On the Adaptive Security of MACs and PRFs[*][†]

Andrew Morgan
Cornell University
asmorgan@cs.cornell.edu

Rafael Pass
Cornell Tech
rafael@cs.cornell.edu

Elaine Shi
Cornell University
runting@gmail.com

September 10, 2020

Abstract

We consider the security of two of the most commonly used cryptographic primitives—message authentication codes (MACs) and pseudorandom functions (PRFs)—in a *multi-user setting with adaptive corruption.* Whereas is it well known that any secure MAC or PRF is also multi-user secure under adaptive corruption, the trivial reduction induces a security loss that is linear in the number of users.

Our main result shows that black-box reductions from "standard" assumptions cannot be used to provide a tight, or even a linear-preserving, security reduction for adaptive multi-user secure deterministic stateless MACs and thus also PRFs. In other words, a security loss that grows with the number of users is necessary for any such black-box reduction.

1 Introduction

Message authentication codes (MACs) are one of the most fundamental cryptographic primitives. MACs are secret-key primitives that enable a party to produce a "tag" for messages in such a way that, while anyone possessing the secret key can verify the validity of the tag, an adversary without access to the key is unable to forge a correct tag for a message. This allows the participating parties to use the tags to confirm that a tagged message is *authentic*—that is, that it originated from a trusted sender and was delivered without modification. A *pseudorandom function* (PRF) is a related primitive which can easily be used to construct a MAC; in addition to being unforgeable by an adversary, the output ("tag") from a PRF is also pseudorandom (i.e., indistinguishable from true randomness to the adversary).

Multi-user Security and Adaptive Corruptions MACs and PRFs are also some of the most commonly used cryptographic primitives in practice; as such, they are often deployed in contexts

[*]A preliminary version of this paper will appear in the proceedings of ASIACRYPT 2020; this is the full version.

[†]Supported in part by NSF Award SATC-1704788, NSF Award RI-1703846, NSF Award CNS-1561209, AFOSR Award FA9550-18-1-0267, DARPA SIEVE award HR00110C0086, and a JP Morgan Faculty Award. This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via 2019-19-020700006. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

with huge numbers of users. For instance, MACs are used in protocols for secure key exchange (as first formalized in [DH76]), including the well-known and widely employed TLS protocol [DA99, DR06, DR08], which is used today by major websites with billions of daily active users. A natural question, then, is to what extent the *multi-user setting* in which MACs or PRFs are practically employed affects the security of these primitives. In particular, in a multi-user setting it is natural to consider an *adaptive* adversary who may decide to corrupt a subset of the users (and as a result of the corruption receive their secret keys); given such an adversary, we would like to guarantee that uncorrupted users' instances remain secure. Indeed, various forms of multi-user security have been considered since the work of Bellare et al. [BCK96] (see also e.g., [BBM00, BPS00, BRT12, ML15, Tes15]). In recent work, Bader et al. [BHJ$^+$15] explicitly consider a notion of adaptive multi-user security for signature schemes and MACs. They remark that a simple "guessing" reduction, originally proposed in [BCK96] for multi-user security of PRFs without corruption, shows that any single-user secure MAC is adaptively multi-user secure. Specifically, given a multi-user adversary that runs, say, ℓ instances of a MAC, one can construct a single-user adversary that, given an instance of the MAC, simulates the game for the multi-user adversary by embedding its own instance into a random one of the multi-user instances and generating $\ell - 1$ keys to simulate the rest of the instances (including returning the respective keys for corruption queries). If the multi-user adversary picks the correct instance to break by forging a tag, then the single-user adversary can use the forgery it returns to win its own game.

Security loss and linear-preserving reductions. The above argument shows that any "single-user" secure MAC also is multi-user secure under adaptive corruption; a similar argument holds also for PRFs [ACD$^+$19]. However, security is only "polynomially" preserved; in a concrete sense, the reduction incurs a significant *security loss* [Lub96], as one might note that the single-user adversary we describe is far less efficient than the multi-user adversary on which it is based. In particular, in a setting where we have a large number ℓ of instances available to the adversary, the single-user adversary's probability of success is indeed reduced by a proportionate factor of ℓ. As discussed in works such as [ML15, Tes15], this has considerable implications on the concrete security of such a primitive in a setting where a large number of instances might be in use at once. More formally, the *security loss* is defined as the "work", or *expected running time*, required by the reduction to break the underlying assumption (in the above example, single-user security) using a particular adversary against a primitive (adaptive multi-user security) as an oracle, divided by the work required by the adversary to break the primitive. Intuitively, the "best possible" type of reduction is one with a constant security loss, or a *tight* [Lub96] reduction, which guarantees that the primitive will inherit roughly the same level of concrete security as the underlying assumption. A reduction with a security loss equal to a fixed polynomial $p(n)$ in the security parameter, also known as a *linear-preserving* reduction, is still intuitively desirable. The "guessing" reduction above, however, has a security loss of ℓ, or the number of instances in the multi-user security game, and so it is neither tight nor linear-preserving. A natural question, then, is whether we can do better than this trivial reduction and construct a provably secure MAC with a linear-preserving reduction.

In fact, the work of [BHJ$^+$15] shows how to overcome the security loss of this "trivial" guessing reduction: as a key building block towards an "almost-tightly secure" authenticated key exchange protocol, the authors present an elegant construction of an adaptively multi-user secure *digital signature scheme* with a linear-preserving reduction. In particular, the security loss of their constructions is linear in the security parameter n, and *independent* of the number of users!

On the importance of determinstic tagging. However, the signature construction given in [BHJ$^+$15] requires introducing *randomness* into the signing algorithm. While this scheme can indeed be interpreted as a MAC, the fact that the signing algorithm is randomized means that the resulting MAC also becomes *randomized.* While some theoretical textbooks (see e.g., [Gol09]) allow the tagging mechanism in the definition of a MAC to be randomized, practical texts (e.g., the Handbook of Applied Cryptography [MVO96]), as well as NIST standardizations [Bar16], require the tagging algorithm to be *deterministic.* As far as we know, all constructions used in practice, as well as all standardized constructions of MACs, are deterministic; indeed, there are several good reasons for sticking to deterministic constructions. First, reliable randomness is hard to generate, and thus randomized constructions are avoided in practice for time-critical primitives that are used repeatedly and on a large scale, as is the case for MACs. Furthermore, any PRF, when viewed as a MAC, is by definition deterministic, and additionally is internally *stateless*; in fact, we remark that almost all practical MAC constructions are also stateless, a notable exception being GMAC [Dwo07].[1] Obtaining a tightly secure PRF in a multi-user setting requires, at a minimum, a tightly secure deterministic and stateless MAC.

As such, the current state of affairs leaves open the important problem of determining the concrete multi-user security of the MACs and PRFs used in practice today. In particular, focusing on the case of stateless MACs, we consider the question of whether either deterministic MACs or PRFs can, in an adaptive multi-user setting, have a security loss that is independent of the number of users:

> *Can there exist tight or linear-preserving reductions for proving the adaptive multi-user security of any deterministic (and stateless) MAC or any PRF based on some "standard assumption"?*

At first glance, it may seem that answering this question is trivial, since any randomized MAC can be made deterministic. Indeed, as shown in [Gol09] for signature schemes, one may simply fix the randomness to be the result of applying a PRF to the input message. This construction, however, only preserves the tightness of the reduction if the underlying PRF itself is tightly secure in the adaptive multi-user setting—however, since any such PRF is already trivially a (deterministic) MAC, we end up precisely where we started.

1.1 Our Results

Our main result, in fact, provides a strong *negative* answer to the above question. We demonstrate that there exists no linear-preserving (or, hence, tight) black-box reduction for basing adaptive multi-user security of any deterministic MAC, and thus also any PRF, on any secure "standard" assumption. By a "standard" assumption, we here refer to any assumption that can be modeled as a game, or an interaction between a challenger $\mathcal{C}$ and a polynomial-time adversary $\mathcal{A}$, that proceeds in an *a priori* bounded number of rounds—following [Pas11], we refer to this class of assumptions as *bounded-round assumptions.*

Theorem 1 (Informal)**.** If there exists a linear-preserving black-box reduction $\mathcal{R}$ for basing adaptive multi-user security of a deterministic MAC on some bounded-round assumption $\mathcal{C}$, then $\mathcal{C}$ can be broken in polynomial time.

[1] GMAC is deterministic but stateful; it keeps an internal counter to use as an additional non-reusable input, or *nonce.* Stateful MACs such as GMAC are not subject to the results we prove here.

In particular, we show that any such black-box reduction (to a secure bounded-round assumption) requires a security loss of $\Omega(\sqrt{\ell})$, where ℓ is the number of users.

We remark that since any PRF or deterministic digital signature scheme trivially implies a deterministic MAC (via a tight security reduction), our theorem also directly rules out linear-preserving black-box reductions for basing adaptive multi-user security of PRFs or deterministic signatures on standard assumptions.

Related results. A few prior works have in fact dealt with this question for other types of primitives. Non-adaptive multi-user security was originally introduced by Bellare, Canetti, and Krawczyk [BCK96] for pseudorandom function families; the authors also introduced the original version of the classical "guessing" reduction from multi-user to single-user security in that context. As mentioned above, [BHJ+15] introduced adaptive multi-user security in the context of signatures and MACs (and presented applications for secure key exchange), and [ACD+19] considered it in the context of PRFs. Recently, there has been a wealth of positive results demonstrating the achievability of tight reductions from multi-user to single-user security of authenticated encryption protocols and block cipher-based schemes (see, e.g., [BHJ+15,HT16,LMP17,BHT18,HTT18]); some of these results, as we have noted, consider the case of *randomized* or *stateful* (nonce-based) MACs such as GMAC, which are not subject to our security bound.

Concerning negative results, several prior works have ruled out certain *restricted* classes of linear-preserving reductions from multi-user security of various primitives. Bellare et al. [BBM00] first introduced the (non-adaptive) notion of multi-user security for public-key encryption and demonstrated that there does not exist an efficient *generic* reduction from multi-user to single-user security which works for *every* encryption scheme. But one may still hope to circumvent this by constructing a specific encryption scheme for which such a reduction exists, or by directly basing multi-user security on some other (standard) assumption; indeed, [BBM00] does demonstrate certain schemes for which security loss can be avoided. A later work by Jager et al. [JSSOW17] proves a negative result for *authenticated encryption*, showing that certain *restricted* types of black-box reductions—in particular, "straight-line" (i.e., non-rewinding) reductions—from *adaptive* multi-user security to single-user security of *any* authenticated encryption scheme possessing a strong "key uniqueness" property (i.e., that any two keys which produce the same ciphertexts for some polynomial number of inputs must agree on *all* inputs) must inherit a similarly large security loss.

Most relevantly to our work, Chatterjee et al. [CMS12] show a negative result for the case of *generic* reductions from adaptive multi-user to single-user security of MACs. Specifically, the authors propose a "collision-finding" attack on multi-user MAC security whose success probability increases by a factor of roughly ℓ (the number of instances) in a multi-user setting as compared to its single-user analogue against an *idealized* MAC. Similarly to [BBM00], this elegantly demonstrates that a security loss is inherent in *generic* reductions from multi-user to single-user security; however, their results still leave open the question of whether the same holds true for a reduction to a specific MAC (where, as [BBM00] shows for public-key encryption, there may be more effective single-user attacks), let alone whether it holds for directly reducing multi-user security to an underlying assumption without relying on single-user security.

In contrast to the above results, the bound we show here applies to *any* (i.e., not a restricted class of) black-box reduction and to any "standard" (bounded-round) assumption; additionally, it applies to any construction of the primitives we consider (i.e., deterministic MACs and PRFs).

Our work builds on a line of research on using "meta-reductions" [BV98] to prove impossibility results for black-box reductions, and in particular to study the inherent security loss of (single-user) secure digital signatures. Most recently, expanding upon earlier results [Cor02, KK12, HJK12, BJLS16] which dealt with restricted reductions, [MP18] provides a security loss bound ruling out linear-preserving reductions for single-user security of a primitive called *unique signature schemes*. While we rely on a significant amount of insight from these prior results (and in particular from [MP18]), adapting their techniques to our setting is quite non-trivial (as we shall explain below). Indeed, as far as we are aware, all known black-box separations using the meta-reduction paradigm only apply to primitives that embody some form of uniqueness or rerandomizability (which in turn can be viewed as a "distributional uniqueness") property—we will return to what this uniqueness property means shortly (and how it is used). In contrast, our impossibility result does not (explicitly) refer to or require a primitive that embodies such a property.

Summarizing the above discussion, as far as we know, our results not only constitute the first "complete" black-box lower bound (in the sense that we consider "unrestricted" reductions) on the security loss of any primitive in the multi-user setting, but also address the security of two of the most fundamental primitives—MACs and PRFs—used practically in a multi-user setting. Additionally, we present the first usage of the meta-reduction paradigm to rule out reductions from a primitive that does not itself embody a uniqueness (or rerandomizability) property.

1.2 Overview

The meta-reduction paradigm. We prove our security loss bound using an adaptation of the "meta-reduction" paradigm, originally devised in [BV98] (see also [BMV08, HRS09, FS10, Pas11, GW11, AGO11, BBF13, BFW16] for related work concerning meta-reductions). The paradigm was originally used to show black-box *impossibility* results, but Coron in [Cor02] pioneered the usage of meta-reductions to instead show *lower bounds on security loss*; this line of work was continued in [KK12, BJLS16, MP18]. Meta-reductions were first used in relation to multi-user security in [JSSOW17], which dealt with multi-user to single-user reductions for authenticated encryption (satisfying a key-uniqueness property).

At a high level, the meta-reduction paradigm proves the impossibility of any black-box reduction from a primitive Π to a *secure* assumption $\mathcal{C}$.[2] To illustrate this approach for the case of an impossibility result, consider attempting to prove the impossibility of such a reduction $\mathcal{R}$ that breaks the assumption $\mathcal{C}$ by using black-box access to some "ideal adversary" $\mathcal{A}$ (which in turn breaks security of the constructed primitive). By definition, if $\mathcal{A}$ breaks the primitive with probability 1, then $\mathcal{R}^{\mathcal{A}}$ should break $\mathcal{C}$ with non-negligible probability, even if we construct $\mathcal{A}$ to be inefficient (e.g., win by brute force).

It remains then to show that, if such an $\mathcal{R}$ exists, then $\mathcal{C}$ can be broken *efficiently*, contradicting the assumption of $\mathcal{C}$'s security. While $\mathcal{R}^{\mathcal{A}}$ itself clearly will not break $\mathcal{C}$ *efficiently* if $\mathcal{A}$ uses brute force, one can instead create an efficient *meta-reduction* $\mathcal{B}$ that efficiently "emulates" $\mathcal{A}$ while running $\mathcal{R}$. If one can show that the meta-reduction $\mathcal{B}$ always succeeds in emulating the real interaction $\mathcal{R}^{\mathcal{A}}$, then the meta-reduction breaks $\mathcal{C}$ with non-negligible probability.

On the other hand, it might be impossible to create a meta-reduction that emulates $\mathcal{R}^{\mathcal{A}}$ perfectly; instead, it might be the case that we can construct $\mathcal{B}$ that emulates $\mathcal{R}^{\mathcal{A}}$ with probability at least $1 - p(n)$ for some inverse polynomial $p(\cdot)$. In this case, if $\mathcal{R}^{\mathcal{A}}$ breaks $\mathcal{C}$ with probability

[2]Consider $\mathcal{C}$ to be the "challenger" for the security game; an efficient adversary "breaks" $\mathcal{C}$ by forcing it to output Accept with probability non-negligibly better than a certain threshold t.

non-negligibly greater than $p(n)$, then $\mathcal{B}$, being identically distributed to $\mathcal{R}^{\mathcal{A}}$ except with probability $p(n)$, will in fact still break $\mathcal{C}$ with non-negligible probability, thus ruling out any such $\mathcal{R}$. By bounding $\mathcal{R}$'s success probability in terms of its running time, this observation can be used to derive a *security loss bound* for any reduction $\mathcal{R}$ in cases where such reductions may not be fully impossible.

