr[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}_2 \rangle(1^n) = \mathsf{Accept}] - t_2(n)$ *(that is, the probability with which $\mathcal{R}^{\mathcal{A}}$ breaks $(\mathcal{C}_2, t_2(\cdot))$ taken over all randomness of $\mathcal{A}$, $\mathcal{C}_2$, and $\mathcal{R}$).*
- *Let* $\mathsf{Query}_{\mathcal{R}^{\mathcal{A}}}(n)$ *denote the maximum, over all randomness of $\mathcal{A}$, $\mathcal{C}_2$, and $\mathcal{R}$, of the possible number of messages sent from $\mathcal{R}$ to $\mathcal{A}$ during the experiment $[\mathcal{R}^{\mathcal{A}} \leftrightarrow \mathcal{C}_2](1^n)$.*

Then we say that the ***security loss*** *of $\mathcal{R}$ is given by:*

$$\lambda_{\mathcal{R}}(n) = max_{\mathcal{A}} \left(\frac{\mathsf{Success}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)} \frac{\mathsf{Query}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Query}_{\mathcal{A}}(n)} \right)$$

Furthermore, we say that $\mathcal{R}$ is ***linear-preserving*** *if its security loss is bounded above by a fixed polynomial independent of $\mathcal{A}$—that is, there is a polynomial $p(\cdot)$ for which, for all sufficiently large $n \in \mathbb{N}$ and every $\mathcal{A}$, $\lambda_{\mathcal{R}}(n) \leq p(n)$.*

We note that, as we consider black-box reductions, we consider the ratio between the communication complexities of $\mathcal{R}$ and $\mathcal{A}$ as opposed to the running

times when determining the security loss. While many other recent works (e.g., [3,24]) use a definition which, though similar to the above, measures actual running time rather than rounds of communication, we note that our definition is at least as strong as time-based alternatives, and formally prove this fact in the full version.

3 Main Theorem

As our main theorem, we prove the following result:

Theorem 2. *Let* $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *be a unique signature scheme, and let* $(\mathcal{C}, t(\cdot))$ *be some* $r(\cdot)$*-round intractability assumption for polynomial* $r(\cdot)$*. If there exists some fixed-parameter black-box reduction* $\mathcal{R}$ *for basing weak unforgeability of* Π *on the hardness of* $(\mathcal{C}, t(\cdot))$*, then either:*

(1) $\mathcal{R}$ *is not a linear-preserving reduction, or*
(2) there exists a polynomial-time adversary $\mathcal{B}$ *that breaks* $(\mathcal{C}, t(\cdot))$*.*

We note that this result also applies to the slightly more general notion of *rerandomizable* signatures through an almost identical argument; we discuss this in more detail in the full version. Theorem 2 follows in a straightforward manner from the following lemma, which is a concrete security loss bound analogous to Coron's in [14], but generalized so that it handles arbitrary (i.e., not just simple) reductions:

Lemma 1. *Let* $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ *be a unique signature scheme, and let* $(\mathcal{C}, t(\cdot))$ *be some* $r(\cdot)$*-round intractability assumption for polynomial* $r(\cdot)$*. If for some polynomial* $\ell(\cdot)$ *there exists some fixed-parameter black-box reduction* $\mathcal{R}$ *for basing weak* $\ell(\cdot)$*-unforgeability of* Π *on the hardness of* $(\mathcal{C}, t(\cdot))$*, then either* $\mathcal{R}$*'s security loss is at least*

$$\lambda_{\mathcal{R}}(n) \geq \sqrt{\ell(n)} - (r(n) + 1)$$

for all sufficiently large $n \in \mathbb{N}$*, or there exists a polynomial-time adversary* $\mathcal{B}$ *that breaks the assumption* $(\mathcal{C}, t(\cdot))$*.*

Lemma 1 implies Theorem 2 by the definition of a linear-preserving reduction (we defer the formal proof to the full version, however). Hence, the remainder of the section is dedicated to proving Lemma 1. Our proof of Lemma 1 follows four major steps, which we shall describe here at a high level before beginning the full argument.

Constructing an Ideal Adversary. First, we describe an "ideal" adversary $\mathcal{A}$ which is *guaranteed* to break the security of Π while sending $\ell(n)$ queries, but does so by brute force and hence does not run in polynomial time. Our objective then is to create a meta-reduction $\mathcal{B}$ that *almost* always emulates the interaction $\mathcal{R}^{\mathcal{A}}$ between $\mathcal{R}$ and $\mathcal{A}$. If it does so with, say, probability $1 - 1/p(n)$, then

$\mathcal{B}$ will break the assumption $(\mathcal{C}, t(\cdot))$ with probability at least $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n) - 1/p(n)$. However, this means that $\mathcal{R}^{\mathcal{A}}$ itself cannot have success probability non-negligibly greater than $1/p(n)$; otherwise, $\mathcal{C}$ would be broken with non-negligible probability by $\mathcal{B}$.

The "ideal" $\mathcal{A}$ will pick $\ell(n)$ messages $(\vec{m})$ at random and query $\mathcal{R}$ for the signatures of each of these messages in turn, and will finally brute-force a secret key from the results and use that key to forge a signature for another random message m^*, which it will return. Crucially, $\mathcal{A}$ will also verify $\mathcal{R}$'s responses to its queries and return $\perp$ instead if not all are correct signatures. By construction $\mathcal{A}$ breaks $\ell(\cdot)$-weak unforgeability; however, due to the brute-force step, it (and consequently $\mathcal{R}^{\mathcal{A}}$) will not run in polynomial time.

Constructing a Meta-reduction. Hence, to *efficiently* emulate $\mathcal{R}^{\mathcal{A}}$, we create the meta-reduction $\mathcal{B}$. $\mathcal{B}$ will run $\mathcal{R}$ and forward communicate with $\mathcal{C}$ as normal; when $\mathcal{R}$ would start a new instance of its adversary $\mathcal{A}$, $\mathcal{B}$ will generate messages $\vec{m}$ and m^* randomly (i.e., identically to $\mathcal{A}$) and forward queries to $\mathcal{R}$ in the same manner as $\mathcal{A}$. However, when $\mathcal{R}$ requires an instance to provide a forged signature, $\mathcal{B}$ will also "rewind" the simulated execution to the start of each query for that instance and try to query $\mathcal{R}$ with m^* instead of the message it would normally query. If $\mathcal{R}$ gives a response to the rewound query, then $\mathcal{B}$ has (efficiently) found a forgery for m^*, which it can return to $\mathcal{R}$ when it requests a forgery from the corresponding instance of $\mathcal{A}$.

$\mathcal{B}$, while rewinding, will abort (and try rewinding the next slot instead) if either $\mathcal{R}$ would communicate externally with $\mathcal{C}$ (which $\mathcal{B}$ of course cannot rewind) or $\mathcal{R}$ would request a forgery for some other simulated instance of $\mathcal{A}$ during the simulated execution of $\mathcal{R}^{\mathcal{A}}$ (i.e., before responding to the rewound query). In particular, this strategy ensures that recursive rewinding as in [33] will not be required, since $\mathcal{B}$ will never attempt to start rewinding some instance while rewinding a different one.