Rewinding techniques. Of course, a useful meta-reduction requires two important constructions: (1) the ideal and inefficient adversary $\mathcal{A}$, and (2) the meta-reduction $\mathcal{B}$. Most importantly, while it would be simple to construct an adversary $\mathcal{A}$ that breaks $\mathcal{C}$ by brute force, $\mathcal{B}$ must also be able to gain enough information by simulating and receiving responses to $\mathcal{A}$'s messages in order to determine, with high probability, the secret information necessary to break $\mathcal{C}$ without brute force.

Coron's original meta-reduction presents an effective way of accomplishing this in the setting where $\mathcal{A}$ breaks the unforgeability of unique signatures, or, more generally, any "one-more" style security game where an adversary, after making some number of queries, must then guess the result of querying a new input. Specifically, if we assume $\mathcal{A}$ makes a significant number of queries $\ell(n)$ with inputs $x_1, \ldots, x_{\ell(n)}$ before brute-forcing its guess (and, importantly, will return $\perp$ instead if the answers to its queries are incorrect), $\mathcal{B}$ can make the same set of queries and, rather than brute-forcing a guess, may instead pick the new input x^* and *rewind* the reduction $\mathcal{R}$ up to $\ell(n)$ different times[3], each time replacing a different one of the messages with x^* in the hopes that $\mathcal{R}$ will provide a valid response that $\mathcal{B}$ can use in the main execution.

This rewinding technique in fact can be shown to emulate $\mathcal{A}$ except with probability $O(1/\ell(n))$; intuitively, this is because, if $\mathcal{B}$ is *unable* to extract a correct response in some rewinding, that rewinding corresponds to a sequence of randomness where, if it occurs in the non-rewound execution, $\mathcal{A}$ receives an incorrect response to one of its queries and hence does not need to return a forgery. Hence, at a very high level, for each sequence of messages $x_1, \ldots, x_{\ell(n)}$ where $\mathcal{B}$ fails to extract a forgery, $\mathcal{B}$ must receive an incorrect response to x^* in each of the $\ell(n)$ rewindings, and so there are $\ell(n)$ sequences where $\mathcal{B}$ can successfully emulate $\mathcal{A}$ (as both can return $\perp$).

It is important to note where *uniqueness* of the signature scheme comes in: to ensure that $\mathcal{B}$ is correctly simulating the *distribution* of $\mathcal{A}$'s messages, we need to make sure that the forgery extracted by $\mathcal{B}$ from $\mathcal{R}$ is the same as the forgery that $\mathcal{A}$ would have generated. In the case of a unique signature, we know there can only be a single valid forgery; as such, $\mathcal{B}$ indeed generates the right distribution if it manages to extract a forgery from $\mathcal{R}$.

The case of adaptive multi-user unforgeability. Coron's meta-reduction was tailored to the specific case of unique signatures; however, in our case, adaptive multi-user unforgeability—that is, the security of a MAC—can also be thought of as a type of "one-more" assumption. Specifically, an adversary against ℓ instances of a MAC can make $\ell - 1$ key-opening queries and subsequently guess the last key in order to break the security of the respective unopened instance (i.e., by guessing the MAC on an unqueried input); a natural approach to creating a meta-reduction for this case, then, would be to have $\mathcal{B}$ rewind these key-opening queries and try opening the final instance in the rewindings, similar to Coron's treatment of queries for unique signatures. However, there are several complications with this approach that, for various reasons, did not need to be considered

[3]In fact, in Coron's theorem, it was sufficient to pick a random *one* of these rewindings, but this is not sufficient for our result.

in [Cor02]; we next present a high-level overview of these issues and how we approach them in this work.

"Effective" key uniqueness. First, recall that $\mathcal{R}$ does not necessarily need to act as an honest challenger, and so $\mathcal{B}$ must have a way to verify that $\mathcal{R}$'s responses to its queries (in this case, key-opening queries) are correct. As mentioned above, this is why Coron's results (and those following) only applied to *unique* signatures, and why, for the case of adaptive multi-user security, [JSSOW17] considered only schemes with a key uniqueness property.

We do not want to require any sort of inherent "key uniqueness" for the class of MACs we rule out; hence, we instead move to considering a more elaborate "ideal" adversary $\mathcal{A}$. In particular, we let $\mathcal{A}$ first make a large number of random tag queries to each instance of the MAC; then, upon receiving a response to a key-opening query, $\mathcal{A}$ will verify that all of the responses to the tag queries are consistent with the returned key. Towards analyzing this technique, we present an information-theoretic lemma showing that if the number of queries $q(n)$ is sufficiently larger than the length of the key n, then, with high probability, any pair of keys that are consistent with one another on the $q(n)$ tag queries is such that the keys will also agree on another random input (i.e., the input for which we produce the forgery to break security of the MAC).

In essence, then, our approach makes keys "effectively" unique in the sense that, with high probability, they operate indistinguishably on random inputs with respect to our particular ideal adversary $\mathcal{A}$. As far as we know, this stands in contrast to all prior impossibility results following the meta-reduction paradigm, which explicitly worked only with primitives where the adversary's responses to the queries to be rewound are unique or "distributionally unique" (i.e., rerandomizable).

Reductions with concurrency and rewinding. Furthermore, Coron's result in [Cor02], as well as many subsequent security loss bounds proven using meta-reductions (e.g., [KK12, BJLS16, JSSOW17]) only apply to *restricted* reductions that are *"straight-line"* in the sense that $\mathcal{R}$ will never attempt to rewind $\mathcal{A}$ and $\mathcal{R}$ will always finish executing a single instance of $\mathcal{A}$ before starting another one. In general, reductions may run multiple instances of the adversary concurrently, which can be highly problematic for rewinding-based meta-reductions, as $\mathcal{B}$ may have to rewind a "nested" instance of its adversary to produce a correctly-distributed output while already in the middle of rewinding another instance. If many instances need to be rewound concurrently, the running time of $\mathcal{B}$ can potentially be super-polynomial, which fails to uphold the requirement that $\mathcal{B}$ break $\mathcal{C}$ efficiently.

Luckily, some recent works have presented meta-reductions that deal with concurrent interactions, primarily by using techniques from concurrent zero-knowledge (see [Pas11, MP18]). We build on the technique established in the generalization of Coron's bound given in [MP18], which shows that $\mathcal{B}$ can safely ignore any rewindings which would require any sort of nested rewinding. At a high level, if $\mathcal{R}$ runs few instances of $\mathcal{A}$, then other instances rarely interfere with rewinding during $\mathcal{B}$, resulting in virtually no change to the failure probability; on the other hand, if $\mathcal{R}$ runs many instances, then the time taken by $\mathcal{R}$ compared to $\mathcal{A}$ will be the dominant factor in the security loss, so the increase in failure probability caused by potentially having many ignored rewindings has very limited relevance in the analysis.

This approach nonetheless requires non-trivial modification to work in our case, due to the additional caveat that $\mathcal{R}$ may attempt to rewind instances of $\mathcal{A}$. While [MP18] relied on a "rewinding-proof" construction of $\mathcal{A}$ and $\mathcal{B}$ where the randomness was determined at the start, so that the uniqueness property would guarantee only a single possible accepting transcript (thus making rewinding pointless), recall that we no longer have a guaranteed uniqueness property, but instead one that holds "most of the time". Furthermore, we can no longer construct $\mathcal{A}$ to be fully resilient to rewinding, due to the additional complexity of having both tag queries and key-opening queries; instead, we construct $\mathcal{A}$ to be resilient to *most* rewinding—particularly, all rewinding except from the key-opening query phase to the tag query phase—and prove our bound in terms of how often "meaningful" rewinding (i.e., rewinding that does affect the result) can occur in addition to the number of instances of $\mathcal{A}$.

This requires some additional care, however: while $\mathcal{A}$ can easily be made rewinding-proof (with the exception of the "meaningful" rewinding), we in fact can only show that $\mathcal{B}$ is resilient to rewinding as long as key uniqueness holds; otherwise, while $\mathcal{A}$ can always pick a determinstic one of the brute-forced keys for a forgery, $\mathcal{B}$ cannot necessarily do this efficiently just from the responses to rewound queries, and so $\mathcal{R}$ could theoretically rewind $\mathcal{B}$ to try and get multiple different forgeries to correspond to multiple different keys. We thus require a hybrid argument with an unconditionally rewinding-proof but inefficient hybrid $\mathcal{B}'$ (which acts identically to $\mathcal{B}$ when uniqueness holds and to $\mathcal{A}$ when it does not) for the majority of our analysis, subsequently showing that $\mathcal{B}'$ is identically distributed to $\mathcal{B}$ except in the rare case when uniqueness fails.

Interactive assumptions. Lastly, many of the preceding works were restricted to ruling out reductions to *non-interactive*, or two-round, assumptions, since $\mathcal{B}$ rewinding the reduction $\mathcal{R}$ might require additional, or different, queries to be made to the challenger $\mathcal{C}$ for the underlying assumption, which cannot be rewound and whose output may be dependent on the number, order, or content of queries made. However, as demonstrated in earlier rewinding-based meta-reductions such as [Pas11, MP18], we may once again safely ignore rewindings that contain such external communication as long as the number of rounds of external communication is bounded by some polynomial $r(\cdot)$ in the security parameter—that is, as long as the underlying assumption is a *bounded-round* assumption.

1.3 Outline

We present the definitions necessary for our main theorem in Section 2. Our main theorem statement and a technical overview are given in Section 3, and the detailed proof is given in Section 4. Lastly, Appendices A and B contain a discussion of the adaptation of our proof to the case of deterministic digital signatures and pseudorandom functions, respectively.

2 Preliminaries and Definitions

We note that the definitions we provide in Sections 2.3 through 2.5 are adapted from [MP18].

2.1 Preliminaries

Let $\mathbb{N} = \{1, 2, 3, \ldots\}$ and $[n] = \{m \in \mathbb{N} \mid m \leq n\} = \{1, 2, \ldots, n\}$. Take 1^n (given as an input to certain algorithms to represent the security parameter) to be the string of n ones. Let $\mathbf{1}_P$ be the *indicator function* of some equality or inequality statement P, defined as being 1 when P is true and 0 otherwise.

We say a statement is true for "sufficiently large $n \in \mathbb{N}$" if there exists $N \in \mathbb{N}$ such that the statement holds for all $n \geq N$. We shall call a function $\epsilon(\cdot)$ *negligible* if, for any polynomial $p(\cdot)$, $\epsilon(n) < 1/p(n)$ for sufficiently large $n \in N$; conversely, a function is *non-negligible* or *polynomial* if the above is not true.

Interactive Algorithms. We model interaction between algorithms by Turing machine interaction as given in [GMR85]. For oracle-aided algorithms $\mathcal{R}^{(\cdot)}$ with deterministic oracles, we assume that $\mathcal{R}$'s oracle will take as input the current partial transcript of the interaction and return the result of applying the oracle's respective "next-message" function to the transcript. Using a stateless oracle in such a way allows us to consider the case where $\mathcal{R}$ can "rewind" its oracle to a previous point in the interaction by providing an earlier transcript. In fact, we note that even stateful oracles can be modeled in this way (by "replaying" or reading from the transcript to simulate the state of the oracle), as can non-deterministic oracles (by sampling a deterministic oracle randomly from a family of oracles with different fixed randomness); we take advantage of both of these observations in the proof of our main theorem.

Given oracle-aided algorithm $\mathcal{R}$ and interactive algorithms $\mathcal{A}$ and $\mathcal{C}$, let $\mathcal{R}^{\mathcal{A}}(x)$ denote the distribution over the output of $\mathcal{R}$ when interacting with oracle $\mathcal{A}$ on common input x, $\langle \mathcal{A}, \mathcal{C} \rangle(x)$ denote the distribution over the output of $\mathcal{C}$ after interaction between $\mathcal{A}$ and $\mathcal{C}$ on common input x (also provided to oracles if $\mathcal{A}$ or $\mathcal{C}$ are oracle-aided), and $[\mathcal{A} \leftrightarrow \mathcal{C}](x)$ denote the complete *view* (all messages sent, random coins used, and outputs) of the interaction between $\mathcal{A}$ and $\mathcal{C}$ on common input x.

If we wish to describe an interaction with certain randomness fixed, we write $\langle \mathcal{A}, \mathcal{C} \rangle_{y \leftarrow a}(x)$ (resp. $[\mathcal{A} \leftrightarrow \mathcal{C}]_{y \leftarrow a}(x)$) to denote that variable y is fixed to value a, or $\langle \mathcal{A}, \mathcal{C} \rangle_{y}(x)$ to simply indicate that y is fixed when the value is clear from context.

2.2 Multi-User Secure MACs under Adaptive Corruption

First, we define the notion of a *message authentication code.*

Definition 1. We refer to a tuple of efficient (poly(n)-time) algorithms $\Pi = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$, where:

- $\mathsf{Gen}(1^n) \to k$ takes as input a security parameter n and outputs a secret key $k \in \{0,1\}^n$,
- $\mathsf{Tag}_k(m) \to \sigma$ takes as input a secret key k and a message m from some message space $\mathcal{M}_n$ of size super-polynomial in n, and outputs a tag σ for the message, and
- $\mathsf{Ver}_k(m, \sigma) \to \{\mathsf{Accept}, \mathsf{Reject}\}$ takes as input a secret key k, a message m, and a tag σ, and outputs Accept or Reject denoting whether the tag σ is valid for the message m, specifically in such a manner that $\Pr[k \leftarrow \mathsf{Gen}(1^n); \mathsf{Ver}_k(m, \mathsf{Tag}_k(m)) \to \mathsf{Accept}] = 1$ for any valid message $m \in \mathcal{M}_n$,

as a **message authentication code** (MAC). If, in addition, the following properties hold:

- $\mathsf{Tag}_k(m)$ is a deterministic function, and
- $\mathsf{Ver}_k(m, \sigma) \to \mathsf{Accept}$ if and only if $\mathsf{Tag}_k(m) = \sigma$,

then we refer to Π as a **deterministic MAC**.

Note that we focus here on MACs having both an input (message) and output (tag) space superpolynomial in the length of a key (the security parameter n), a property which is satisfied by virtually all standard definitions and constructions.

The traditional notion of security for a MAC states that, given some instance of a MAC (i.e., a secret key $k \leftarrow \mathsf{Gen}(1^n)$), an efficient adversary given an oracle for the Tag algorithm is unable to forge a valid tag for a new message (i.e., return a pair (m, σ) where $\mathsf{Ver}_k(m, \sigma) \to \mathsf{Accept}$) without having queried a tag for that message using the oracle. Our definition of multi-user security with adaptive corruption expands this to a polynomial number $\ell(n)$ of instances of the MAC, and allows the adversary to make *key-opening* queries (i.e., to "corrupt" an instance and recover its key) in addition to tag queries; the adversary wins if they produce a valid forgery (m, σ) for some instance without having either queried the tag for m on that instance or corrupted the instance itself. Formally:

Definition 2. A MAC $\Pi = (\mathsf{Gen}, \mathsf{Tag}, \mathsf{Ver})$ is an $\ell(n)$**-key unforgeable MAC under adaptive corruption** (or **adaptively** $\ell(n)$**-key unforgeable**) if, for any interactive oracle-aided non-uniform probabilistic polynomial-time algorithm $\mathcal{A}$, there is a negligible function $\epsilon(\cdot)$ such that, for all $n \in \mathbb{N}$,

$$\Pr\left[\langle \mathcal{A}, \mathcal{C}_\Pi^{\ell(n)} \rangle(1^n) = \mathsf{Accept}\right] \leq \epsilon(n)$$

where $\mathcal{C}_\Pi^{\ell(n)}$ is the interactive challenger that does as follows on input 1^n:

- Let $(k_1, \ldots, k_{\ell(n)}) \leftarrow \mathsf{Gen}(1^n)$. Initialize empty transcript τ.
- Upon receiving a *tag* query (Query, i, m) for $i \in [\ell(n)]$, append $((\mathsf{Query}, i, m), \mathsf{Tag}_{k_i}(m))$ to τ and send τ.
- Upon receiving a *key-opening* query (Open, i) for $i \in [\ell(n)]$, append the tuple $((\mathsf{Open}, i), k_i)$ to τ and send τ.
- Upon receiving a forgery (m^*, σ^*, i^*) from $\mathcal{A}$, output Reject if one of the following three conditions is true:
  - τ contains a key opening query (Open, i^*).
  - τ contains an oracle query $(\mathsf{Query}, i^*, m^*)$.
  - $\mathsf{Ver}_{k_{i^*}}(m^*, \sigma^*) \to \mathsf{Reject}$.
- Otherwise, output Accept.

We call a MAC Π an **adaptively multi-key unforgeable MAC** if it is adaptively $\ell(n)$-key unforgeable for every polynomial $\ell(\cdot)$.

For syntactic clarity, we will assume that a machine interacting with a multi-key MAC adversary will begin interaction with a new instance of the adversary by sending a special message (Init, s), where s is the "identifier" for the instance, and communicate with the adversary by sending a partial transcript and receiving a next message as described above for oracle interaction.

2.3 Intractability Assumptions

We define a notion of "game-based security assumptions" as in [Nao03, Pas11]. Informally, an assumption can be thought of as a pair of a challenger and a threshold function, where an adversary is able to "break" the assumption by causing the challenger to accept an interaction with probability non-negligibly greater than the given threshold.

Definition 3. For polynomial $r(\cdot)$, we call a pair $(\mathcal{C}, t(\cdot))$ an $r(\cdot)$**-round intractability assumption** if $t(\cdot) \in [0, 1]$ is a function and $\mathcal{C}$ is a (possibly randomized) interactive algorithm taking input 1^n and outputting either Accept or Reject after at most $r(n)$ rounds of external communication.

Given a probabilistic interactive algorithm $\mathcal{A}$ which interacts with $\mathcal{C}$, we say that $\mathcal{A}$ **breaks** the assumption $(\mathcal{C}, t(\cdot))$ with some non-negligible probability $p(\cdot)$ if, for infinitely many $n \in \mathbb{N}$: $\Pr[\langle \mathcal{A}, \mathcal{C}\rangle(1^n) = \mathsf{Accept}] \geq t(n) + p(n)$.

Conversely, we refer to $\mathcal{C}$ as **secure** if there exists no $\mathcal{A}$ which breaks $\mathcal{C}$ with non-negligible probability.

Lastly, we call an assumption $(\mathcal{C}, t(\cdot))$ 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.

The general notion of an intractability assumption captures any standard cryptographic assumption, including our earlier definition of adaptive multi-key unforgeability. Specifically, this would be the *unbounded-round* assumption $(\mathcal{C}_{\Pi}^{\ell(n)}, 0)$ (using the challenger defined in Definition 2). Clearly, we cannot hope to rule out tight reductions from, say, adaptive multi-key unforgeability to itself; as such, we focus on ruling out only reductions to bounded-round assumptions, but we note that virtually all "standard" cryptographic assumptions fall into this category. [4]

2.4 Black-Box Reductions

We next formalize what it means to "base the security of one assumption $(\mathcal{C}_1)$ on another assumption $(\mathcal{C}_2)$". Intuitively, this requires a proof that, if there exists an adversary breaking $\mathcal{C}_1$, then there likewise must exist an adversary breaking $\mathcal{C}_2$, which implies the desired result by contrapositive.

In practice, virtually all reductions are "black-box" reductions, where the adversary breaking $\mathcal{C}_2$ is given by an efficient oracle-aided machine $\mathcal{R}$ which interacts in a "black-box" manner with an adversary which breaks $\mathcal{C}_1$ and uses the view of the interaction to break $\mathcal{C}_2$. Formally:

Definition 4. Given a probabilistic polynomial-time oracle-aided algorithm $\mathcal{R}$, we say that $\mathcal{R}$ is 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 algorithm $\mathcal{A}$ that breaks $(\mathcal{C}_1, t_1(\cdot))$ with non-negligible probability $p_1(\cdot)$, $\mathcal{R}^{\mathcal{A}}$ breaks $(\mathcal{C}_2, t_2(\cdot))$ with non-negligible probability $p_2(\cdot)$.

Furthermore, if on common input 1^n $\mathcal{R}^{\mathcal{A}}$ queries $\mathcal{A}$ only on input 1^n, we refer to $\mathcal{R}$ as **fixed-parameter**.

We notably allow reductions to rewind their oracles (by sending a transcript from earlier in the interaction) and even run multiple, potentially interleaved, instances of their oracle.

The restriction to deterministic oracles $\mathcal{A}$ may seem strange at first, but we stress that we can (and will) in fact simply model a randomized oracle by a family of deterministic oracles (where each deterministic oracle represents some fixed setting of the randomness). Using deterministic oracles enables us to reason about cases where the reduction $\mathcal{R}$ can rewind or restart the oracle. We also will restrict to fixed-parameter reductions: this is a restriction inherent to the meta-reduction paradigm, yet it is a natural one (since, as far as we know, all reductions in practice are indeed fixed-parameter).

Of course, we can apply the definition of a reduction to adaptive unforgeability as defined above, using the natural formulation as an intractability assumption:

[4] An example of a "non-standard" assumption that does not fit this definition would be a non-falsifiable assumption, e.g., a "knowledge of exponent" assumption (see, e.g., [Dam92]).

Definition 5. We shall refer to a probabilistic polynomial-time oracle-aided algorithm $\mathcal{R}$ as a **fixed-parameter black-box reduction for basing adaptive $\ell(n)$-key unforgeability of a MAC Π on the hardness of an assumption** $(\mathcal{C}, t(\cdot))$ if it is a fixed-parameter black-box reduction for basing the hardness of assumption $(\mathcal{C}_\Pi^{\ell(n)}, 0)$ on that of $(\mathcal{C}, t(\cdot))$, where $\mathcal{C}_\Pi^{\ell(n)}$ is as given in Definition 2.

We refer to a probabilistic polynomial-time oracle-aided algorithm $\mathcal{R}$ as a **fixed-parameter black-box reduction for basing adaptively secure unforgeability of a MAC Π on the hardness of an assumption** $(\mathcal{C}, t(\cdot))$ if there exists polynomial $\ell(\cdot)$ for which $\mathcal{R}$ is a fixed-parameter black-box reduction for basing adaptively secure $\ell(n)$-key unforgeability of Π on the hardness of $(\mathcal{C}, t(\cdot))$.

2.5 Security Loss

Finally, we define a notion of the "inherent efficiency" of a reduction, or the *security loss*, intuitively representing a worst-case ratio between the "work" (expected time) needed to break the assumption $\mathcal{C}_2$ (i.e., the underlying assumption) and the "primitive" $\mathcal{C}_1$ (in our case, adaptive multi-key unforgeability). If the primitive is significantly easier to break than the underlying assumption, this indicates that the reduction is intuitively "less powerful" at guaranteeing security for the primitive, which corresponds to a higher security loss.

Definition 6. 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 $\mathcal{A}$, we define the following, where $\tau_\mathcal{M}(x)$ denotes the time taken by an algorithm $\mathcal{M}$ in experiment x, $r_\mathcal{A}$ denotes all random coins used by $\mathcal{A}$ and $\mathcal{C}_1$ in the experiment $\langle \mathcal{A}, \mathcal{C}_1 \rangle$, and $r_\mathcal{R}$ denotes all random coins used by $\mathcal{A}$, $\mathcal{C}_2$, and $\mathcal{R}$ in the experiment $\langle \mathcal{R}^\mathcal{A}, \mathcal{C}_2 \rangle$:

- $\mathsf{Success}_\mathcal{A}(n) = \Pr_{r_\mathcal{A}}[\langle \mathcal{A}, \mathcal{C}_1 \rangle_{r_\mathcal{A}}(1^n) = \mathsf{Accept}] - t_1(n)$
- $\mathsf{Success}_{\mathcal{R}^\mathcal{A}}(n) = \Pr_{r_\mathcal{R}}[\langle \mathcal{R}^\mathcal{A}, \mathcal{C}_2 \rangle_{r_\mathcal{R}}(1^n) = \mathsf{Accept}] - t_2(n)$
- $\mathsf{Time}_\mathcal{A}(n) = \max_{r_\mathcal{A}}(\tau_\mathcal{A}([\mathcal{A} \leftrightarrow \mathcal{C}_1]_{r_\mathcal{A}}(1^n)))$
- $\mathsf{Time}_{\mathcal{R}^\mathcal{A}}(n) = \max_{r_\mathcal{R}}(\tau_{\mathcal{R}^\mathcal{A}}([\mathcal{R}^\mathcal{A} \leftrightarrow \mathcal{C}_2]_{r_\mathcal{R}}(1^n)))$.

Then the **security loss** [Lub96] of $\mathcal{R}$ is defined as:

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

If there exists polynomial $p(\cdot)$ for which $\lambda_\mathcal{R}(n) \le p(n)$ given sufficiently large $n \in \mathbb{N}$, we call $\mathcal{R}$ **linear-preserving**. If there exists a constant c for which $\lambda_\mathcal{R}(n) \le c$ given sufficiently large $n \in \mathbb{N}$, we call $\mathcal{R}$ **tight**.

3 Main Theorem

We present our main result, which rules out the possibility of basing the provable security of a deterministic MAC on any "standard" (bounded-round) assumption with a linear-preserving reduction:

Theorem 2. Let Π be a deterministic MAC. If there exists a fixed-parameter black-box reduction $\mathcal{R}$ for basing adaptive multi-key unforgeability of Π on some $r(\cdot)$-round intractability assumption $(\mathcal{C}, t(\cdot))$ (for polynomial $r(\cdot)$), then either:

(1) $\mathcal{R}$ is not a linear-preserving reduction, or

(2) there exists a polynomial-time adversary $\mathcal{B}$ breaking the assumption $(\mathcal{C}, t(\cdot))$.

As we mentioned in the introduction, Theorem 2 can be generalized fairly directly to apply as written to several other primitives besides simply deterministic MACs; however, as we focus on the case of MACs in this paper, we present our result for deterministic MACs in full here and opt to refer the interested reader to the appendices for detailed discussion of its applications to other primitives. Specifically, Appendix A shows that we can rule out linear-preserving reductions from adaptively multi-key unforgeable deterministic *digital signature schemes* to bounded-round assumptions, while Appendix B demonstrates that we can rule out linear-preserving reductions from adaptive multi-key *pseudorandomness* of a family of functions (i.e., adaptive multi-key PRFs).

To prove Theorem 2, we first present the following crucial lemma, which we prove in full in Section 4:

Lemma 1. Let Π be a deterministic MAC, 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 a fixed-parameter black-box reduction $\mathcal{R}$ for basing adaptive $\ell(n)$-key unforgeability of Π on the hardness of $(\mathcal{C}, t(\cdot))$, then either $\mathcal{R}$'s security loss is at least

$$\lambda_{\mathcal{R}}(n) \geq \left(1 - \frac{1}{2\ell(n)^2}\right)\left(\sqrt{\ell(n)} - (r(n)+2)\right)$$

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

Because $p(\cdot)$ in the definition of a linear-preserving reduction is an *a priori* fixed polynomial, and in particular cannot depend on $\ell(n)$, this lemma will prove Theorem 2, as follows:

Proof. Let $\mathcal{R}$ be a reduction from adaptive multi-key unforgeability of Π to the hardness of $(\mathcal{C}, t(\cdot))$. Assume for the sake of contradiction that Lemma 1 is true, yet $\mathcal{R}$ is linear-preserving and $(\mathcal{C}, t(\cdot))$ is secure. Because $\mathcal{R}$ is linear-preserving, there is some polynomial $p(\cdot)$ such that $\lambda_{\mathcal{R}}(n) \leq p(n)$ for sufficiently large n. Furthermore, $\mathcal{R}$ is by definition a reduction from adaptive $\ell(n)$-key unforgeability for *every* polynomial $\ell(n)$, including, say, $\ell(n) = (2p(n) + r(n) + 3)^2$, so by Lemma 1 we have:

$$\lambda_{\mathcal{R}}(n) \geq \left(1 - \frac{1}{2\ell(n)^2}\right)\left(\sqrt{\ell(n)} - (r(n)+2)\right) \geq \frac{1}{2}(2p(n)+1) > p(n)$$

which is a clear contradiction.

□

3.1 Technical Overview

Next, we shall explain the methodology for the proof of Lemma 1 at a high level.

The ideal adversary. We begin by constructing and investigating an "ideal" adversary $\mathcal{A}$. To summarize, $\mathcal{A}$ will first make $q(n)$ random tag queries (where $q(n)$ is a polynomial to be determined later) to each of the $\ell(n)$ instances of the MAC Π, continue by opening all but one of the keys in a random order (while also verifying that the challenger or $\mathcal{R}$ answered its queries consistently with the opened keys), and lastly, if it received correct responses for the opened instances, use the information gained from the queries for the remaining instance to attempt to brute-force a forgery for that instance. (On the other hand, if verification fails, $\mathcal{A}$ will "reject", returning $\perp$ instead of a forgery.)

In virtually all meta-reductions to date, the ideal adversary is able to perfectly brute-force the challenger's secret information and break the primitive with probability 1. Here, however, that is not the case; $\mathcal{A}$ is limited to a polynomial number of tag queries (which is necessary for simulatability) and furthermore has no way to publicly verify whether a certain key or forgery is correct. The most $\mathcal{A}$ can do, in fact, is brute-force the set of all keys consistent with the tag queries it makes for the unopened instance, pick one of those keys, and use it to generate a forgery in the hopes that it will match with the key the challenger has selected.

This is where the "key uniqueness" property discussed in the introduction will first factor in. We show that, since the key picked by the adversary agrees with the key picked by the challenger on all $q(n)$ tag queries, then it must with overwhelming probability also agree on a large fraction $(1-2n/q(n))$ of possible messages. Hence, $\mathcal{A}$ will have a $1-2n/q(n)$ chance of producing a correct forgery when it evaluates the Tag function using the key it extracts on a random message m^* (i.e., the message it eventually will randomly select for its forgery)—that is, $\mathsf{Success}_{\mathcal{A}}(n) \geq 1-2n/q(n)$.

Before proceeding to discuss the meta-reduction, we need to address one final technical issue with the ideal adversary. Namely, since $\mathcal{A}$ works by returning the "next-message" function given a transcript of the interaction thus far, we need to ensure that $\mathcal{R}$ must actually complete the full interaction with $\mathcal{A}$ in order to cause $\mathcal{A}$ to accept and return a forgery, rather than potentially guessing a "fake" accepting transcript for a later point in the interaction to "skip" or avoid responding to certain queries from $\mathcal{A}$. In particular, a reduction $\mathcal{R}$ that skips key-opening queries would be extremely problematic in our analysis of the meta-reduction later on, since the meta-reduction will rely on $\mathcal{R}$'s responses to these queries to properly emulate the ideal adversary $\mathcal{A}$.

Unfortunately, it turns out that $\mathcal{A}$'s key-opening queries, since they convey no information besides the instance to open, have low entropy and thus are easy to predict (and skip) by $\mathcal{R}$. To fix this, we introduce additional "dummy" queries—specifically, random tag queries to instances whose keys have not yet been opened—made after each of the key-opening queries. These serve the purpose of increasing the entropy present in the key-opening phase of the transcript—which guarantees that $\mathcal{R}$ must answer all $\ell(n)-1$ of $\mathcal{A}$'s key-opening queries to successfully complete the interaction (unless it can correctly guess the random input for the dummy query)—but are otherwise ignored.