Furthermore, $\mathcal{B}$ will "verify" all of $\mathcal{R}$'s responses to its signature queries (in the non-rewound part of the execution) in the same manner as $\mathcal{A}$, likewise returning $\perp$ from the simulated instance of $\mathcal{A}$ if not all responses are valid. So, whenever either $\mathcal{R}$ gives a simulated instance one or more incorrect responses or $\mathcal{B}$ successfully extracts a forgery (noting that, by the uniqueness property, the forgeries they return must be identical, which is crucial), $\mathcal{A}$ and $\mathcal{B}$'s simulations of $\mathcal{A}$ will be identically distributed to one another.

Bounding the Failure Probability. So, to bound the probability with which $\mathcal{B}$ does *not* successfully emulate some instance of $\mathcal{A}$, we must bound the probability that all of $\mathcal{B}$'s queries to $\mathcal{R}$ $(\vec{m})$ are correctly answered, yet the rewinding of *every one* of the queries fails due to either $\mathcal{R}$ responding badly to m^*, $\mathcal{R}$ communicating externally with $\mathcal{C}$, or a forgery request for another simulated instance of $\mathcal{A}$ occurring before $\mathcal{R}$ responds.

We bound this probability by using a counting argument similar to that exhibited in the introduction. In particular, we consider the messages $\vec{m}$ and m^*

for a particular instance, fixing the randomness outside of that instance arbitrarily. Then we show that, for any "bad" sequencing of these messages such that the non-rewound execution succeeds but every rewinding fails, many (though not all, because of the possibility for rewindings to fail due to an end message or external communication) of the rewindings of this sequence will correspond to "good" sequences where $\mathcal{B}$ returns $\perp$ due to receiving an incorrect response from $\mathcal{R}$.

What we intuitively show is that, in every set of $\ell(n)+1$ sequences corresponding to a sequence and its various rewindings, at most $M(n)+r(n)+1$ can be bad (since, informally, given a bad sequence, in expectation only $M(n)+r(n)$ of its rewindings can fail for reasons besides $\mathcal{R}$ responding incorrectly, i.e., due to nested end messages or external communication), where $M(n)$ is the maximum number of instances of $\mathcal{A}$ which $\mathcal{R}$ executes (which we show by our construction of $\mathcal{A}$ must be no less than the number of successful end messages). Hence we obtain a bound of $\frac{M(n)+r(n)+1}{\ell(n)+1}$ on the failure probability for each instance, which by the union bound over all $M(n)$ instances sums to an overall failure probability of less than $M(n)\left(\frac{M(n)+r(n)+1}{\ell(n)}\right)$.

Bounding the Security Loss. This does not immediately imply a bound on the security loss $\lambda_{\mathcal{R}}(n)$, since $M(n)$ can be arbitrarily large. However, as in the technical overview, we bound the security loss by showing that, if $M(n)$ is large, this requires a large enough running time of $\mathcal{R}$ that we still obtain a non-trivial lower bound on the security loss. Specifically, recalling that

$$\lambda_{\mathcal{R}}(n) \geq \frac{\mathsf{Success}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)} \frac{\mathsf{Query}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Query}_{\mathcal{A}}(n)}$$

we notice first that $\mathsf{Query}_{\mathcal{R}^{\mathcal{A}}}(n)/\mathsf{Query}_{\mathcal{A}}(n) \geq M(n)$, which follows (with some subtleties which we defer to the main proof) from the fact that $\mathcal{R}$ will in the worst case run $M(n)$ instances of $\mathcal{A}$. So, since $\mathsf{Success}_{\mathcal{A}}(n)=1$ by construction, and since, as we discussed previously, $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$ cannot be non-negligibly larger than the failure probability of $\mathcal{B}$, we have

$$\lambda_{\mathcal{R}}(n) \geq \frac{\ell(n)}{M(n)+r(n)+1}$$

which immediately implies the bound when $M(n) < \sqrt{\ell(n)} - (r(n)+1)$ (and so $\lambda_{\mathcal{R}}(n) > \sqrt{\ell(n)}$). On the other hand, we also know that $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n) \leq 1$ trivially, and so it is also the case that $\lambda_{\mathcal{R}}(n) \geq M(n)$, which implies the bound when $M(n) \geq \sqrt{\ell(n)} - (r(n)+1)$, completing the proof of Lemma 1 and hence Theorem 2.

3.1 The "Ideal" Adversary

We now proceed to the formal proof of Lemma 1. First, we exhibit an inefficient adversary $\mathcal{A}$ that will break weak $\ell(\cdot)$-unforgeability, so that we can later

construct an efficient $\mathcal{B}$ to simulate it while running $\mathcal{R}$ in order to break the assumption $(\mathcal{C}, t(\cdot))$.

Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a unique signature scheme, and let $(\mathcal{C}, t(\cdot))$ be some $r(\cdot)$-round intractability assumption for polynomial $r(\cdot)$. Assume that there exists some black-box reduction $\mathcal{R}$ for basing weak $\ell(\cdot)$-unforgeability of Π on the hardness of $(\mathcal{C}, t(\cdot))$ which, given an oracle breaking weak unforgeability, will break $(\mathcal{C}, t(\cdot))$ with probability $1/p(\cdot)$ for some polynomial $p(\cdot)$.

First, for any polynomial $\ell(n)$, we construct an inefficient but easily emulatable adversary $\mathcal{A}$ which sends at most $\ell(n)$ queries and is guaranteed to break weak unforgeability of Π. Since we will require $\mathcal{A}$ to be deterministic during execution yet generate random messages, we will assume that $\mathcal{A}$ is formally given by a deterministic interactive $\mathcal{A}^{\mathcal{O}}$ which has access to a random oracle $\mathcal{O}$ (of course, $\mathcal{O}$ is not needed for our actual constructions, as we shall emulate $\mathcal{A}$), which, as in [33], is given by a random variable which is uniformly distributed over functions $\{0,1\}^* \to \{0,1\}^\infty$. In particular, this ensures that the queries output by $\mathcal{A}^{\mathcal{O}}$ are uniformly distributed (i.e., over the randomness of $\mathcal{O}$), but are still preserved under rewinding.

We shall henceforth denote by $\mathcal{A}$ the specific adversary $\mathcal{A}^{\mathcal{O}}$ which, on input 1^n, behaves as described in Fig. 3. Informally, $\mathcal{A}$ makes $\ell(n)$ signature queries, generating the message for each query by applying $\mathcal{O}$ to the current partial transcript. Finally, after receiving responses for each query, $\mathcal{A}$ returns a brute-forced forgery, but only if it successfully "verifies" the transcript by ensuring that each query's response is valid and that each query in the transcript was generated in the correct manner (i.e., by $\mathcal{O}$ applied to the prior partial transcript).

It is straightforward to see that $\mathcal{A}$, given any fixed oracle $\mathcal{O}$, will break weak $\ell(\cdot)$-unforgeability; given an honest signing oracle (which will always send the correct partial transcript), it will always return some (m, σ) such that $\mathsf{Ver}_{pk}(m, \sigma) = \mathsf{Accept}$, m was not queried (as m^* is not equal to any of the queries m_i), and only $\ell(n)$ queries were made.

However, when interacting with $\mathcal{R}$, which is not bound by the rules of an honest oracle, the transcript verification is necessary to prevent $\mathcal{R}$ from "cheating" in certain ways during its interaction. First, we wish to ensure that $\mathcal{R}$ will return valid signatures to queries as often as possible. Also, we wish to ensure that $\mathcal{R}$ is actually required to answer $\ell(n)$ signature queries generated randomly by $\mathcal{A}$ and cannot, for instance, immediately send $\mathcal{A}$ an end message with an artificially generated transcript; this is done by using the oracle $\mathcal{O}$ to generate $\mathcal{A}$'s messages and ensuring that the transcript is consistent with the oracle. Formally, we make the following claim, which will be useful later:

Claim 1. *There exists a negligible function $\nu(\cdot)$ such that, for all $n \in \mathbb{N}$, the probability, over all randomness in the experiment $[\mathcal{R}^{\mathcal{A}^{\mathcal{O}}} \leftrightarrow \mathcal{C}](1^n)$, that some instance of $\mathcal{A}$ returns a forgery (i.e., something besides $\perp$) to $\mathcal{R}$ without having received $\ell(n)$ different responses to its signature queries from $\mathcal{R}$, is less than $\nu(n)$.*

The proof, which is straightforward, is deferred to the full version.

- Initially, receive a message pk, the public key; respond (i.e., generate m_1) according to the next step for $i = 1$.
- On receiving a message consisting of a partial transcript $\tau = (pk, m_1, \sigma_1, \cdots, m_{i-1}, \sigma_{i-1})$ for some $i \in [\ell(n)]$, do the following:
  - Generate m_i by taking the first n bits resulting from applying the oracle $\mathcal{O}$ to τ.
  - Return the new partial transcript $\tau || m_i$.
- On receiving a message consisting of a complete transcript $\tau = (pk, m_1, \sigma_1, \cdots, m_{\ell(n)}, \sigma_{\ell(n)})$ (we shall refer to such a message as a "forgery request" or "end message"), do the following:
  - Verify that, for each signature σ_i, $\mathsf{Ver}_{pk}(m_i, \sigma_i) = \mathsf{Accept}$. If not true for all i, return $\perp$.
  - Verify that, for each message m_i, m_i is equal to the first n bits resulting from applying the oracle $\mathcal{O}$ to the prefix transcript $\tau_{<i} = (pk, m_1, \sigma_1, \cdots, m_{i-1}, \sigma_{i-1})$. If not true for all i, then return $\perp$.
  - Finally, generate a random message m^* (distinct from each m_i in τ) by applying $\mathcal{O}$ to the transcript τ, use brute force to find a signature σ^* for which $\mathsf{Ver}_{pk}(m^*, \sigma^*) = \mathsf{Accept}$, and return the forgery (m^*, σ^*).

Fig. 3. Formal description of the "ideal" adversary $\mathcal{A}^{\mathcal{O}}$.

Furthermore, this construction of $\mathcal{A}$ (using the oracle $\mathcal{O}$) allows us to assume, without loss of generality, that the reduction $\mathcal{R}$ will never rewind an instance of $\mathcal{A}$—this is without loss of generality because there is a single accepting transcript for each choice of the oracle $\mathcal{O}$. Namely, given an oracle $\mathcal{O}$, if $\mathcal{R}$ always provides correct signatures, then $\mathcal{A}$'s messages (including the forgery it returns) and $\mathcal{R}$'s responses are fully determined by $\mathcal{O}$ and the uniqueness property of Π. Meanwhile, if $\mathcal{R}$ does not provide correct signatures, $\mathcal{A}$ will not return a forgery.

Because $\mathcal{A}$ breaks unforgeability, and by the assumed properties of $\mathcal{R}$ and the determinism of $\mathcal{A}^{\mathcal{O}}$ for any fixed oracle $\mathcal{O}$, it must be true that there exists polynomial $p(\cdot)$ such that

$$\Pr\left[\langle \mathcal{R}^{\mathcal{A}^{\mathcal{O}}}, \mathcal{C}\rangle(1^n) = \mathsf{Accept}\right] \geq t(n) + \frac{1}{p(n)}$$

for any oracle $\mathcal{O}$. As such, by the fact that $\mathcal{R}$ is fixed-parameter, we can observe that, for any n, averaging this probability over all possible oracles $\mathcal{O}$, we likewise have

$$\Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle(1^n) = \mathsf{Accept}\right] \geq t(n) + \frac{1}{p(n)}$$

even though $\mathcal{A}$ over a randomly-chosen $\mathcal{O}$ is not deterministic.

Of course, $\mathcal{A}$ is inefficient, so, in order to break the assumption $(\mathcal{C}, t(\cdot))$, we must construct an efficient $\mathcal{B}$ that is able to run $\mathcal{R}$ while emulating its interactions with $\mathcal{A}$ most of the time. Hence, the remainder of the proof will be dedicated

to constructing this meta-reduction and analyzing the probability with which it succeeds in emulating the "ideal" $\mathcal{A}$. Intuitively, if $\mathcal{B}$ successfully emulates $\mathcal{A}$ at least $1 - 1/p'(n)$ of the time for some function $p'(\cdot)$, then:

$$\left|\Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle(1^n) = \mathsf{Accept}\right] - \Pr\left[\langle \mathcal{B}, \mathcal{C}\rangle(1^n) = \mathsf{Accept}\right]\right| \leq \frac{1}{p'(n)}$$

$$\Pr\left[\langle \mathcal{B}, \mathcal{C}\rangle(1^n) = \mathsf{Accept}\right] \geq t(n) + \frac{1}{p(n)} - \frac{1}{p'(n)}$$

meaning that $\mathcal{B}$ must break $\mathcal{C}$ with probability at least $1/p(n) - 1/p'(n)$, as desired. Hence, what we shall effectively show in the subsequent steps is that, unless the security loss of $\mathcal{R}$ is large, $1/p'(n)$ will be non-negligibly smaller than $1/p(n)$, and thus $\mathcal{B}$ will break the security of $(\mathcal{C}, t(\cdot))$.

Slots. As a notational aside, we shall for simplicity henceforth refer to the pair of a query made by $\mathcal{A}$ (or something, such as $\mathcal{B}$, which emulates $\mathcal{A}$) and its corresponding response by $\mathcal{R}$ as a *slot* (v_{open}, v_{close}). Such a slot is determined by two views: the "opening" of the slot, or the view v_{open} of the execution of $\mathcal{R}$ immediately before $\mathcal{A}$'s query to $\mathcal{R}$, and the "closing" of the slot, or the view v_{close} of the execution immediately after $\mathcal{R}$ responds to the respective query. (We will also often refer to the view of $\mathcal{R}$ immediately *after* the opening query of a message m, which we shall denote by the concatenation $v_{open}||m$).