The meta-reduction. In our discussion of $\mathcal{A}$, we were able to bound $\mathsf{Success}_{\mathcal{A}}(n)$; thus, we turn next to investigating $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$. To do this, we construct a *meta-reduction* $\mathcal{B}$ which runs $\mathcal{R}$ while attempting to efficiently emulate the interaction between $\mathcal{R}$ and $\mathcal{A}$. $\mathcal{B}$ will simulate instances of $\mathcal{A}$ by, exactly as before, making $q(n)$ random tag queries to each instance, opening the key for all but one instance (in a random order and with the interleaved tag queries as above), and checking $\mathcal{R}$'s responses for consistency.

The key difference, of course, is that $\mathcal{B}$ cannot brute-force a forgery; instead, for the unopened instance, $\mathcal{B}$ will attempt to extract a correct key from $\mathcal{R}$ by rewinding the interaction to the key-opening queries and substituting the unopened instance for each other instance in turn. If $\mathcal{R}$ responds to any of the valid queries with a key that matches with the tag queries for that instance, then $\mathcal{B}$ will apply that key to a random message m^* to generate a forgery. If $\mathcal{B}$ does not receive a valid key in this fashion, then it will abort, returning Fail.

Notably, $\mathcal{B}$ will also have to ignore rewindings where, before returning its response to the key-opening query, either $\mathcal{R}$ attempts to communicate externally with $\mathcal{C}$ (which could change the state of the challenger if forwarded), $\mathcal{R}$ requests a forgery from another instance of $\mathcal{A}$ (as this would require additional "nested" rewinding which could grow exponentially), or $\mathcal{R}$ would rewind $\mathcal{A}$ (which precludes $\mathcal{R}$ returning a key); this will factor into the analysis of the failure probability later.

The main task in proving our lemma, then, reduces to that of bounding $\Pr[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \rightarrow \mathsf{Accept}] - \Pr[\langle \mathcal{B}, \mathcal{C}\rangle \rightarrow \mathsf{Accept}]$. Intuitively, if we come up with such a bound (call it $p(n)$), then, if $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}$ is non-negligibly higher than $p(n)$—that is, $\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle$ accepts with such a probability—then $\langle \mathcal{B}, \mathcal{C}\rangle$ will accept with non-negligible probability, hence breaking $\mathcal{C}$. Bounding this probability $p(n)$ is in fact quite non-trivial, as one cannot, say, naïvely apply earlier techniques for meta-reduction analysis to the meta-reduction $\mathcal{B}$. Intuitively, this is because we no longer have a strong "uniqueness" property characteristic of meta-reductions to date—that is, there is no longer a *unique* possible valid forgery $\mathcal{B}$ can extract from its rewinding. Not only does this make it difficult to guarantee that $\mathcal{A}$ and $\mathcal{B}$ produce close distributions of forgeries, but, in conjunction with $\mathcal{B}$'s rewinding strategy, this makes analyzing the failure probability problematic for more complex reasons. For example, consider a reduction $\mathcal{R}$ which might try to rewind $\mathcal{A}$ and change its responses to queries in order to attempt to change the forgery generated; it is straightforward to see that proof techniques such as that of [MP18] immediately fail (due to a potentially unbounded number of nested forgery requests) if $\mathcal{R}$ can theoretically expect to receive many different forgeries by repeatedly rewinding the same instance.

A "hybrid" meta-reduction. We present a way to effectively separate dealing with the issues of uniqueness and rewinding, namely by defining a "hybrid" meta-reduction $\mathcal{B}'$ which, while inefficient, is easy to compare to either $\mathcal{A}$ or $\mathcal{B}$.

At a high level, we construct $\mathcal{B}'$ so that it behaves identically to $\mathcal{B}$ as long as there is only a single possible forgery to return, and so that it behaves identically to $\mathcal{A}$ whenever rewinding succeeds. More specifically, it acts identically to $\mathcal{B}$ until after rewinding finishes, then, if it obtains a forgery, brute-forces one in the same manner as $\mathcal{A}$.

Clearly, $\mathcal{B}'$ can only diverge from $\mathcal{B}$ if the forgery $\mathcal{B}$ extracts is different from the one $\mathcal{B}'$ brute-forces. A straightforward application of our earlier "key uniqueness" lemma shows that this happens with at most $2n/q(n)$ probability per forgery returned by $\mathcal{B}'$.

On the other hand, $\mathcal{B}'$ will always return the same forgery as $\mathcal{A}$ *if* it returns a forgery, but we still need to determine the probability with which $\mathcal{B}'$ fails to return a forgery due to unsuccessful rewinding. Luckily, since $\mathcal{B}'$ now *does* have the uniqueness property, we can proceed along the same lines as in [MP18] and bound the rewinding failure probability by effectively bounding the probability that a randomly chosen ordering of key-opening queries can result in rewinding failure (while assuming that the rest of the randomness in the interaction is fixed arbitrarily, as, if the bound applies to arbitrarily fixed randomness, it must likewise apply when taken over all possible assignments of the same randomness).

The intuition behind the argument is that, if we assume a bound of $W(n)$ on the number of times $\mathcal{R}$ will rewind past when $\mathcal{B}'$ generates the ordering π of the key-opening queries (and note that, due to uniqueness and careful construction, $W(n)$ will also be a bound on the number of distinct forgery requests $\mathcal{R}$ can make, as we show that any others will be internally simulatable and thus "pointless"), every sequence π that causes $\mathcal{B}'$ to fail must do so because all of its rewindings fail, and the rewindings specifically correspond to other sequences π that can occur. Furthermore, if a rewinding fails due to $\mathcal{R}$ responding to a query incorrectly (as opposed to, e.g., external communication or a nested forgery request), then this rewinding corresponds to a "good" sequence where $\mathcal{A}$ and $\mathcal{B}'$ return $\perp$ (and emulation is successful). So, if some sequence π contains more than $W(n)+r(n)+1$ queries at which rewindings of other sequences fail, then, since we can have at most $W(n)$ (unique) forgery requests and $r(n)$ rounds of external communication, at least one query must fail due to an incorrect response, which shows that π is a "good" sequence. A counting argument then allows us to achieve a bound of $(W(n)+r(n)+1)/\ell(n)$ on the failure probability of $\mathcal{B}'$ each time it performs rewinding, or $W(n)(W(n)+r(n)+1)/\ell(n)$ overall failure probability.

Bounding security loss. Combining all of the facts so far, we know that the above quantity is equivalent to the probability with which $\mathcal{A}$ and $\mathcal{B}'$ diverge, while the probability with which $\mathcal{B}'$ and $\mathcal{B}$ diverge is $2nW(n)/q(n)$ (i.e., the probability that uniqueness fails for at least one of the $W(n)$ forgeries). Thus, $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$, as we have argued, is bounded above by the sum of these, which (taking $q(n)$ sufficiently large) is at most $W(n)(W(n)+r(n)+2)/\ell(n)$. Furthermore, $\mathsf{Time}_{\mathcal{R}^{\mathcal{A}}}(n)/\mathsf{Time}_{\mathcal{A}}(n) \geq W(n)$ by our assumption that in the worst case $\mathcal{R}$ runs $W(n)$ instances of $\mathcal{A}$. Lastly, $\mathsf{Success}_{\mathcal{A}}(n) \geq 1 - 2n/q(n)$ as we noted earlier.

Hence, either $(\mathcal{C}, t(\cdot))$ is insecure (and our bound for $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$ does not apply), or, by the above facts and case analysis (to deal with the possibility that $W(n)$ might be arbitrarily large), we obtain the result:

$$\lambda_{\mathcal{R}}(n) \geq \left(1 - \frac{1}{2\ell(n)^2}\right)\left(\sqrt{\ell(n)} - (r(n)+2)\right)$$

4 Proof of Lemma 1

We continue by formally proving Lemma 1. Assume a deterministic MAC Π, a reduction $\mathcal{R}$, and an assumption $(\mathcal{C}, t(\cdot))$ as defined in the statement of Lemma 1. Consider an ideal but inefficient adversary $\mathcal{A}$, which technically is given by a random selection from a *family* of inefficient adversaries $\mathcal{A} \leftarrow \{\mathcal{A}^{\mathcal{O}}\}$ (where $\mathcal{O}$ is a uniformly chosen random function) defined as in Figures 1 and 2; also consider an efficient meta-reduction $\mathcal{B}$ defined as in Figures 3 and 4.

Before analyzing the properties of $\mathcal{A}$ and $\mathcal{B}$, we verify that $\mathcal{B}$ runs efficiently through the following claim:

Claim 1. $\mathcal{B}(1^n)$ runs in time polynomial in n.

Proof. First consider the execution of $\mathcal{B}$ without rewinding. Because $\mathcal{R}$ is an efficient reduction by definition, $\mathcal{R}$ can begin at most a polynomial number of instances of the adversary $\mathcal{A}$, which means at most some polynomial number (henceforth we shall call this bound $M(n)$) of simulated instances of $\mathcal{A}$ are invoked during the execution of $\mathcal{B}$.

Now, for each instance of $\mathcal{A}$, $\mathcal{B}$ sends a total of $q(n)(\ell(n)+1)$ tag queries, $\ell(n)-1$ key opening queries, and a forgery, for a total of $(q(n)+2)\ell(n)$ total messages. It is clear that each message prior to the forgery takes at most a polynomial amount of time to compute (since no rewinding is involved until the forgery and all other computations, including the Valid predicate, are efficient). Furthermore, there are at most $r(n)$ rounds of communication between $\mathcal{B}$ and $\mathcal{C}$ that must be forwarded, but each of these is guaranteed to be efficiently computable as well because of $\mathcal{R}$'s efficiency.

To factor in rewinding, consider that $\mathcal{R}$ can rewind $\mathcal{A}$, but only up to a polynomial number of times; $\mathcal{B}$ also rewinds simulated instances of $\mathcal{A}$ while attempting to extract a valid forgery key from $\mathcal{R}$, but once again also only up to a polynomial number ($\ell(n)$) of times. Hence, rewinding will at most multiply the running time of $\mathcal{B}$ by a polynomial factor, which results in a polynomial total running time since $\mathcal{B}$ without rewinding is polynomial-time.

□

4.1 Analyzing the Ideal Adversary

In order to establish a bound to the security loss $\lambda_{\mathcal{R}}(n)$, we shall determine bounds for $\mathsf{Success}_{\mathcal{A}}(n)$ and $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$; time analysis will follow naturally.

We begin by analyzing the probability $\mathsf{Success}_{\mathcal{A}}(n)$. This is fairly straightforward, following from the critical "key uniqueness" lemma which states that two keys agreeing on all of the $q(n)$ tag queries made by $\mathcal{A}$ are overwhelmingly likely to agree on "most" messages m. Hence, the key chosen by $\mathcal{A}$, even if not the same as that chosen by the challenger, is by definition consistent with it on all of the tag queries and thus should agree on a large fraction of the possible forgery inputs m^*. Formally:

Claim 2. There exists a negligible function $\nu(\cdot)$ such that:

$$\mathsf{Success}_{\mathcal{A}}(n) \geq 1 - \frac{2n}{q(n)} - \nu(n)$$

Proof. The claim follows readily from the following lemma (and the fact that there is only a negligible chance that $\mathcal{A}$ generates an invalid m^*):

Lemma 2. There exists negligible $\nu(\cdot)$ such that, for any family of functions $\mathcal{U} = \{f_k : \mathcal{X}_n \to \mathcal{Y}_n\}_{k\in\{0,1\}^n, n\in\mathbb{N}}$, except with probability $\nu(n)$ over $q(n)$ uniformly random queries $(x_{1,j^*}, \ldots, x_{q(n),j^*}) \leftarrow (\mathcal{X}_n)^{q(n)}$, for any $k_1, k_2 \in (\{0,1\}^n)^2$ such that $f_{k_1}(x_{i,j^*}) = f_{k_2}(x_{i,j^*})$ for all $i \in [q(n)]$, it is true that:

$$\Pr\left[x^* \leftarrow \mathcal{X}_n : f_{k_1}(x^*) = f_{k_2}(x^*)\right] \geq 1 - \frac{2n}{q(n)} \qquad (1)$$

Proof. For any key pair (k_1, k_2), let

$$S_{k_1,k_2} \triangleq \{x^* \in \mathcal{X}_n : f_{k_1}(x^*) = f_{k_2}(x^*)\}$$

be the set of inputs where the two keys' outputs are identical.

So, if (1) is false for some pair (k_1, k_2), i.e., $|S_{k_1,k_2}| \leq |\mathcal{X}_n| \left(1 - \frac{2n}{q(n)}\right)$; then the probability over $\{x_{i,j^*}\}$ that both keys agree in all $q(n)$ queries to f made by $\mathcal{A}$, or equivalently the probability that

- On receiving an initialization message (Init, s), let $m_{1,1}$ denote a uniformly random message in $\mathcal{M}_n$ generated by random coins resulting from applying the oracle $\mathcal{O}$ to the input $(s, 1, 1, 1)$, and send $(\mathsf{Query}, 1, m_{1,1})$.
- On receiving a transcript of the form

$$\tau = (q_{1,1}, q_{1,2}, \ldots, q_{1,\ell(n)}, q_{2,1}, \ldots, q_{i,j})$$

  where either $i < q(n)$ or $i = q(n)$ and $j < \ell(n)$, such that each $q_{u,v}$ is of the form $((\mathsf{Query}, v, m_{u,v}), \sigma_{u,v})$, do the following:

  - Let $j' = (j \bmod \ell(n)) + 1$.
  - Let $i' = i + 1$ if $j' = 1$ and $i' = i$ otherwise.
  - Let $m_{i',j'}$ be a uniformly random message in $\mathcal{M}_n$ generated by random coins resulting from applying the oracle $\mathcal{O}$ to the input $(s, i', j', 1)$.
  - Return $(\mathsf{Query}, j', m_{i',j'})$.

- On receiving a transcript of the form $\tau = \tau_1 || \tau_2$, where

$$\tau_1 = (q_{1,1}, q_{1,2}, \ldots, q_{1,\ell(n)}, q_{2,1}, \ldots, q_{q(n),\ell(n)})$$

  and where each $q_{u,v}$ is of the form $((\mathsf{Query}, v, m_{u,v}), \sigma_{u,v})$, do the following:

  - Let c be the number of Open queries that have so far appeared in τ_2.
  - Let $\pi = (\pi_1, \ldots, \pi_{\ell(n)})$ be a uniformly random permutation of $[\ell(n)]$, generated by random coins resulting from applying $\mathcal{O}$ to the input τ_1.
  - If τ_2 is empty or ends with messages of the form $((\mathsf{Open}, j), k_j)$, then:
    * Generate ω_{c+1} as a uniformly random message in $\mathcal{M}_n$ generated by random coins resulting from applying the oracle $\mathcal{O}$ to the input $\tau_1 || (s, q(n) + 1, c + 1, \mathsf{Valid}(\mathcal{O}, \tau^*, s))$ and return $(\mathsf{Query}, q, \omega_{c+1})$, where q is the lexicographically first instance for which τ_2 does not contain an Open query.
  - Otherwise, if $c < \ell(n) - 1$ and the last part of τ_2 contains messages of the form $((\mathsf{Query}, q, \omega_{c+1}), \cdot)$, then return $(\mathsf{Open}, \pi_{c+1})$.
  - Lastly, if τ_2 ends with $((\mathsf{Query}, q, \omega_{\ell(n)}), \cdot)$ and $c = \ell(n) - 1$, return a forgery as follows:
    * If $\mathsf{Valid}(\mathcal{O}, \tau, s) = 0$, return $\perp$.
    * Otherwise, use exhaustive search to find the set K^* of all keys k^* such that, given $j^* = \pi_{\ell(n)}$ as determined above, and for each $i \in [q(n)]$, $\mathsf{Ver}_{k^*}(m_{i,j^*}, \sigma_{i,j^*}) = \mathsf{Accept}$. If K^* is empty then return $\perp$.
    * Lastly, using random coins generated by applying $\mathcal{O}$ to a new input $\tau_1 || (s, q(n) + 2, 0, 1)$, generate a uniformly random message m^* (which will be distinct from all m_{i,j^*} with all-but-negligible probability) and take the *lexicographically first* element k^* of K^*. Return the forgery $(m^*, \mathsf{Tag}_{k^*}(m^*), j^*)$.