3.2 The Meta-reduction

We next construct the meta-reduction $\mathcal{B}$ which will efficiently emulate $\mathcal{A}$. Let $\mathcal{B}$ be as described formally in Fig. 4; informally, $\mathcal{B}$ will run $\mathcal{R}$ internally, forwarding communication to $\mathcal{C}$ as $\mathcal{R}$ would while also internally simulating instances of $\mathcal{A}$ interacting with $\mathcal{R}$. The primary difference between $\mathcal{B}$ and the "ideal" execution of $\mathcal{A}$ interacting with $\mathcal{R}$ is that $\mathcal{B}$, being restricted to polynomial time, cannot brute-force forgeries as $\mathcal{A}$ does; instead, while simulating each instance of $\mathcal{A}$, $\mathcal{B}$ will select at random a message m^* for which to forge a signature and attempt to rewind each slot for that instance, substituting m^* for the original message.[4]

If $\mathcal{R}$ ever returns a valid signature σ^* for m^*, then $\mathcal{B}$ may store that signature and finally return (m^*, σ^*) when $\mathcal{R}$ requests a forgery for that instance. However, if one of the following "bad events" occurs:

- $\mathcal{R}$ fails to return a valid signature of m^*.
- $\mathcal{R}$ asks for a forgery for another instance before returning a signature.
- $\mathcal{R}$ requires external communication with $\mathcal{C}$ (which cannot be rewound) before returning a signature.

then $\mathcal{B}$ will abort and try the next slot. In this way we circumvent the issue of having to recursively rewind nested end messages as in [33].

First, we can show that $\mathcal{B}$, unlike $\mathcal{A}$, is efficient:

[4] That is, when "rewinding" a slot (v_{open}, v_{close}), $\mathcal{B}$ will simulate interaction with $\mathcal{R}$ starting from the view $v_{open}||m^*$.

- Set initial view $v \leftarrow \perp$ and set $k \leftarrow 1$. Execute $\mathcal{R}$, updating the current view v according to the following rules.
- When $\mathcal{R}$ begins a new instance of $\mathcal{A}$ and sends a public key pk, label this instance as instance k. Generate and store $\ell(n)$ random queries $\vec{m}_k = (m_{k,1}, \ldots, m_{k,\ell(n)})$ and a target forgery m_k^*. (Abort and return Fail if m_k^* is equal to a message in $\vec{m}_k$.) Also let $pk_k \leftarrow pk$ and initialize the forgery $f_k \leftarrow \{\}$. Lastly, respond with $\tau_k^* = pk_k || m_{k,1}$ and increment k.
- When $\mathcal{R}$ attempts to communicate externally with $\mathcal{C}$, forward the message, return $\mathcal{C}$'s response to $\mathcal{R}$, and update v accordingly.
- When $\mathcal{R}$ sends a transcript $\tau = (pk, m_{I,1}, \sigma_{I,1}, \cdots, m_{I,j}, \sigma_{I,j})$ to some simulated instance I of $\mathcal{A}$, store the signature $\sigma_{I,j}$ and do the following:
  - If $j = \ell(n)$ (i.e., this is an end message), then do the following:
    - If τ is an inconsistent transcript (i.e., $m_{I,i}$ or $\sigma_{I,i}$ in τ is different from the stored $m_{I,i}$ or $\sigma_{I,i}$ (respectively) for some $i \in [\ell(n)]$, or not all $\sigma_{I,i}$ have been stored) or $\mathcal{R}$'s response to some signature query j was invalid (i.e., $\mathsf{Ver}_{pk_I}(m_{I,i}, \sigma_{I,i}) = \mathsf{Reject}$ for some $i \in [\ell(n)]$), then return $\perp$.
    - Otherwise, if f_I is still empty (i.e., not $\perp$), run the procedure Rewind detailed below for the instance I.
    - If, at this point, there is a stored forgery $f_I = (m_I^*, \sigma_I^*)$, then return it and continue executing $\mathcal{R}$ as above. Otherwise, abort the entire execution of $\mathcal{B}$ and return Fail.
  - If $\mathsf{Ver}_{pk_I}(m_{I,j}, \sigma_{I,j}) = \mathsf{Reject}$, then store $f_I \leftarrow \perp$.
  - Lastly, respond with $\tau || m_{I,j+1}$ and continue the execution of $\mathcal{R}$.

Rewind procedure:

- Given instance I, for $j \in [\ell(n)]$ let $(v_{open}^j, v_{close}^j)$ denote the slot corresponding to the j^{th} signature query for instance I.
- For each $j \in [\ell(n)]$, "rewind" the slot $(v_{open}^j, v_{close}^j)$ as follows: Let $k' \leftarrow k$, and begin executing $\mathcal{R}$ from the view $v' = v_{open}^j || m_I^*$ as in the main routine, with the following exceptions:
  - When $\mathcal{R}$ begins a new instance of $\mathcal{A}$, label this instance as instance k' and increment k'. (That is, continue creating new instances, but preserve the counter k in the outer execution for after the rewinding.)
  - When $\mathcal{R}$ attempts to communicate externally with $\mathcal{C}$, abort the rewinding and continue to the next j.
  - When $\mathcal{R}$ sends an end message for an instance $I' \neq I$ of $\mathcal{A}$, abort the rewinding and continue to the next j, *unless* $\mathcal{R}$ has not sent responses to $\ell(n)$ signature queries for I' (in which case reply with $\perp$).
  - If v' ever contains a message whose transcript contains a response σ_I^* to the query for m_I^*, then, if it is the case that $\mathsf{Ver}_{pk_I}(m_I^*, \sigma_I^*) = \mathsf{Accept}$, store $f_i \leftarrow (m_I^*, \sigma_I^*)$ and end the Rewind procedure (i.e., return to the outer execution); otherwise, if $\mathsf{Ver}_{pk_I}(m_I^*, \sigma_I^*) = \mathsf{Reject}$, store nothing to f_I and continue to the next j.

Fig. 4. Formal description of the meta-reduction $\mathcal{B}$.

Claim 2. *There exists a polynomial $t(n)$ such that, for all $n \in \mathbb{N}$, $\mathsf{Real}(1^n)$ is guaranteed to run in time at most $t(n)$.*

The proof, which is straightforward, is deferred to the full version.

Next, to reason about the failure probability of $\mathcal{B}$ (and through it the success probability of $\mathcal{R}^{\mathcal{A}}$), let us define the following experiments:

- Let $\mathsf{Ideal}(1^n)$ denote $[\mathcal{R}^{\mathcal{A}} \leftrightarrow \mathcal{C}](1^n)$—that is, the experiment where $\mathcal{R}(1^n)$, using the "ideal" adversary $\mathcal{A}(1^n)$ as a black box, communicates with $\mathcal{C}(1^n)$.
  - When we refer to probabilities in the context of this experiment, they are taken over a uniform distribution over random oracles $\mathcal{O}$ (which results in uniformly distributed messages $\vec{m}_I$ and m_I^*) for each instance I of $\mathcal{A}$ started by $\mathcal{R}$.
  - When we wish to fix the randomness of a particular execution of Ideal, we will denote this with the notation $\mathsf{Ideal}_{\{\mathcal{O}_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$, or, for more clarity, $\mathsf{Ideal}_{\{\vec{m}_I,m_I^*\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$. $\vec{m}_I$ and m_I^* are the messages and forgery generated for instance I by each oracle $\mathcal{O}_I$ given the (deterministic) prefix; $\mathcal{O}_{ext}$ is a random variable representing the random coins used by $\mathcal{R}$ and $\mathcal{C}$, containing a number of bits equal to the maximum number of coins needed (which must be polynomially many since $\mathcal{R}$ and $\mathcal{C}$ are polynomial-time). When all $\vec{m}_I$ and m_I^* are fixed, and $\mathcal{O}_{ext}$ is fixed, note that the execution of Ideal is deterministic for each instance.
- Let $\mathsf{Real}(1^n)$ denote $[\mathcal{B} \leftrightarrow \mathcal{C}](1^n)$—that is, the "real" experiment where $\mathcal{B}(1^n)$ communicates directly with $\mathcal{C}(1^n)$ by attempting to simulate the interaction between $\mathcal{A}(1^n)$ and $\mathcal{R}(1^n)$ while forwarding any external communications.
  - When we refer to probabilities in the context of this experiment, they are taken over uniformly distributed messages $\vec{m}_I$ and m_I^* for each simulated instance I of $\mathcal{A}$ started by $\mathcal{R}$ in the context of $\mathcal{B}$.
  - When we wish to fix the randomness of a particular execution of Real, we will again denote this with the notation $\mathsf{Real}_{\{\vec{m}_I,m_I^*\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$, where $\vec{m}_I$ and m_I^* are the messages and forgery for each simulated instance I of $\mathcal{A}$ and $\mathcal{O}_{ext}$ is again a random variable representing the random coins used by $\mathcal{R}$ and $\mathcal{C}$. When all $\vec{m}_I$ and m_I^* are fixed, and $\mathcal{O}_{ext}$ is fixed, note that the execution of Real is deterministic for each instance, just as with Ideal.
  - Furthermore, we may opt to isolate a particular simulated instance k by fixing all randomness *except for* that instance's; we denote this by $\mathsf{Real}^*_{\{\vec{m}_I,m_I^*\}_{-k},\mathcal{O}_{ext}}(1^n)$ and note that the probability space in this altered experiment is over uniformly distributed $\vec{m}_k$ and m_k^*. Further note that in experiment Real^* the execution up to the start of the isolated instance k is deterministic, as is the execution for any choice of $\vec{m}_k$ and m_k^*.
- In all experiments, we will denote the *view*, or *execution*, as the transcript of all messages sent between, and all randomness consumed by, real or simulated machines (i.e., between $\mathcal{R}$, $\mathcal{A}$ or $\mathcal{B}$'s simulation of $\mathcal{A}$, and $\mathcal{C}$). We will notate this using just the notation for the experiment, e.g., $\mathsf{Real}(1^n)$.

– In all experiments, we will denote the *result*, or *output*, as either the final output of $\mathcal{C}$ (either Accept or Reject) if $\mathcal{C}$ finishes, or as Fail if $\mathcal{C}$ does not finish (i.e., when $\mathcal{B}$ aborts returning Fail). This will be notated by Output, e.g., $\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Accept}$.

3.3 Analyzing the Meta-reduction

Using this terminology, we wish to show that $\mathsf{Real}(1^n)$ is identically distributed to $\mathsf{Ideal}(1^n)$ with high probability. To that end, we make the following claim:

Claim 3. *For all $n \in \mathbb{N}$:*

$$\begin{aligned}|\Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Accept}] - \Pr[\mathsf{Output}[\mathsf{Ideal}(1^n)] = \mathsf{Accept}]| \\ \leq \Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Fail}]\end{aligned}$$

Proof Sketch. Intuitively, this follows from the uniqueness property of Π; for any message m^*, there is only a single possible signature σ^*. Thus, given some setting of the randomness in the experiments, Real must proceed identically to Ideal (that is, each instance of $\mathcal{A}$ in Real will have an identical view to the corresponding simulated instance in Ideal) *unless* $\mathcal{B}$'s attempt to extract a forgery fails for some m^*, in which case by construction Real must return Fail. The complete proof is deferred to the full version.

We can use this claim to bound $\mathcal{R}^{\mathcal{A}}$'s success probability by bounding the probability that $\mathcal{B}$ will return Fail for some simulated instance I of $\mathcal{A}$. Let $M(n)$ be the maximum, over all randomness of $\mathcal{A}$, $\mathcal{C}$, and $\mathcal{R}$, of the number of instances of $\mathcal{A}$ that $\mathcal{R}$ runs to completion (i.e., for which it responds to all $\ell(n)$ queries) during the experiment $\mathsf{Real}(1^n)$. Then we show the following:

Proposition 1. *There exists a negligible function $\epsilon(\cdot)$ such that, for all $n \in \mathbb{N}$:*

$$\Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Fail}] \leq M(n)\left(\frac{M(n) + r(n) + 1}{\ell(n) + 1}\right) + \epsilon(n)$$

Proof. We first prove the following claim for any $\mathsf{Real}^*_{\{\vec{m}_I, m_I^*\}_{-k}, \mathcal{O}_{ext}}(1^n)$ (i.e., for any fixed setting of all randomness aside from $\vec{m}_k$ and m_k^*), and notice that, since it applies to arbitrarily fixed randomness, it must thus apply over all possible randomness of the experiment $\mathsf{Real}(1^n)$:

Claim 4. *There exists a negligible $\nu(\cdot)$ such that, given any setting of the randomness in the experiment $\mathsf{Real}^*_{\{\vec{m}_I, m_I^*\}_{-k}, \mathcal{O}_{ext}}(1^n)$, the probability, over the uniformly chosen messages $m_{k,1}, \ldots, m_{k,\ell(n)}, m_k^*$, that the simulated instance k will return Fail is, for all $n \in \mathbb{N}$, at most*

$$\frac{M(n) + r(n) + 1}{\ell(n) + 1} + \nu(n)$$

Proof. Let us begin by assuming that other simulated instances (besides k) of $\mathcal{A}$ in the experiment $\mathsf{Real}^*_{\{\vec{m}_I, m_I^*\}_{-k}, \mathcal{O}_{ext}}(1^n)$ will never return Fail (i.e., that they will "magically" produce a correct forgery in the case where they otherwise would return Fail). Clearly, this can only increase the probability that instance k will return Fail by ensuring that the experiment never aborts early.

Now let us consider the messages $\vec{m^*}_k \triangleq (m_{k,1}, \ldots, m_{k,\ell(n)}, m_k^*)$ in instance k; note that by the definition of $\mathsf{Real}^*_{\{\vec{m}_I, m_I^*\}_{-k}, \mathcal{O}_{ext}}(1^n)$ the execution is fully determined by $\vec{m^*}_k$. Let us also define for $i \in [\ell(n) - 1]$ the "rewound" sequence

$$\rho(\vec{m^*}_k, i) \triangleq (m_{k,1}, \ldots, m_{k,i-1}, m_k^*)$$

that is, the rewinding of $\vec{m}_k$ where the message in slot i is replaced by m_k^* to attempt to extract a forgery.

In order for $\mathcal{B}$ to return Fail, one of the following "bad events" must occur for each $i \in [\ell(n)]$:

- $E_1(\rho(\vec{m^*}_k, i))$: $\mathcal{R}$ fails to return a valid signature of m_I^* in the rewinding of the last slot in the sequence (slot i).
- $E_2(\rho(\vec{m^*}_k, i))$: During the rewinding of the last slot in the sequence (slot i), $\mathcal{R}$ asks for a forgery for another instance $k' \neq k$ before returning a signature for m_I^*, *or* $\mathcal{R}$ requires external communication with $\mathcal{C}$ (which cannot be rewound) before returning a signature for m_I^*.

In addition, for k to fail, the non-rewound execution of the instance must succeed, in that the event $E_1(\vec{m}_{k,\leq i})$ (where $\mathcal{R}$ fails to return a valid signature) *cannot* occur for any prefix $\vec{m}_{k,\leq i} = (m_{k,1}, \ldots, m_{k,i})$, where $i \in [\ell(n)]$.

Since, as we have noted, the behavior of k in $\mathsf{Real}^*_{\{\vec{m}_I, m_I^*\}_{-k}, \mathcal{O}_{ext}}(1^n)$ is fully determined by $\vec{m^*}_k$, every sequence $\vec{m^*}_k$ will deterministically either result in instance k returning something (either a forgery or $\perp$) or aborting and returning Fail; we shall refer to the former type of sequence (where k succeeds) as a "good" sequence, and the latter type as a "bad" sequence. To describe the relationship between these "good" and "bad" sequences, we first introduce the following terminology:

For any $k > 0$ and any arbitrary set $\vec{m}$ of k distinct messages in $[2^n]$, let $\Pi_k(\vec{m})$ denote the set of ordered permutations of the elements of $\vec{m}$. Given a sequence $\pi = (m_1, \ldots, m_{k-1}, m^*) \in \Pi_k(\vec{m})$, we let the "rewinding" operator ρ for $i \in [k-1]$ be defined as before—that is:

$$\rho(\pi, i) \triangleq (m_1, \ldots, m_{i-1}, m^*)$$

(Note that this is a sequence of length i.) We shall say that a sequence $a = (a_1, \ldots, a_{k-1}, a^*) \in \Pi_k(\vec{m})$ *blocks* a sequence $b = (b_1, \ldots, b_{k-1}, b^*) \in \Pi_k(\vec{m})$ with respect to some $i \in [k-1]$ if

$$\rho(a, i) = (b_1, \ldots, b_i)$$

that is, if a has a rewinding equivalent to a prefix of b. If we wish to denote that a blocks b with respect to a particular i, we shall say that a blocks b *in*

slot i. Furthermore, we will say that sequences $a^1, a^2, \ldots a^c \in \Pi_k(\vec{m})$ *c-block* a sequence $b \in \Pi_k(\vec{m})$ if there exist *distinct* $i_1, \ldots, i_c \in [k-1]$ such that, for any $j \in [c]$, a^j blocks b in slot i_j.

We next formalize the relationship between the "blocking" property and good/bad sequences that will allow us to use this property to bound the number of bad sequences that may occur. Specifically, we prove the following lemma:

Lemma 2. *Any sequence that is* $(M(n) + r(n) + 1)$*-blocked by bad sequences must be a good sequence.*

Proof. Consider a sequence $\vec{m'}_k$ which is $(M(n) + r(n) + 1)$-blocked by bad sequences. This means that $\vec{m'}_k$ must have $M(n) + r(n) + 1$ distinct slots i_j for which E_1 or E_2 occurs in its (non-rewound) execution, as the execution is at each of those points identical to a rewinding of one of the bad sequences by the fact that the sequence blocks $\vec{m'}_k$ in slot i_j. However, because at most $M(n)$ end messages (to completed instances of $\mathcal{A}$, note that others are answered with $\perp$) and at most $r(n)$ rounds of external communication can occur in any given execution, we observe that E_2 can happen for at most $M(n) + r(n)$ of these slots, and thus that E_1 must happen for at least one slot. In this case, we can deduce that $\vec{m'}_k$ must be a *good* sequence, because it must contain some slot for which $\mathcal{R}$ fails to return a correct response (meaning that $\mathcal{B}$ can successfully emulate $\mathcal{A}$ by returning $\perp$)—that is, the event $E_1(\vec{m}_{k,\leq i})$ must occur for that slot, which we have previously stated cannot be the case for bad $\vec{m^*}_k$. □

Consider, then, a set S of "bad" sequences $\vec{m^*}_k$ which are permutations of any set of $\ell(n) + 1$ distinct messages (i.e., an unordered set containing $\vec{m}_{k,i}$ and m^*_k). The following lemma, combined with Lemma 2, allows us to bound the size of such a set S:

Lemma 3. *Let* $\vec{m}$ *be an arbitrary set of* $\ell + 1$ *distinct messages in* $[2^n]$*, and let* $S \subset \Pi_{\ell+1}(\vec{m})$ *be a set of permutations of* $\vec{m}$*. If it is the case that, for some* $B \in \mathbb{N}$*, any member of* $\Pi_{\ell+1}(\vec{m})$ *which is* $(B+1)$*-blocked by a subset of* S *cannot itself lie in* S*, then* $|S| \leq (B+1)\ell!$

Proof. We begin with the following crucial claim: □

Subclaim 1. *No member of* $\Pi_{\ell+1}(\vec{m})$ *is* $(B+2)$*-blocked by a subset of* S*.*

Proof. Assume for the sake of contradiction that there exists some $\pi \in \Pi_{\ell+1}(\vec{m})$, $B+2$ sequences $\pi^1, \ldots, \pi^{B+2} \in S$, and $B+2$ distinct integers $i_1, \ldots, i_{B+2} \in [\ell]$ such that each partial sequence $\rho(\pi^j, i_j)$ is equivalent to the first i_j elements of π.

Assume without loss of generality that the integers i_j are in strictly ascending order. Consider the last sequence $\pi^{B+2} = (\pi_1^{B+2}, \ldots, \pi_*^{B+2})$; we shall show that π^{B+2} is $(M+1)$-blocked by sequences in S, leading to a contradiction because by definition no element of S can be $(M+1)$-blocked by other members of S.

We know that, since by assumption the first i_{B+2} elements of π are equivalent to $\rho(\pi^{B+2}, i_{B+2}) = (\pi_1^{B+2}, \ldots, \pi_{i_{B+2}-1}^{B+2}, \pi_*^{B+2})$, then the first $i_{B+2} - 1$ elements

of π^{B+2} and π must be identical. However, notice that, for any $j < B+2$, we have that $\rho(\pi^j, i_j)$ is identical to π in the first i_j elements, which, since by assumption $i_j \leq i_{B+2} - 1$, also indicates that $\rho(\pi^j, i_j)$ is identical to π^{B+2} in the first i_j elements.

This in turn implies that π^{B+2} is $(B+1)$-blocked by $\pi^1, \ldots, \pi^{B+1} \in S$, contradicting that $\pi^{B+2} \in S$ by the requisite property of S. □

So, we know that any member of S can be at most B-blocked by a subset of S, while any non-member can be at most $(B+1)$-blocked by a subset of S. We will combine this fact (an effective upper bound on the number of blocked sequences) with the subsequent claim (a respective lower bound) to derive our final bound on $|S|$.

Subclaim 2. *For each $i \in [\ell]$, there exist $|S|$ distinct sequences blocked in slot i by sequences in S.*

Proof. Beginning with $i = 1$, we observe that sequences with at least $|S|/\ell!$ different last elements m^* must occur in S (as there are only $\ell!$ sequences with any given last element). Furthermore, any sequence in S with a certain last element m^* must block in slot 1 a total of $\ell!$ different sequences (i.e., anything beginning with m^*), and different m^* will produce disjoint sets of sequences blocked. Thus, we conclude that the sequences in S will block in slot 1 at least $(|S|/\ell!)\ell! = |S|$ distinct sequences.