Figure 1: Formal description of the "ideal" adversary $\mathcal{A}^{\mathcal{O}}$ (1).

$q(n)$ uniformly random queries $\{x_{i,j^*}\}$ lie in S_{k_1,k_2}, is bounded above by:

$$\left(1 - \frac{2n}{q(n)}\right)^{q(n)} = \left(\left(1 - \frac{2n}{q(n)}\right)^{q(n)/2n}\right)^{2n} < \left(\frac{1}{e}\right)^{2n} = \exp(-2n)$$

There exist no more than $(2^n)^2 = 2^{2n}$ possible key pairs $(k_1, k_2) \in (\{0,1\}^n)^2$, each of which by the

Let the predicate $\mathsf{Valid}(\mathcal{O}, \tau, s)$ be defined as follows:

- Parse τ as $\tau_1 || \tau_2$, where
$$\tau_1 = (q_{1,1}, q_{1,2}, \ldots, q_{1,\ell(n)}, q_{2,1}, \ldots, q_{q(n),\ell(n)})$$
such that each $q_{u,v}$ is of the form $((\mathsf{Query}, v, m_{u,v}), \sigma_{u,v})$. If τ cannot be parsed as such, return 0.
- Let $\pi = (\pi_1, \ldots, \pi_{\ell(n)})$ be a permutation of $[\ell(n)]$ generated in the same manner as in $\mathcal{A}$, using random coins generated by applying $\mathcal{O}$ to the input τ_1.
- Parse
$$\tau_2 = (q_1^*, q_2^*, \ldots, q_c^*[, q_{c+1}^*])$$
such that each q_i^* is of the form $((\mathsf{Query}, q_i, \omega_i), \cdot, (\mathsf{Open}, \pi_i), k_{\pi_i})$ and q_{c+1}^*, if present, is of the form $((\mathsf{Query}, q_i, \omega_{c+1}), \cdot)$. If τ_2 cannot be parsed as such, or if $c > \ell(n) - 1$, return 0.
- Verify that each q_i in τ_2 is equal to the lexicographically first instance $q \in [\ell(n)]$ such that q does not appear in an Open query earlier in τ_2. If not true, return 0.
- Verify that, for all $i \in [q(n)]$ and all $j \in \{\pi_1, \ldots, \pi_c\}$, $\mathsf{Ver}_{k_j}(m_{i,j}, \sigma_{i,j}) = \mathsf{Accept}$. (Do not verify the responses to queries ω_i in τ_2.) If not true, return 0.
- Verify that every $m_{i,j}$ parsed from the transcript is correctly generated by random coins resulting from applying $\mathcal{O}$ to the input $(s, i, j, 1)$ (for $i \in [q(n)]$) and that each ω_j is correctly generated by random coins resulting from applying $\mathcal{O}$ to the input $\tau_1 || (s, q(n)+1, j, 1)$. If not true, return 0.
- Otherwise, return 1.

Figure 2: Formal description of the "ideal" adversary $\mathcal{A}^{\mathcal{O}}$ (2).

above must either have the property (1) or be such that

$$\Pr\left[(x_{1,j^*}, \ldots, x_{q(n),j^*}) \leftarrow (\mathcal{X}_n)^{q(n)} : f_{k_1}(x_{i,j^*}) = f_{k_2}(x_{i,j^*}) \forall i \in [q(n)]\right]$$
$$= \Pr\left[(x_{1,j^*}, \ldots, x_{q(n),j^*}) \leftarrow (\mathcal{X}_n)^{q(n)} : x_{i,j^*} \in S_{k_1,k_2} \forall i \in [q(n)]\right]$$
$$\leq \exp(-2n)$$

Then the probability over $\{x_{i,j^*}\}$ that *some* key pair exists which does not have property (1) yet does have $f_{k_1}(x_{i,j^*}) = f_{k_2}(x_{i,j^*})$ for all x_{i,j^*} is, by a union bound, at most:

$$\Pr\Big[(x_{1,j^*}, \ldots, x_{q(n),j^*}) \leftarrow (\mathcal{X}_n)^{q(n)} : \exists (k_1, k_2) \in (\{0,1\}^n)^2 :$$
$$x_{i,j^*} \in S_{k_1,k_2} \forall i \in [q(n)] \text{ and } |S_{k_1,k_2}| \leq |\mathcal{X}_n| \left(1 - \frac{2n}{q(n)}\right)\Big]$$
$$\leq \sum_{(k_1,k_2) \in (\{0,1\}^n)^2} \mathbf{1}_{|S_{k_1,k_2}| \leq |\mathcal{X}_n|(1-2n/q(n))} \Pr\Big[(x_{1,j^*}, \ldots, x_{q(n),j^*}) \leftarrow (\mathcal{X}_n)^{q(n)} :$$
$$x_{i,j^*} \in S_{k_1,k_2} \forall i \in [q(n)]]$$
$$< 2^{2n} e^{-2n} = (2/e)^{2n}$$

which is clearly negligible in n. □

To prove the claim, we consider the above lemma, letting f_k be the deterministic function Tag_k. When interacting with an honest challenger, the responses to tag queries for each instance will always be consistent with the respective keys, and so $\mathcal{A}$ will never return $\perp$ due to the Valid

predicate failing or K^* being empty. Furthermore, for the instance $\pi_{\ell(n)}$ for which $\mathcal{A}$ outputs a forgery, it is overwhelmingly likely (with probability $1 - \nu(n)$), by Lemma 2, that all keys in the set K^* recovered by $\mathcal{A}$ will agree with the *correct* (challenger's) key k' for that instance on a large $(1 - 2n/q(n))$ fraction of random messages m^*. Specifically, this means that, given any choice of key k^* from K^*, $\mathcal{A}$ will produce a correct forgery (m^*, σ^*) (i.e., such that $\sigma^* = \mathsf{Tag}_{k'}(m^*)$, or equivalently $\mathsf{Ver}_{k'}(m^*, \sigma^*) = \mathsf{Accept}$) given random m^* with probability at least $1 - 2n/q(n)$.

Thus, $\mathcal{A}$ succeeds in the interaction in the event that Lemma 2 does not fail (i.e., property (1) holds for every key pair) and that $\mathcal{A}$ chooses a "good" m^* (i.e., one which does not repeat a previous query and produces the same tag under k^* as under the challenger's key k') given its choice of $k^* \leftarrow K^*$; the claim follows from the union bound over these events. □

We require one additional claim concerning the adversary, which states that the reduction $\mathcal{R}$ must have actually responded to all $\ell(n) - 1$ key-opening queries to have a non-negligible chance of receiving a forgery. This will be important later, to ensure that $\mathcal{R}$ cannot "cheat" by sending a fake transcript while interacting with $\mathcal{B}$.

Claim 3. 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 responses to *all* $\ell(n) - 1$ (Open, i) (key-opening) queries from $\mathcal{R}$, is less than $\nu(n)$.

Proof. We demonstrate that, if $\mathcal{A}$ returns a forgery (i.e., not $\perp$) to $\mathcal{R}$ after $\mathcal{R}$ responds to strictly fewer than $\ell(n) - 1$ distinct key-opening queries from $\mathcal{A}$, then this requires $\mathcal{R}$ to guess a uniformly random message generated using the output of $\mathcal{A}$'s random oracle $\mathcal{O}$ on a new input, which can happen with at most probability $p(n)/|\mathcal{M}_n|$ for some polynomial $p(\cdot)$ due to $\mathcal{O}$ being uniformly random.

Assume that $\mathcal{R}$ responds to fewer than $\ell(n) - 1$ key-opening queries. Then there exists some $i \in [\ell(n) - 1]$ for which $\mathcal{R}$ does not send $\mathcal{A}$ a partial transcript ending with $((\mathsf{Open}, \pi_i), k_{\pi_i})$ (i.e., a response to $\mathcal{A}$'s i^{th} key-opening query). By the definition of the Valid predicate, in order for $\mathcal{R}$ to receive a final message from $\mathcal{A}$ that contains a forgery (and not $\perp$), $\mathcal{R}$ must send to $\mathcal{A}$ a complete transcript

$$\tau = \tau_1 || (\ldots, (\mathsf{Open}, \pi_i), k_{\pi_i}, (\mathsf{Query}, q, \omega_{i+1}), \ldots)$$

where ω_{i+1} is a uniformly random message generated by random coins resulting from applying $\mathcal{O}$ to $\tau_1 || (s, q(n) + 1, i + 1, 1)$.

By construction of $\mathcal{A}$ and the assumption that $\mathcal{R}$ does not send $\mathcal{A}$ a partial transcript ending with $((\mathsf{Open}, \pi_i), k_{\pi_i})$, however, $\mathcal{R}$ can never have received either ω_{i+1} or any message depending on the correct input $\tau_1 || (s, q(n) + 1, i + 1, 1)$ to $\mathcal{O}$. Hence, since ω_{i+1} is uniformly distributed and independent of any other message, we can conclude that $\mathcal{R}$ will send the correct ω_{i+1} in its final transcript with at most probability $1/|\mathcal{M}_n|$ (i.e., by guessing a random message correctly). While $\mathcal{R}$ can attempt to retrieve a forgery multiple times, it is restricted to polynomial time, so the probability with which it can guess ω_{i+1} (which is necessary to receive a forgery from $\mathcal{A}$) is bounded above by $\nu(n) = p(n)/|\mathcal{M}_n|$ for polynomial $p(\cdot)$, which is negligible because we assume the message space to be super-polynomial (asymptotically greater than any polynomial) in n.

□

- Set initial view $v \leftarrow \perp$ and set $J \leftarrow 1$. Execute $\mathcal{R}$, updating the current view v according to the following rules.
- When $\mathcal{R}$ begins a new instance of $\mathcal{A}$ with some message (Init, s), label this instance as instance J. Generate and store $\ell(n)q(n)$ uniformly random queries $\vec{m}_J^1 = (m^1_{J,1,1}, \ldots, m^1_{J,q(n),\ell(n)})$. Also initialize a variable $k_J \leftarrow \{\}$. Lastly, respond with $\tau_J^* = (\mathsf{Query}, 1, m^1_{J,1,1})$ and increment J.
- 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.
- For any $i \in [q(n)]$, when $\mathcal{R}$ sends to some instance I of $\mathcal{A}$ a transcript of the form

$$\tau = (q_{1,1}, q_{1,2}, \ldots, q_{1,\ell(n)}, q_{2,1}, \ldots, q_{i,j})$$

  where either $i < q(n)$ or $i = q(n)$ and $j < \ell(n)$, such that each $q_{u,v}$ is of the form $((\mathsf{Query}, v, m_{u,v}), \sigma_{u,v})$, do the following:
  - Let $j' = (j \bmod \ell(n)) + 1$.
  - Let $i' = i + 1$ if $j' = 1$ and $i' = i$ otherwise.
  - Return the response $(\mathsf{Query}, j', m^1_{I,i',j'})$.
- When $\mathcal{R}$ sends to some instance I of $\mathcal{A}$ a transcript of the form $\tau = \tau_1 || \tau_2$, where

$$\tau_1 = (q_{1,1}, q_{1,2}, \ldots, q_{1,\ell(n)}, q_{2,1}, \ldots, q_{q(n),\ell(n)})$$

  and where each $q_{u,v}$ is of the form $((\mathsf{Query}, v, m_{u,v}), \sigma_{u,v})$, do the following:
  - Let c be the number of Open messages appearing in τ_2 so far.
  - If there is some tuple $(\tau_1, I, \pi, \vec{\omega}, m^*)$ stored, let π, $\vec{\omega}$, and m^* be as stored in the tuple. Otherwise, let $\pi = (\pi_1, \ldots, \pi_{\ell(n)})$ be a uniformly random permutation of $[\ell(n)]$, generate $2\ell(n)$ additional messages $\vec{\omega} = (\omega^0_1, \ldots, \omega^0_{\ell(n)}, \omega^1_1, \ldots, \omega^1_{\ell(n)})$ and a target forgery m^*, and store the tuple $(\tau_1, I, \pi, \vec{\omega}, m^*)$.
  - Consider the suffix transcript τ_2. If τ_2 is empty or ends with messages of the form $((\mathsf{Open}, j), k_j)$, then return $(\mathsf{Query}, q, \omega_{c+1}^{\mathsf{Valid}(\tau,I)})$, where q is the lexicographically first instance for which τ_2 does not contain an Open query.
  - Otherwise, if $c < \ell(n) - 1$ and τ_2 ends with messages of the form $((\mathsf{Query}, q, \omega_{c+1}), \cdot)$, then return $(\mathsf{Open}, \pi_{c+1})$.
  - Otherwise, if τ_2 ends with $((\mathsf{Query}, q, \omega_{\ell(n)}), \cdot)$ and $c = \ell(n) - 1$, generate a forgery as follows:
    - If $\mathsf{Valid}(\tau, I) = 0$, then return $\perp$.
    - Otherwise, run the procedure Rewind detailed below for the instance I.
    - If, after running Rewind, there is a stored key k_I, then return the forgery $(m^*, \mathsf{Tag}_{k_I}(m^*), \pi_{\ell(n)})$ and continue executing $\mathcal{R}$ as above. Otherwise, abort the entire execution of $\mathcal{B}$ and return Fail.

Let the predicate $\mathsf{Valid}(\tau, I)$ be defined as follows:

- Parse τ as $\tau_1 || \tau_2$, where $\tau_1 = (q_{1,0}, q_{1,1}, \ldots, q_{1,\ell(n)-1}, q_{2,0}, \ldots, q_{q(n),\ell(n)})$ such that each $q_{u,v}$ is of the form $((\mathsf{Query}, v, m_{u,v}), \sigma_{u,v})$. If τ cannot be parsed as such, return 0.

Figure 3: Formal description of the meta-reduction $\mathcal{B}$ (1).

- If there is a stored tuple $(\tau_1, I, \pi, \vec{\omega}, m^*)$ then set $\pi = (\pi_1, \ldots, \pi_{\ell(n)})$ equal to the third element of this tuple and set $\vec{\omega} = (\omega_1^0, \ldots, \omega_{\ell(n)}^0, \omega_1^1, \ldots, \omega_{\ell(n)}^1)$ equal to the fourth element. If there is no such tuple then return 0.
- Parse $\tau_2 = (q_1^*, q_2^*, \ldots, q_c^*[, q_{c+1}^*])$ such that each q_i^* is of the form $((\mathsf{Query}, q_i, \omega_i), \cdot, (\mathsf{Open}, \pi_i), k_{\pi_i})$ and q_{c+1}^*, if present, is of the form $((\mathsf{Query}, q_i, \omega_{c+1}), \cdot)$. If τ_2 cannot be parsed as such, or if $c > \ell(n) - 1$, return 0.
- Verify that each q_i in τ_2 is equal to the lexicographically first instance $q \in [\ell(n)]$ such that q does not appear in an Open query earlier in τ_2. If not true, return 0.
- Verify that, for all $i \in [q(n)]$ and all $j \in \{\pi_1, \ldots, \pi_c\}$, $\mathsf{Ver}_{k_j}(m_{i,j}, \sigma_{i,j}) = \mathsf{Accept}$. (Do not verify the responses to queries ω_i in τ_2.) If not true, return 0.
- Verify that every $m_{i,j}$ parsed from the transcript τ_1 is equal to the stored $m_{I,i,j}^1$, and that every ω_j parsed from τ_2 is equal to the respective ω_j^1. If not, then return 0. Otherwise, return 1.