For the remaining slots $i > 1$, we can apply the same logic to the distinct arrangements of the elements $(m_1, \ldots, m_{i-1})$ and m^*. Among the sequences in S there must be a minimum of $|S|/(\ell+1-i)!$ such arrangements (since, given a fixed $(m_1, \ldots, m_{i-1}, m^*)$, there are $(\ell+1-i)!$ sequences possible), and sequences with each arrangement will block in slot i a total of $(\ell+1-i)!$ distinct sequences (i.e., any sequence beginning with $(m_1, \ldots, m_{i-1}, m^*)$). Hence, the sequences in S will block in slot i at least $(|S|/(\ell+1-i)!)(\ell+1-i)! = |S|$ distinct sequences. □

In total, we notice that at least $|S|$ distinct sequences are blocked in slot i for any $i \in [\ell]$, and so there are at least $|S|\ell$ distinct pairs (π, i) such that the sequence π is blocked in slot i by sequences in S. Furthermore, we recall that the sequences in S are each blocked in slot i by sequences in S for at most B different i, while the remaining $(\ell+1)! - |S|$ elements are each blocked in slot i by sequences in S for at most $B+1$ different i. This provides an upper bound of $B|S| + ((\ell+1)! - |S|)(B+1)$ on the number of "blocking" pairs (π, i). We lastly combine these lower and upper bounds (noting that, if the lower bound exceeded the upper bound, there would be a contradiction) to bound $|S|$:

$$|S|\ell \leq B|S| + ((\ell+1)! - |S|)(B+1) = B(\ell+1)! + (\ell+1)! - |S|$$
$$|S|(\ell+1) \leq (B+1)(\ell+1)!$$
$$|S| \leq (B+1)\ell!$$

□

Recall that, if S is the set of all bad sequences which are permutations of some set $\vec{m}^*$ of $\ell(n)+1$ distinct messages, we have by Lemma 2 that any sequence which is $(M(n)+r(n)+1)$-blocked by bad sequences in S must be good and thus lie outside of S. Hence, by Lemma 3, S has size at most $(M(n)+r(n)+1)\,(\ell(n))!$. Given any set of $\ell(n)+1$ distinct messages, then, the above applies to show that at most an

$$\frac{(M(n)+r(n)+1)\,(\ell(n))!}{(\ell(n)+1)!} = \frac{M(n)+r(n)+1}{\ell(n)+1}$$

fraction of the sequences defined by the permutations of this set can be bad, and the remainder must be good. Applying this to every possible set of $\ell(n)+1$ distinct messages, we get that at most the same fraction of *all* sequences of distinct messages can be bad. While the property that messages are distinct is not necessarily guaranteed, we note that the probability that they are not over uniformly randomly chosen messages is negligible—specifically, we notice that the probability of any pair of elements colliding is 2^{-n}, and so, by the union bound, the probability that any of the $\frac{\ell(n)(\ell(n)+1)}{2}$ pairs of elements can collide is smaller than $\nu(n) \triangleq \ell(n)^2 2^{-n}$ (which is negligible in n because $\ell(\cdot)$ is polynomial).

Hence, the chance that a sequence chosen at random is bad, which by definition is equal to the probability that a randomly chosen sequence of messages $\vec{m}^*{}_k = (m_{k,1}, \ldots, m_{k,\ell(n)}, m^*_k)$ will result in instance k returning Fail, can be at most the fraction of sequences without repeated elements which are bad plus the fraction of sequences with repeated elements, or $\frac{M(n)+r(n)+1}{\ell(n)+1} + \nu(n)$, as desired. □

Recall that, because this result holds for any execution of the experiment $\mathsf{Real}^*_{\{\vec{m}_I, m^*_I\}-k, \mathcal{O}_{ext}}(1^n)$, it must also hold over a *random* such execution—i.e., the actual execution $\mathsf{Real}(1^n)$ of $\mathcal{B}$, where the messages for all instances are chosen uniformly at random. Furthermore, it holds for any instance k of $\mathcal{A}$.

To conclude the proof of the proposition, by Claim 1 we know that $\mathcal{R}$ must send a total of at least $\ell(n)$ messages to each instance of $\mathcal{A}$ in order for the failure probability of $\mathcal{B}$ to emulate that instance to be more than negligible; if not, then $\mathcal{B}$ will always respond with $\perp$ (having not received $\ell(n)$ signature query responses), while $\mathcal{A}$ will return $\perp$ with all but negligible probability. Recall that $M(n)$ is an upper bound to the number of instances of $\mathcal{A}$ to which $\mathcal{R}$ sends $\ell(n)$ messages. By Claim 4 and a union bound over all $M(n)$ completed instances of $\mathcal{A}$, the failure probability of $\mathcal{B}$ for those instances is at most

$$M(n)\left(\frac{M(n)+r(n)+1}{\ell(n)+1} + \nu(n)\right)$$

for negligible $\nu(\cdot)$, and the failure probability for all other instances (of which there can only be a polynomial number by the time constraint on $\mathcal{R}$) is negligible by the union bound applied to Claim 1. Hence the overall failure probability of Real (i.e., the execution of $\mathcal{B}$) must be bounded above by

$$\Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Fail}] \leq M(n)\left(\frac{M(n)+r(n)+1}{\ell(n)+1} + \nu(n)\right) + \nu'(n)$$
$$< M(n)\left(\frac{M(n)+r(n)+1}{\ell(n)+1}\right) + \epsilon(n)$$

for some negligible functions $\nu'(\cdot)$ and $\epsilon(\cdot)$. □

Completing the Proof of Lemma 1. Finally, in order to bound the security loss, we note that, if the probability $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$ (which specifically is by definition a lower bound to the probability $\Pr[\mathsf{Output}[\mathsf{Ideal}(1^n)] = \mathsf{Accept}] - t(n)$; recall that $t(\cdot)$ is the threshold for the underlying assumption $\mathcal{C}$) is non-negligibly greater than the failure probability of Real, there exists a polynomial $p(\cdot)$ such that:

$$\Pr[\mathsf{Output}[\mathsf{Ideal}(1^n)] = \mathsf{Accept}] - t(n) \geq \Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Fail}] + \frac{1}{p(n)}$$

But, by Claim 3, this would imply that

$$\Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Accept}] - t(n) \geq \frac{1}{p(n)}$$

that is, that $\mathcal{B}$ breaks the security of $(\mathcal{C}, t(\cdot))$. So, by Proposition 1, unless $\mathcal{B}$ breaks the security of $(\mathcal{C}, t(\cdot))$, the above cannot be the case—that is, there must exist negligible $\epsilon(\cdot), \epsilon'(\cdot)$ such that, for sufficiently large n:

$$\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n) \leq \Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Fail}] + \epsilon'(n)$$
$$< M(n)\left(\frac{M(n)+r(n)+1}{\ell(n)+1}\right) + (\epsilon(n)+\epsilon'(n)) < M(n)\left(\frac{M(n)+r(n)+1}{\ell(n)}\right)$$

Of course, $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$, being a probability, is also trivially bounded above by 1. Furthermore, by the definition of $M(n)$, we know that $\mathsf{Query}_{\mathcal{R}^{\mathcal{A}}}(n) \geq M(n)\ell(n)$. Lastly, we consider two cases to derive our bound on the security loss.