Rewind procedure:

- Given instance I, for $j \in [\ell(n)]$ let V^j denote the view immediately before the query $(\omega_{\pi_j}, (\mathsf{Open}, \pi_j))$ for instance I (i.e., the query corresponding to the opening of the j^{th} instance after the order of instances π to open is randomized).
- For $j \in [\ell(n)]$, "rewind" the view to V^j as follows: Let $J' \leftarrow J$, let π' be identical to π except with $\pi_{\ell(n)}$ and π_j swapped (i.e., $\pi'_j = \pi_{\ell(n)}$ and $\pi'_{\ell(n)} = \pi_j$), and begin executing $\mathcal{R}$ from the view $V' \leftarrow V^j$ as in the main routine, with the following exceptions:
  - Replace any instances of π with π' (including in Valid).
  - When $\mathcal{R}$ begins a new instance of $\mathcal{A}$, label this instance as instance J' and increment J'.
  - When $\mathcal{R}$ attempts to communicate externally with $\mathcal{C}$ or "rewind" the current instance of $\mathcal{A}$ by sending a message corresponding to a point in the interaction before V^j, abort the rewinding and continue to the next repetition.
  - When $\mathcal{R}$ sends an end message for a valid instance $I' \neq I$ of $\mathcal{A}$ (i.e., a transcript τ such that $\mathcal{A}$'s next message would be a forgery (m^*, σ^*) for instance I'), abort the rewinding and continue to the next repetition. (If instead $\mathcal{A}$'s next message would be $\perp$ because $\mathsf{Valid}(\tau, I) = 0$, return $\perp$.)
  - If v' ever contains a message whose transcript contains a response k_I to any query for $(\mathsf{Open}, \pi_{\ell(n)})$ (i.e., (Open, π'_k)), then, if it is the case that $\mathsf{Ver}_{k_I}(m_{I,i,\pi_{\ell(n)}}, \sigma_{I,i,\pi_{\ell(n)}}) = \mathsf{Accept}$ for every $i \in [q(n)]$ (letting m and σ variables be defined as in the Valid predicate), store k_I and end the Rewind procedure (i.e., return to the outer execution); otherwise, if k_I is not a correct key, store nothing to k_I and continue to the next repetition.

Figure 4: Formal description of the meta-reduction $\mathcal{B}$ (2).

4.2 Analyzing the Meta-Reduction

The remaining part of the proof is devoted to analyzing the success probability $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$. This, as previously discussed, involves investigating the probability with which the meta-reduction $\mathcal{B}$ and the ideal adversary $\mathcal{R}^{\mathcal{A}}$ diverge while interacting with $\mathcal{C}$. We formalize this with the following claim:

Claim 4. If $(\mathcal{C}, t(\cdot))$ is a secure assumption and we can bound

$$\Pr[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \to \mathsf{Accept}] - \Pr[\langle \mathcal{B}, \mathcal{C}\rangle \to \mathsf{Accept}] \leq p(n)$$

then there is a negligible $\epsilon(\cdot)$ such that $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n) \leq p(n) + \epsilon(n)$.

Proof. Since $\mathcal{B}$ is efficient and $(\mathcal{C}, t(\cdot))$ is secure, there is a negligible $\epsilon(\cdot)$ such that $\Pr[\langle \mathcal{B}, \mathcal{C}\rangle \to \mathsf{Accept}] \leq t(n) + \epsilon(n)$.

So, given $\Pr[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \to \mathsf{Accept}] - \Pr[\langle \mathcal{B}, \mathcal{C}\rangle \to \mathsf{Accept}] \leq p(n)$, then we conclude that $\Pr[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \to \mathsf{Accept}] \leq t(n) + p(n) + \epsilon(n)$, and thus:

$$\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n) = \Pr[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \to \mathsf{Accept}] - t(n) \leq p(n) + \epsilon(n)$$

□

So it suffices to bound $\Pr[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle \to \mathsf{Accept}] - \Pr[\langle \mathcal{B}, \mathcal{C}\rangle \to \mathsf{Accept}]$. In order to do so, we begin by defining an *inefficient* "hybrid" meta-reduction $\mathcal{B}'$ which acts identically to $\mathcal{B}$, with the sole exception that, during the Rewind procedure, if $\mathcal{B}'$ encounters a response k_I to a query for $(\mathsf{Open}, \pi_{\ell(n)})$ (i.e., a key for the instance for which $\mathcal{B}'$ must produce a forgery), and if the recovered k_I is valid (i.e., $\mathsf{Ver}_{k_I}(m_{I,i,\pi_{\ell(n)}}, \sigma_{I,i,\pi_{\ell(n)}}) = \mathsf{Accept}$ for every $i \in [q(n)]$), then $\mathcal{B}'$ will first determine, using brute force, whether there are any other keys k' such that $\mathsf{Ver}_{k'}(m_{I,i,\pi_{\ell(n)}}, \sigma_{I,i,\pi_{\ell(n)}}) = \mathsf{Accept}$ for every $i \in [q(n)]$ but $\mathsf{Tag}_{k'}(m^*_I) \neq \mathsf{Tag}_{k_I}(m^*_I)$. If not (i.e., either k_I is the only such key or there is a unique correct forgery (m^*_I, σ^*_I)), then $\mathcal{B}'$ stores k_I, identically to $\mathcal{B}$; otherwise, $\mathcal{B}'$ stores the *lexicographically first* such key k' and uses that key instead of k_I to produce the forgery (identically to $\mathcal{A}$).

For ease of notation, let us further define some experiments and variables:

- Let $\mathsf{Real}(1^n)$ denote the experiment $[\mathcal{B} \leftrightarrow \mathcal{C}](1^n)$, and $\mathsf{Output}[\mathsf{Real}(1^n)]$ the output distribution $\langle \mathcal{B}, \mathcal{C}\rangle(1^n)$. Let $\mathsf{Hyb}(1^n)$ and $\mathsf{Output}[\mathsf{Hyb}(1^n)]$ be defined analogously for the "hybrid" experiment $[\mathcal{B}' \leftrightarrow \mathcal{C}](1^n)$, and lastly $\mathsf{Ideal}(1^n)$ and $\mathsf{Output}[\mathsf{Ideal}(1^n)]$ for the "ideal" experiment $[\mathcal{R}^{\mathcal{A}} \leftrightarrow \mathcal{C}](1^n)$.

- For any such experiment, let $\{\vec{m}_I, \pi_I\}$ define the randomness used to generate, respectively, all query variables ($m_{(\cdot)}$ or $\omega_{(\cdot)}$) and the permutation π for an instance I (real or simulated) of $\mathcal{A}$ (including the case where a query or permutation might be regenerated after, e.g., rewinding). Let $\mathcal{O}_{ext}$ denote all other randomness. Furthermore, let $M(n)$ be an upper bound to the number of instances of $\mathcal{A}$ started by $\mathcal{R}$.

- For instance, an experiment $\mathsf{Real}_{\{\vec{m}_I,\pi_I\}_{I \in [M(n)] \setminus J}, \mathcal{O}_{ext}}(1^n)$ (which we henceforth abbreviate as $\mathsf{Real}_{\{\vec{m}_I,\pi_I\}_{-J}, \mathcal{O}_{ext}}(1^n)$) would indicate the interaction between $\mathcal{B}$ and $\mathcal{C}$ with all randomness fixed *except* for the variables m and π for a particular instance J of $\mathcal{A}$ (simulated by $\mathcal{B}$).

- Naturally, an experiment denoted by, e.g., $\mathsf{Real}_{\{\vec{m}_I,\pi_I\}_{I \in [M(n)]}, \mathcal{O}_{ext}}(1^n)$, has all randomness fixed and hence is deterministic.

Let $\mathsf{Unique}(\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}, \mathcal{O}_{ext})$ be the "key-uniqueness" predicate on the randomness of Real (or Ideal) which is true if, during execution of the experiment $\mathsf{Real}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$, whenever $\mathcal{B}$ returns a forgery (m^*, σ^*), it is the case that $\sigma^* = \mathsf{Tag}_{k^*}(m^*)$, where k^* is the lexicographically first key k such that $\mathsf{Ver}_k(m_{I,i,\pi_{I,j}}, \sigma_{I,i,\pi_{I,j}}) = \mathsf{Accept}$ for all $i \in [q(n)]$. That is, Unique is true whenever, given the randomness of an experiment, $\mathcal{B}$ (if rewinding succeeds) returns the same forgery as $\mathcal{A}$ would in the Ideal experiment. The occurrence of Unique is hence fully determined by the randomness ($\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}$ and $\mathcal{O}_{ext}$) that fully determines the execution of Real or Ideal.

We must also deal with the fact that $\mathcal{R}$ may rewind $\mathcal{A}$. Let $W(n)$ be a polynomial upper bound to the number of times that $\mathcal{R}$ causes $\mathcal{A}$ to generate a permutation π (including by rewinding) in the experiment $\mathsf{Ideal}(1^n)$, and note that, trivially, $W(n) \geq M(n)$.

Now, with setup completed, we can proceed in two major steps. Our goal is to bound

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

which we can do by bounding

$$|\Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Accept}] - \Pr[\mathsf{Output}[\mathsf{Hyb}(1^n)] = \mathsf{Accept}]|$$

and

$$|\Pr[\mathsf{Output}[\mathsf{Hyb}(1^n)] = \mathsf{Accept}] - \Pr[\mathsf{Output}[\mathsf{Ideal}(1^n)] = \mathsf{Accept}]|$$

4.3 Comparing the Real and Hybrid Experiments

We begin with the first of these quantities, which is relatively straightforward to bound. Informally, whenever Unique holds (the probability of which is dictated by Lemma 2), $\mathcal{B}$ and $\mathcal{B}'$ behave identically by construction.

Claim 5. There exists negligible $\nu(\cdot)$ such that, for all $n \in \mathbb{N}$:

$$|\Pr[\mathsf{Output}[\mathsf{Real}(1^n)] = \mathsf{Accept}] - \Pr[\mathsf{Output}[\mathsf{Hyb}(1^n)] = \mathsf{Accept}]|$$

$$< \frac{2nW(n)}{q(n)} + \nu(n)$$

taken over the randomness of $\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}$ and $\mathcal{O}_{ext}$.

Proof. This follows fairly easily from the construction of $\mathcal{B}'$ and definition of Unique. Recall that $\mathsf{Unique}(\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}, \mathcal{O}_{ext})$ is true exactly when $\mathcal{B}$ would return the same forgery as $\mathcal{A}$ given successful rewinding; since $\mathcal{B}'$ by construction succeeds at rewinding whenever $\mathcal{B}$ does yet always returns the same forgery as $\mathcal{A}$, we can see that $\mathsf{Real}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$ is identically distributed to $\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$ whenever $\mathsf{Unique}(\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}, \mathcal{O}_{ext})$ is true.

By Lemma 2, we can see that, whenever $\mathcal{B}$ (or $\mathcal{B}'$) generates a permutation π and forgery input m^*, $\mathsf{Unique}(\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}, \mathcal{O}_{ext})$ has at most a $2n/q(n) + \nu'(n)$ chance (for negligible $\nu'(\cdot)$) to *not* hold for the respective forgery if it is eventually returned, since, by the lemma and the fact that the key retrieved by $\mathcal{B}$ must agree with the lexicographically first key k^* for every one of the $q(n)$ tag queries, it must (with probability $1 - \nu'(n)$) agree with k^* for at least a $1 - 2n/q(n)$ fraction of inputs m^*.

Since by assumption π (and m^*) are generated at most $W(n)$ times during the execution of Real or Hyb, the probability (over $\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}$ and $\mathcal{O}_{ext}$) that $\mathsf{Unique}(\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}, \mathcal{O}_{ext})$

fails, and hence the probability that Real and Hyb diverge, is, by the union bound (and letting $\nu(n) = W(n)\nu'(n)$), at most

$$\frac{2nW(n)}{q(n)} + \nu(n)$$

as desired.

□

4.4 Comparing the Hybrid and Ideal Experiments

To relate the hybrid $\mathcal{B}'$ to the "ideal" interaction with $\mathcal{R}^{\mathcal{A}}$, we next present the following claim, which informally holds because, by construction, $\mathcal{B}'$ behaves identically to $\mathcal{A}$ as long as rewinding does not fail (in which case it would return Fail).

Claim 6.

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

taken over the randomness of $\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}$ and $\mathcal{O}_{ext}$.

Proof. This follows immediately from the lemma below:

Lemma 3. For any assignment of $\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}$ and $\mathcal{O}_{ext}$, either:

- $\mathsf{Output}[\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)] = \mathsf{Fail}$, or
- $\mathsf{Ideal}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n) = \mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$, and thus $\mathsf{Output}[\mathsf{Ideal}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)] = \mathsf{Output}[\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)]$

Proof. First, observe that, given the same (fixed) queries $m_{(\cdot)}$ and $\omega_{(\cdot)}$ and permutation π for each instance, and since $\mathcal{C}$ is the same in each case, the only points at which the views of Real (and hence Hyb) and Ideal can possibly diverge is when $\mathcal{B}$ (or $\mathcal{B}'$ or $\mathcal{A}$) generates a forgery. This is by construction; given a particular setting of $\{\vec{m}_I, \pi_I\}_{I\in[M(n)]}$ and $\mathcal{O}_{ext}$, an instance I of $\mathcal{A}$ and its simulation in $\mathcal{B}'$ are guaranteed to behave identically up to the point where a forgery is produced (assuming no other instance diverges before that point), as the queries they make to $\mathcal{C}$ are identical by fixing $\{\vec{m}_I, \pi_I\}$ and $\mathcal{C}$'s responses are identical by fixing $\mathcal{O}_{ext}$.

So, consider any point at which $\mathcal{B}'$ generates a forgery in Hyb, and assume that rewinding was successful (i.e., $\mathcal{B}'$ does not output Fail) and that Ideal and Hyb have not diverged yet. If there is a unique correct forgery to return at this point, $\mathcal{B}'$ will by construction return it, as will $\mathcal{A}$. Conversely, if there are multiple possible forgeries, then $\mathcal{B}'$ will in fact *also* behave identically to $\mathcal{A}$ by brute-forcing the lexicographically first key consistent with all tag queries and using that to generate a forgery guess. Either way, if $\mathcal{B}'$ does not return Fail (i.e., $\mathsf{Output}[\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)] \neq \mathsf{Fail}$), then Ideal and Hyb will continue to behave identically, and thus, proceeding in this fashion over all forgeries produced, as long as $\mathcal{B}'$ never returns Fail we can conclude that

$$\mathsf{Ideal}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n) = \mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{I\in[M(n)]},\mathcal{O}_{ext}}(1^n)$$

as desired. □

□

4.5 Bounding the Hybrid's Failure Probability

So all that remains is to investigate the probability of Hyb outputting Fail; to do this we can make a critical observation about rewinding in the context of our construction. Formally, we prove the following:

Proposition 1. *There exists a negligible function $\epsilon(\cdot)$ such that, for all $n \in \mathbb{N}$, taken over the randomness of $\{\vec{m}_I, \pi_I\}_{I \in [M(n)]}$ and $\mathcal{O}_{ext}$:*

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

Proof. First, we show that without loss of generality $\mathcal{R}$ can never rewind *except* from a point after π is generated to a point before π is generated; intuitively, this is because all of $\mathcal{A}$'s queries to $\mathcal{R}$ are dependent only on (1) the permutation π and (2) the validity of $\mathcal{R}$'s responses, and as such any rewinding that does not result in π being regenerated can in fact be internally simulated by $\mathcal{R}$. Formally, we state the following claim:

Claim 7. *Given any $\mathcal{R}$ that rewinds any instance of $\mathcal{A}$ either (1) from a point before π is generated or (2) to a point after π is generated, there exists an $\mathcal{R}'$ with identical success probability that does not perform such rewinding.*

Proof. Given any rewinding of type (1), we notice that $\mathcal{A}$'s or $\mathcal{B}$'s tag queries are completely determined on initialization; hence, we can simply let $\mathcal{R}'$ completely skip the pre-rewinding messages to effectively "collapse" this rewinding into a single execution thread, as rewinding in this case will not change any of the queries made by $\mathcal{A}$.

Similarly, given any rewinding of type (2), we can observe that the order π of key-opening queries made by $\mathcal{A}$ (as well as the order of the instances q to which the tag queries are made, since this solely depends on π) is completely determined at the time that π is generated.