Case 1. If $M(n) \geq \sqrt{\ell(n)} - (r(n)+1)$, then:

$$\lambda_{\mathcal{R}}(n) \geq \frac{\mathsf{Success}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)} \frac{\mathsf{Query}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Query}_{\mathcal{A}}(n)} \geq \frac{1}{1}\frac{M(n)\ell(n)}{\ell(n)}$$
$$= M(n) \geq \sqrt{\ell(n)} - (r(n)+1)$$

Case 2. Otherwise, if $M(n) < \sqrt{\ell(n)} - (r(n)+1)$ we have $M(n) + r(n) + 1 < \sqrt{\ell(n)}$, and so:

$$\lambda_{\mathcal{R}}(n) \geq \frac{\mathsf{Success}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)} \frac{\mathsf{Query}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Query}_{\mathcal{A}}(n)} \geq \frac{1}{M(n)\left(\frac{M(n)+r(n)+1}{\ell(n)}\right)} \frac{M(n)\ell(n)}{\ell(n)}$$
$$= \frac{\ell(n)}{M(n)+r(n)+1} > \frac{\ell(n)}{\sqrt{\ell(n)}} = \sqrt{\ell(n)}$$

Either way, we observe that $\lambda_{\mathcal{R}}(n) \geq \sqrt{\ell(n)} - (r(n)+1)$, thus completing the proof of both Lemma 1 and Theorem 2.

References

1. Abdalla, M., Fouque, P.-A., Lyubashevsky, V., Tibouchi, M.: Tightly-secure signatures from lossy identification schemes. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 572–590. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_34
2. Abe, M., Groth, J., Ohkubo, M.: Separating short structure-preserving signatures from non-interactive assumptions. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 628–646. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_34
3. Bader, C., Jager, T., Li, Y., Schäge, S.: On the impossibility of tight cryptographic reductions. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 273–304. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_10
4. Baecher, P., Brzuska, C., Fischlin, M.: Notions of black-box reductions, revisited. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part I. LNCS, vol. 8269, pp. 296–315. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42033-7_16
5. Bellare, M., Rogaway, P.: The exact security of digital signatures-how to sign with RSA and Rabin. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_34
6. Bernhard, D., Fischlin, M., Warinschi, B.: On the hardness of proving CCA-security of signed ElGamal. In: Cheng, C.-M., Chung, K.-M., Persiano, G., Yang, B.-Y. (eds.) PKC 2016, Part I. LNCS, vol. 9614, pp. 47–69. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49384-7_3
7. Bernstein, D.J.: Proving tight security for Rabin-Williams signatures. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 70–87. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_5
8. Boneh, D., Venkatesan, R.: Breaking RSA may not be equivalent to factoring. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 59–71. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054117
9. Brakerski, Z., Goldwasser, S., Rothblum, G.N., Vaikuntanathan, V.: Weak verifiable random functions. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 558–576. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_33
10. Bresson, E., Monnerat, J., Vergnaud, D.: Separation results on the "one-more" computational problems. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 71–87. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-79263-5_5
11. Canetti, R., Lin, H., Pass, R.: Adaptive hardness and composable security in the plain model from standard assumptions. In: 2010 IEEE 51st Annual Symposium on Foundations of Computer Science, pp. 541–550, October 2010
12. Chen, J., Gong, J., Weng, J.: Tightly secure IBE under constant-size master public key. In: Fehr, S. (ed.) PKC 2017. LNCS, vol. 10174, pp. 207–231. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54365-8_9
13. Chen, J., Wee, H.: Fully, (almost) tightly secure IBE and dual system groups. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 435–460. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_25
14. Coron, J.-S.: Optimal security proofs for PSS and other signature schemes. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 272–287. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_18

15. Deng, Y., Goyal, V., Sahai, A.: Resolving the simultaneous resettability conjecture and a new non-black-box simulation strategy. In: Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, pp. 251–260. IEEE Computer Society, Washington, DC (2009)
16. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theor. **22**(6), 644–654 (2006)
17. Dwork, C., Naor, M., Sahai, A.: Concurrent zero-knowledge. In: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, STOC 1998, pp. 409–418. ACM, New York (1998)
18. Fischlin, M., Schröder, D.: On the impossibility of three-move blind signature schemes. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 197–215. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_10
19. Gilad, Y., Hemo, R., Micali, S., Vlachos, G., Zeldovich, N.: Algorand: scaling byzantine agreements for cryptocurrencies. In: Proceedings of the 26th Symposium on Operating Systems Principles, SOSP 2017, pp. 51–68. ACM, New York (2017)
20. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems. In: Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, STOC 1985, pp. 291–304. ACM, New York (1985)
21. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. **17**(2), 281–308 (1988)
22. Haitner, I., Rosen, A., Shaltiel, R.: On the (im)possibility of Arthur-Merlin witness hiding protocols. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 220–237. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_14
23. Hofheinz, D., Jager, T.: Tightly secure signatures and public-key encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 590–607. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_35
24. Hofheinz, D., Jager, T., Knapp, E.: Waters signatures with optimal security reduction. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 66–83. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_5
25. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, STOC 1989, pp. 44–61. ACM, New York (1989)
26. Jager, T.: Verifiable random functions from weaker assumptions. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 121–143. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_5
27. Kakvi, S.A., Kiltz, E.: Optimal security proofs for full domain hash, revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 537–553. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_32
28. Lindell, Y.: Bounded-concurrent secure two-party computation without setup assumptions. In: Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing, STOC 2003, pp. 683–692. ACM, New York (2003)
29. Luby, M.: Pseudorandomness and Cryptographic Applications. Princeton University Press, Princeton (1996)
30. Lysyanskaya, A.: Unique signatures and verifiable random functions from the DH-DDH separation. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 597–612. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_38
31. Micali, S., Vadhan, S., Rabin, M.: Verifiable random functions. In: Proceedings of the 40th Annual Symposium on Foundations of Computer Science, FOCS 1999, p. 120–130. IEEE Computer Society, Washington, DC (1999)

32. Naor, M.: On cryptographic assumptions and challenges. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 96–109. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_6
33. Pass, R.: Limits of provable security from standard assumptions. In: Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing, STOC 2011, pp. 109–118. ACM, New York (2011)
34. Pass, R., Shi, E.: The sleepy model of consensus. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10625, pp. 380–409. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70697-9_14
35. Pass, R., Tseng, W.-L., Venkitasubramaniam, M.: Concurrent zero knowledge, revisited. J. Cryptol. **27**(1), 45–66 (2014)
36. Pass, R., Venkitasubramaniam, M.: On constant-round concurrent zero-knowledge. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 553–570. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78524-8_30
37. Prabhakaran, M., Rosen, A., Sahai, A.: Concurrent zero knowledge with logarithmic round-complexity. In: Proceedings of the 43rd Symposium on Foundations of Computer Science, FOCS 2002, pp. 366–375. IEEE Computer Society, Washington, DC (2002)
38. Rabin, M.: Digitalized signatures and public-key functions as intractable as factorization. Technical report, Cambridge, MA, USA (1979)
39. Richardson, R., Kilian, J.: On the concurrent composition of zero-knowledge proofs. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 415–431. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_29
40. Rivest, R., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2), 120–126 (1978)
41. Shamir, A.: A fast signature scheme. Technical report, Cambridge, MA, USA (1978)
42. Wang, Y., Matsuda, T., Hanaoka, G., Tanaka, K.: Memory lower bounds of reductions revisited. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 61–90. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_3