While the inputs ω to the tag queries and the final forgery output depend on the Valid predicate as well, one can additionally observe that, *as long as the* Valid *predicate holds*, these inputs and the forgery are also fully determined when π is generated, *even if* Unique *does not hold.* Specifically, this is because $\mathcal{A}$ and $\mathcal{B}'$ generate the forgery input m^* when π is generated, and because the key used to produce the forgery is uniquely determined then as well (by lexicographical ordering and by the fact that the responses to ω_i are never verified and hence cannot change the set of usable keys). Furthermore, the predicate Valid can be computed entirely by $\mathcal{R}$ given a particular transcript, since all information $(f, \pi, \vec{\omega})$ required to compute it is either public or given by $\mathcal{A}$ during the execution which produces the transcript.

Hence, $\mathcal{R}'$ can again collapse this latter type of rewinding by restricting to transcripts where Valid is true (as if Valid is false then it is guaranteed to receive $\perp$ instead of a forgery) and again skipping pre-rewinding messages (as they have no effect on $\mathcal{A}$'s queries and output given that Valid is true). $\square$

Hence, we assume without loss of generality that $\mathcal{R}$ sends at most $W(n)$ "end messages" (i.e., forgery requests) requiring rewinding, as π is by assumption generated no more than $W(n)$ times and the responses to any further end messages are effectively simulatable by $\mathcal{R}$. We disregard end messages sent for instances for which $\mathcal{R}$ has not answered all $\ell(n) - 1$ key opening queries, since, with all-but-negligible probability, $\mathcal{A}$ or $\mathcal{B}'$ can directly respond to these with $\perp$ (as Valid will evaluate to 0 unless $\mathcal{R}$ guesses a random and unknown ω_i correctly).

At this point, we have shown that our hybrid experiment gives us a setting with minimal rewinding and guaranteed key uniqueness, much like the setting discussed in [MP18] for the case of unique signatures. Hence, we can leverage this observation to prove the following claim, analogous to the key "rewinding lemma" therein. Consider the following for any possible execution $\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{-J},\vec{m}_J,\mathcal{O}_{ext}}(1^n)$ (i.e., for any fixed setting of all randomness aside from π_J), and notice that, since it applies to arbitrarily fixed randomness, it must thus apply over all possible randomness of the experiment $\mathsf{Hyb}(1^n)$:

Claim 8. Given any experiment $\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{-J},\vec{m}_J,\mathcal{O}_{ext}}(1^n)$, the probability, over the uniformly chosen permutation π_J, that the simulated instance J will return Fail when rewinding any end message, is, for all $n \in \mathbb{N}$, at most

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

The claim is nearly identical to its analogue in [MP18], but for completeness we provide a proof in Appendix C. We can conclude as desired that the probability of any forgery request causing $\mathcal{B}'$ to return Fail in the experiment Hyb is at most

$$W(n)\left(\frac{W(n) + r(n) + 1}{\ell(n)}\right) + \epsilon(n)$$

by combining Claim 8 (taken over all possible assignments of the fixed randomness) with the union bound over our bound of $W(n)$ possible (unique) forgery requests for which $\mathcal{R}$ has answered all key-opening queries. For any requests for which this is *not* the case, we know by Claim 3 that the probability of such requests causing $\mathcal{B}'$ to return anything besides $\perp$ is negligible, so, since $\mathcal{R}$ is polynomial-time, these requests add at most a negligible $\epsilon(n)$ to the probability of Hyb returning Fail.

□

4.6 Bounding the Security Loss

Finally, we must translate this bound on the failure probability of $\mathcal{B}'$ into a bound on the security loss of the reduction $\mathcal{R}$. This argument is quite similar to that presented in [MP18], albeit with the minor difference that the success probability of $\mathcal{A}$ is no longer 1 but instead very close to 1 for sufficiently large $q(n)$.

Take sufficiently large $q(n)$—say, $q(n) \geq 4n\ell(n)^2$. Recall that, by Claim 2, we know that, for sufficiently large n:

$$\mathsf{Success}_{\mathcal{A}}(n) \geq 1 - \frac{2n}{q(n)} - \nu(n) \geq 1 - \frac{1}{2\ell(n)^2}$$

By combining Claim 4, Claim 5, Claim 6, and Proposition 1, we can derive that, for negligible $\epsilon(\cdot), \nu(\cdot)$ and sufficiently large n:

$$\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n) \leq \Pr[\mathsf{Output}[\mathsf{Hyb}(1^n)] = \mathsf{Fail}] + \frac{2nW(n)}{q(n)} + \nu(n)$$

$$< W(n)\left(\frac{W(n)+r(n)+1}{\ell(n)}\right) + \frac{2nW(n)}{q(n)} + (\epsilon(n)+\nu(n))$$
$$< W(n)\left(\frac{W(n)+r(n)+2}{\ell(n)}\right)$$

To deal with running time, Lemma 5 (Appendix B) of [MP18] shows that we can substitute the number of times a specific computation is performed (in this case, the number of times π is generated for some instance of $\mathcal{A}$) for Time and receive a lower bound to the time-based security loss; effectively, this follows because we can rewrite $\mathcal{A}$ so that the computation dominates the running time (for both $\mathcal{A}$ and $\mathcal{R}$). Formally, if we let $\mathsf{Query}^{\pi}_{\mathcal{A}}(n)$ denote the number of times π is generated during $\mathcal{A}$, and respectively for $\mathcal{R}^{\mathcal{A}}$, this lemma gives:

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

By definition $\mathsf{Query}^{\pi}_{\mathcal{A}}(n) = 1$ and we may take $\mathsf{Query}^{\pi}_{\mathcal{R}^{\mathcal{A}}}(n) = W(n)$. We can separate our analysis into two cases to finish:

Case 1. If $W(n) \geq \sqrt{\ell(n)} - (r(n)+2)$, then:

$$\lambda_{\mathcal{R}}(n) \geq \frac{\mathsf{Success}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)} \frac{\mathsf{Query}^{\pi}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Query}^{\pi}_{\mathcal{A}}(n)} \geq \frac{1-\frac{1}{2\ell(n)^2}}{1}\frac{W(n)}{1}$$
$$= \left(1-\frac{1}{2\ell(n)^2}\right)W(n) \geq \left(1-\frac{1}{2\ell(n)^2}\right)(\sqrt{\ell(n)} - (r(n)+2))$$

Case 2. Otherwise, $W(n) < \sqrt{\ell(n)} - (r(n)+2)$ then $W(n) + r(n) + 2 < \sqrt{\ell(n)}$, and we have:

$$\lambda_{\mathcal{R}}(n) \geq \frac{\mathsf{Success}_{\mathcal{A}}(n)}{\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)} \frac{\mathsf{Query}^{\pi}_{\mathcal{R}^{\mathcal{A}}}(n)}{\mathsf{Query}^{\pi}_{\mathcal{A}}(n)} \geq \frac{1-\frac{1}{2\ell(n)^2}}{W(n)\left(\frac{W(n)+r(n)+2}{\ell(n)}\right)}\frac{W(n)}{1}$$
$$= \frac{\ell(n)\left(1-\frac{1}{2\ell(n)^2}\right)}{W(n)+r(n)+2} > \frac{\ell(n)\left(1-\frac{1}{2\ell(n)^2}\right)}{\sqrt{\ell(n)}} = \left(1-\frac{1}{2\ell(n)^2}\right)\sqrt{\ell(n)}$$

Either way, we conclude that, if $(\mathcal{C}, t(\cdot))$ is secure (and so our bound on the probability $\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$ holds), then:

$$\lambda_{\mathcal{R}}(n) \geq \left(1-\frac{1}{2\ell(n)^2}\right)(\sqrt{\ell(n)} - (r(n)+2))$$

which finishes the proof of Lemma 1.

References

[AGO11] Masayuki Abe, Jens Groth, and Miyako Ohkubo. Separating short structure-preserving signatures from non-interactive assumptions. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 628–646. Springer, Heidelberg, December 2011.

[ACD^{+}19] Ittai Abraham, T.-H. Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. Communication complexity of byzantine agreement, revisited. In Peter Robinson and Faith Ellen, editors, *38th ACM PODC*, pages 317–326. ACM, July / August 2019.

[BHJ^{+}15] Christoph Bader, Dennis Hofheinz, Tibor Jager, Eike Kiltz, and Yong Li. Tightly-secure authenticated key exchange. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part I*, volume 9014 of *LNCS*, pages 629–658. Springer, Heidelberg, March 2015.

[BJLS16] Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 273–304. Springer, Heidelberg, May 2016.

[BBF13] Paul Baecher, Christina Brzuska, and Marc Fischlin. Notions of black-box reductions, revisited. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 296–315. Springer, Heidelberg, December 2013.

[Bar16] Elaine Barker. Guideline for Using Cryptographic Standards in the Federal Government: Cryptographic Mechanisms. *NIST Special Publication 800-175B*, 2016.

[BPS00] Olivier Baudron, David Pointcheval, and Jacques Stern. Extended notions of security for multicast public key cryptosystems. In Ugo Montanari, José D. P. Rolim, and Emo Welzl, editors, *ICALP 2000*, volume 1853 of *LNCS*, pages 499–511. Springer, Heidelberg, July 2000.

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, Heidelberg, May 2000.

[BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *37th FOCS*, pages 514–523. IEEE Computer Society Press, October 1996.

[BRT12] Mihir Bellare, Thomas Ristenpart, and Stefano Tessaro. Multi-instance security and its application to password-based cryptography. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 312–329. Springer, Heidelberg, August 2012.

[BFW16] David Bernhard, Marc Fischlin, and Bogdan Warinschi. On the hardness of proving CCA-security of signed ElGamal. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 47–69. Springer, Heidelberg, March 2016.

[BV98] Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 59–71. Springer, Heidelberg, May / June 1998.

[BHT18] Priyanka Bose, Viet Tung Hoang, and Stefano Tessaro. Revisiting AES-GCM-SIV: Multi-user security, faster key derivation, and better bounds. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 468–499. Springer, Heidelberg, April / May 2018.

[BMV08] Emmanuel Bresson, Jean Monnerat, and Damien Vergnaud. Separation results on the "one-more" computational problems. In Tal Malkin, editor, *CT-RSA 2008*, volume 4964 of *LNCS*, pages 71–87. Springer, Heidelberg, April 2008.

[CMS12] Sanjit Chatterjee, Alfred Menezes, and Palash Sarkar. Another look at tightness. In Ali Miri and Serge Vaudenay, editors, *SAC 2011*, volume 7118 of *LNCS*, pages 293–319. Springer, Heidelberg, August 2012.

[Cor02] Jean-Sébastien Coron. Optimal security proofs for PSS and other signature schemes. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 272–287. Springer, Heidelberg, April / May 2002.

[Dam92] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 445–456. Springer, Heidelberg, August 1992.

[DA99] Tim Dierks and Christopher Allen. The TLS Protocol Version 1.0. RFC 2246, January 1999.

[DR06] Tim Dierks and Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346, April 2006.

[DR08] Tim Dierks and Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246, August 2008.

[DH76] Witfield Diffie and Martin Hellman. New Directions in Cryptography. *IEEE Trans. Inf. Theor.*, 22(6):644–654, November 1976.

[Dwo07] Morris Dworkin. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. *NIST Special Publication 800-38D*, 2007.

[FS10] Marc Fischlin and Dominique Schröder. On the impossibility of three-move blind signature schemes. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 197–215. Springer, Heidelberg, May / June 2010.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 99–108. ACM Press, June 2011.

[Gol09] Oded Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications.* Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *17th ACM STOC*, pages 291–304. ACM Press, May 1985.

[HRS09] Iftach Haitner, Alon Rosen, and Ronen Shaltiel. On the (im)possibility of Arthur-Merlin witness hiding protocols. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 220–237. Springer, Heidelberg, March 2009.

[HT16] Viet Tung Hoang and Stefano Tessaro. Key-alternating ciphers and key-length extension: Exact bounds and multi-user security. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 3–32. Springer, Heidelberg, August 2016.

[HTT18] Viet Tung Hoang, Stefano Tessaro, and Aishwarya Thiruvengadam. The multi-user security of GCM, revisited: Tight bounds for nonce randomization. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 1429–1440. ACM Press, October 2018.

[HJK12] Dennis Hofheinz, Tibor Jager, and Edward Knapp. Waters signatures with optimal security reduction. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 66–83. Springer, Heidelberg, May 2012.

[JSSOW17] Tibor Jager, Martijn Stam, Ryan Stanley-Oakes, and Bogdan Warinschi. Multi-key authenticated encryption with corruptions: Reductions are lossy. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 409–441. Springer, Heidelberg, November 2017.

[KK12] Saqib A. Kakvi and Eike Kiltz. Optimal security proofs for full domain hash, revisited. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 537–553. Springer, Heidelberg, April 2012.

[Lub96] Michael Luby. *Pseudorandomness and Cryptographic Applications.* Princeton University Press, Jan. 1996.

[LMP17] Atul Luykx, Bart Mennink, and Kenneth G. Paterson. Analyzing multi-key security degradation. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 575–605. Springer, Heidelberg, December 2017.

[MVO96] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot. *Handbook of Applied Cryptography.* CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 1996.

[MP18] Andrew Morgan and Rafael Pass. On the security loss of unique signatures. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 507–536. Springer, Heidelberg, November 2018.

[ML15] Nicky Mouha and Atul Luykx. Multi-key security: The Even-Mansour construction revisited. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 209–223. Springer, Heidelberg, August 2015.

[Nao03] Moni Naor. On cryptographic assumptions and challenges (invited talk). In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 96–109. Springer, Heidelberg, August 2003.

[Pas11] Rafael Pass. Limits of provable security from standard assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 109–118. ACM Press, June 2011.

[Tes15] Stefano Tessaro. Optimally secure block ciphers from ideal primitives. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 437–462. Springer, Heidelberg, November / December 2015.

Appendix

A Digital Signatures

Though we focus on MACs, we note that our main result (Theorem 2) also applies to reductions from the adaptive multi-user unforgeability of deterministic *digital signature schemes*. Intuitively, this holds because digital signature schemes are the public-key analogue of a MAC and, as such, every deterministic digital signature scheme implies a deterministic MAC by a tight reduction. Hence, if there were a linear-preserving reduction from a deterministic digital signature to some standard assumption, this would imply a linear-preserving reduction from a deterministic MAC to the same assumption, contradicting our main result.

A.1 Definitions

To formalize this, let us first recall the definition of a digital signature scheme; the key difference from a MAC is that verification is now performed with a *public* key, rather than a secret key.

Definition 7. We refer to a tuple of efficient (poly(n)-time) algorithms $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, where:

- $\mathsf{Gen}(1^n) \to (pk, sk)$ takes as input a security parameter n and outputs a public key and secret key $pk, sk \in \{0,1\}^n$,
- $\mathsf{Sign}_{sk}(m) \to \sigma$ takes as input a secret key sk and a message m from some message space $\mathcal{M}_n$ of size super-polynomial in n, and outputs a signature σ for the message, and
- $\mathsf{Ver}_{pk}(m, \sigma) \to \{\mathsf{Accept}, \mathsf{Reject}\}$ takes as input a public key pk, a message m, and a signature σ, and outputs Accept or Reject denoting whether the tag σ is valid for the message m, specifically in such a manner that $\Pr[(pk, sk) \leftarrow \mathsf{Gen}(1^n); \mathsf{Ver}_{pk}(m, \mathsf{Sign}_{sk}(m)) \to \mathsf{Accept}] = 1$ for any valid message $m \in \mathcal{M}_n$,

as a **digital signature scheme**. If, in addition, $\mathsf{Sign}_{sk}(m)$ is a deterministic function, then we refer to Π as a **deterministic digital signature scheme**.

As with MACs, we focus on digital signatures with input and output space superpolynomial in the key size n. Note that, for determinism, we do not require $\mathsf{Ver}_{pk}(m, \sigma)$ to accept *only* if $\mathsf{Sign}_{sk}(m) = \sigma$—this is an additional property that, together with determinism, comprises *uniqueness.* This distinction is not meaningful for MACs; since verification requires the secret key, it is always possible to check whether $\mathsf{Tag}_{sk}(m) = \sigma$. Next, multi-user security for digital signatures can be formulated similarly to the analogous definition for MACs:

Definition 8. A digital signature scheme $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ is an $\ell(n)$-**key unforgeable signature scheme under adaptive corruption** (or **adaptively** $\ell(n)$-**key unforgeable**) if, for any interactive oracle-aided non-uniform probabilistic polynomial-time algorithm $\mathcal{A}$, there is a negligible function $\epsilon(\cdot)$ such that, for all $n \in \mathbb{N}$,

$$\Pr\left[\langle \mathcal{A}, \mathcal{C}_{\Pi}^{\ell(n)} \rangle (1^n) = \mathsf{Accept}\right] \leq \epsilon(n)$$

where $\mathcal{C}_{\Pi}^{\ell(n)}$ is the interactive challenger that does as follows on input 1^n:

- Let $((pk_1, sk_1), \ldots, (pk_{\ell(n)}, sk_{\ell(n)})) \leftarrow \mathsf{Gen}(1^n)$. Initialize the transcript $\tau = (pk_1, \ldots, pk_{\ell(n)})$ and send it.
- Upon receiving a *signing* query (Query, i, m) for $i \in [\ell(n)]$, append $((\mathsf{Query}, i, m), \mathsf{Sign}_{sk_i}(m))$ to τ and send τ.
- Upon receiving a *key-opening* query (Open, i) for $i \in [\ell(n)]$, append the tuple $((\mathsf{Open}, i), sk_i)$ to τ and send τ.
- Upon receiving a forgery (m^*, σ^*, i^*) from $\mathcal{A}$, output Reject if one of the following three conditions is true:
  - τ contains a key opening query (Open, i^*).
  - τ contains an oracle query $(\mathsf{Query}, i^*, m^*)$.
  - $\mathsf{Ver}_{pk_{i^*}}(m^*, \sigma^*) \rightarrow \mathsf{Reject}$.
- Otherwise, output Accept.

We call a digital signature scheme Π **adaptively multi-key unforgeable** if it is adaptively $\ell(n)$-key unforgeable for every polynomial $\ell(\cdot)$.

A.2 Results for Digital Signatures

Perhaps unsurprisingly, every adaptively multi-key unforgeable deterministic digital signature implies an adaptively multi-key unforgeable deterministic MAC, as follows:

Claim 9. For any deterministic digital signature scheme Π, there exists a deterministic MAC Π' for which there exists a tight reduction (with security loss 1) from adaptive multi-key unforgeability of Π' to adaptive multi-key unforgeability of Π.

Proof. Given $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, let $\Pi' = (\mathsf{Gen}', \mathsf{Tag}', \mathsf{Ver}')$ be defined on the same message space and given by:

- $\mathsf{Gen}'(1^n)$ takes $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$ and returns sk.
- $\mathsf{Tag}'_{sk}(m)$ returns $\mathsf{Sign}_{sk}(m)$.

- $\mathsf{Ver}'_{sk}(m, \sigma)$ returns Accept if $\mathsf{Sign}_{sk}(m) = \sigma$ and Reject otherwise.

Correctness and determinism of Π' follow directly from the definition of Tag' and Ver' and determinism of Π. As for the reduction, assume an adversary $\mathcal{A}'$ that breaks adaptive $\ell(n)$-key unforgeability of the MAC Π'. We can construct an adversary $\mathcal{A}$ to break adaptive $\ell(n)$-key unforgeability of the digital signature scheme Π by essentially letting the adversary $\mathcal{A}'$ interact directly with its challenger. When $\mathcal{A}'$ makes either a tag or a corruption query, $\mathcal{A}$ forwards the respective query to its challenger, and returns the challenger's response to $\mathcal{A}'$. Finally, $\mathcal{A}'$ returns a forgery (m^*, σ^*, i^*); if $\mathcal{A}'$ is successful, then, by the properties of the security game for Π', instance i^* will be uncorrupted and $\mathsf{Tag}'_{sk_{i^*}}(m^*)$ will not have been made as a tag query, and the respective properties (for Sign rather than Tag) will hold in the game for Π as well because communication is simply forwarded. Furthermore, in this case, it will be true that $\mathsf{Sign}_{sk_{i^*}}(m^*) = \sigma^*$ (i.e., Ver' will accept), meaning that, by correctness of Π, it must be true that $\mathsf{Ver}_{sk_{i^*}}(m^*, \sigma^*) = \mathsf{Accept}$. Hence, $\mathcal{A}$ can forward the forgery to its own challenger, and it will necessarily be correct when $\mathcal{A}'$ is successful.

The above reduction merely forwards communication, so it inherits the same running time as the adversary it uses (i.e., $\mathsf{Time}_{A'}(n) = \mathsf{Time}_{\mathcal{A}}(n)$); furthermore, since $\mathcal{A}$ succeeds whenever $\mathcal{A}'$ does, we have $\mathsf{Success}_{\mathcal{A}}(n) \geq \mathsf{Success}_{\mathcal{A}'}(n)$, and so the above reduction has security loss at most 1. □

From this, we can conclude the following:

Claim 10. Let Π be a deterministic digital signature scheme. If there exists some fixed-parameter black-box reduction $\mathcal{R}$ for basing adaptive multi-key unforgeability of Π on some $r(\cdot)$-round intractability assumption $(\mathcal{C}, t(\cdot))$ (for polynomial $r(\cdot)$), then there exists a deterministic MAC Π' and a fixed-parameter black-box reduction $\mathcal{R}'$ for basing adaptive multi-key unforgeability of Π' on $(\mathcal{C}, t(\cdot))$. In addition, $\mathcal{R}'$ has security loss no greater than $\mathcal{R}$.

Proof. Let Π' be as defined in Claim 9, and let the reduction $\mathcal{R}^*$ be the reduction outlined in the same claim. Then, for any adversary $\mathcal{A}$ breaking adaptive $\ell(n)$-key unforgeability of Π', let $\mathcal{R}'^{\mathcal{A}}$ be given by $\mathcal{R}^{\mathcal{R}^{*\mathcal{A}}}$, which will break $(\mathcal{C}, t(\cdot))$ since $\mathcal{R}^{*\mathcal{A}}$ is an adversary against adaptive $\ell(n)$-key unforgeability of Π and $\mathcal{R}$ reduces adaptive $\ell(n)$-key unforgeability of Π to $(\mathcal{C}, t(\cdot))$. Furthermore, because $\mathcal{R}^*$ preserves the running time and success probability of $\mathcal{A}$, we have that, for any $\mathcal{A}$, $\mathsf{Time}_{\mathcal{R}'^{\mathcal{A}}}(n) \leq \mathsf{Time}_{\mathcal{R}^{\mathcal{A}}}(n)$ and $\mathsf{Success}_{\mathcal{R}'^{\mathcal{A}}}(n) \geq \mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$, which means that $\lambda_{\mathcal{R}'}(n) \leq \lambda_{\mathcal{R}}(n)$. □

Naturally, since $\mathcal{R}$ being linear-preserving will imply that $\mathcal{R}'$ is also linear-preserving, we can combine this claim with Theorem 2 to immediately obtain the analogous impossibility for digital signatures:

Theorem 3. Let Π be a deterministic digital signature scheme. If there exists some fixed-parameter black-box reduction $\mathcal{R}$ for basing adaptive multi-key unforgeability of Π on some $r(\cdot)$-round intractability assumption $(\mathcal{C}, t(\cdot))$ (for polynomial $r(\cdot)$), then either:

(1) $\mathcal{R}$ is not a linear-preserving reduction, or

(2) there exists a polynomial-time adversary $\mathcal{B}$ breaking the assumption $(\mathcal{C}, t(\cdot))$.

B Pseudorandom Functions

Additionally, we can apply our main result directly to rule out linear-preserving reductions from *pseudorandomness* of a family of functions. Given the appropriate definition of adaptive multi-user security, this will follow from the fact that, even in an adaptive multi-user context, every PRF can be shown to be unpredictable—that is, a deterministic MAC—via a tight reduction. Hence, a linear-preserving reduction from adaptive multi-user pseudorandomness implies a linear-preserving reduction from an adaptive multi-user unforgeable deterministic MAC, just as with signatures above.

B.1 Definitions

First, we provide the definition of adaptive multi-key pseudorandomness, which is in fact quite similar to "security under selective key-opening" as presented in [ACD+19][5]:

Definition 9. A family of functions $\mathcal{U} = \{f_k : \mathcal{X}_n \to \mathcal{Y}_n\}_{k \in \{0,1\}^n, n \in \mathbb{N}}$ (for some input and output spaces $\mathcal{X}_n$ and $\mathcal{Y}_n$) is a family of **adaptively secure $\ell(n)$-key pseudorandom functions** if f_k is computable in polynomial time, and, for any interactive oracle-aided non-uniform probabilistic polynomial-time algorithm $\mathcal{A}$, there is a negligible function $\epsilon(\cdot)$ such that, for all $n \in \mathbb{N}$,

$$\Pr\left[\langle \mathcal{A}, \mathcal{C}_{\mathcal{U}}^{\ell(n)} \rangle (1^n) = \mathsf{Accept}\right] \leq 1/2 + \epsilon(n)$$

where $\mathcal{C}_{\mathcal{U}}^{\ell(n)}$ is the interactive challenger that does as follows:

- Let $(k_1, \ldots, k_{\ell(n)}) \leftarrow (\{0,1\}^n)^{\ell(n)}$. Pick $b \leftarrow \{0,1\}$. Initialize empty transcript τ.
- Upon receiving a *function* query (Query, i, x) for $i \in [\ell(n)]$ and $x \in \mathcal{X}_n$, append the tuple $((\mathsf{Query}, i, x), f_{k_i}(x))$ to τ and send τ.
- Upon receiving a *key-opening* query (Open, i) for $i \in [\ell(n)]$, append the tuple $((\mathsf{Open}, i), k_i)$ to τ and send τ.
- Upon receiving a challenge (x^*, i^*) from $\mathcal{A}$, let y^* be $f_{k_{i^*}}(x^*)$ if $b = 0$, or $y^* \leftarrow \{0,1\}^n$ if $b = 1$. Append $((x^*, i^*), y^*)$ to τ and send τ.
- On receiving an output b' from $\mathcal{A}$, output Reject if one of the following three conditions is true:
  - τ contains a key opening query (Open, i^*).
  - τ contains an oracle query $(\mathsf{Query}, i^*, x^*)$.
  - $b' \neq b$.
- Otherwise, output Accept.

We call a family of functions $\mathcal{U}$ a family of **adaptively secure pseudorandom functions** if it is a family of **adaptively secure $\ell(n)$-key pseudorandom functions** for any polynomial $\ell(\cdot)$.

[5] In fact, our definition is slightly weaker, as it does not allow the adversary to adaptively *create* instances of the PRF; however, as we are proving a lower bound, this makes our result stronger.

B.2 Results for Pseudorandomness

Now we can formalize the fact that every pseudorandom function family implies a secure deterministic MAC—this is true even in our multi-key setting using our definitions of adaptive multi-key pseudorandomness and unforgeability. We present the following relatively straightforward claim:

Claim 11. For any efficiently computable function family $\mathcal{U} = \{f_k : \mathcal{X}_n \to \mathcal{Y}_n\}_{k \in \{0,1\}^n, n \in \mathbb{N}}$ such that $\mathcal{X}_n$ and $\mathcal{Y}_n$ have size super-polynomial in n, there exists a deterministic MAC Π' for which there is a tight reduction from adaptive $\ell(n)$-key unforgeability of Π' to $\ell(n)$-key pseudorandomness of $\mathcal{U}$, and additionally this tight reduction's security loss is bounded above by 6 for sufficiently large n.

Proof. Given $\mathcal{U} = \{f_k : \mathcal{X}_n \to \mathcal{Y}_n\}_{k \in \{0,1\}^n, n \in \mathbb{N}}$, let $\Pi' = (\mathsf{Gen}', \mathsf{Tag}', \mathsf{Ver}')$ be defined over message space $\mathcal{X}_n$ (which, in particular, gives us a sufficiently large message space to apply Theorem 2), and given by:

- $\mathsf{Gen}'(1^n)$ takes $sk \leftarrow \{0,1\}^n$ and returns sk.
- $\mathsf{Tag}'_{sk}(m)$ returns $f_{sk}(m)$.
- $\mathsf{Ver}'_{sk}(m, \sigma)$ returns Accept if $f_{sk}(m) = \sigma$ and Reject otherwise.

Consider an adaptive multi-key pseudorandomness adversary $\mathcal{A}$ against $\mathcal{U}$ that, given a respective multi-key unforgeability adversary $\mathcal{A}'$ (with the same number of instances $\ell(n)$) against Π, forwards all tag queries and all key-opening queries made by $\mathcal{A}'$ to its own oracle and returns the responses, receives a forgery (m^*, σ^*, i^*), queries its own oracle with a challenge on (x^*, i^*), and returns 0 if the result is equal to y^* and 1 otherwise.

If $\mathcal{A}'$ breaks unforgeability with non-negligible probability $p(n)$, then, if $b = 0$, $\mathcal{A}$ will return 0 with at least probability $p(n)$ (since the forgery from $\mathcal{A}'$ will be valid and match with the output of the final query in that case); otherwise, if $b = 1$, then $\mathcal{A}$ will return 1 with probability $1 - 1/|\mathcal{Y}_n|$—that is, the probability that the forgery from $\mathcal{A}'$ coincidentally matches the random output of the challenge.

Thus, if $\mathcal{A}'$ breaks unforgeability with probability $p(n)$, $\mathcal{A}$ succeeds with at least probability $1/2(1 + p(n) - 1/|\mathcal{Y}_n|)$. Hence (recalling that $\mathsf{Success}_{\mathcal{A}}(n)$ is defined by the probability of success *minus* the assumption's threshold, which in this case is 1/2):

$$\mathsf{Success}_{\mathcal{A}}(n) = p(n)/2 - 1/2|\mathcal{Y}_n| \geq p(n)/3 = 1/3(\mathsf{Success}_{\mathcal{A}'}(n))$$

for sufficiently large n, because of our assumption that $\mathcal{Y}_n$ has superpolynomial size (and so $1/|\mathcal{Y}_n|$ is negligible). Furthermore, $\mathsf{Time}_{\mathcal{A}}(n) \leq 2\mathsf{Time}_{\mathcal{A}'}(n)$ (since $\mathcal{A}$ makes only a single additional query), demonstrating that the reduction is tight and has security loss bounded above by 6.[6] □

Because of the above reduction, we can show that Theorem 2 applies immediately to reductions from adaptive multi-key pseudorandomness of a function family to bounded-round assumptions:

[6] In fact, the actual security loss is bounded above by $2 + \epsilon$ for any constant ϵ, but, as we only require a constant bound for the subsequent arguments, we simplify the analysis here.

Claim 12. For any efficiently computable function family $\mathcal{U} = \{f_k : \mathcal{X}_n \to \mathcal{Y}_n\}_{k\in\{0,1\}^n, n\in\mathbb{N}}$ such that $\mathcal{X}_n$ and $\mathcal{Y}_n$ have size super-polynomial in n, if there exists some fixed-parameter black-box reduction $\mathcal{R}$ for basing adaptive multi-key pseudorandomness of $\mathcal{U}$ on some $r(\cdot)$-round intractability assumption $(\mathcal{C}, t(\cdot))$ (for polynomial $r(\cdot)$), then there exists a deterministic MAC Π' and a fixed-parameter black-box reduction $\mathcal{R}'$ for basing adaptive multi-key unforgeability of Π' on $(\mathcal{C}, t(\cdot))$. In addition, $\mathcal{R}'$ has security loss no greater than 6 times that of $\mathcal{R}$ for sufficiently large n.

Proof. Let Π' be as defined in Claim 11, and let $\mathcal{R}^*$ be the tight reduction from adaptive $\ell(n)$-key unforgeability of Π' to adaptive $\ell(n)$-key pseudorandomness of $\mathcal{U}$ also presented in Claim 11. Clearly, if $\mathcal{A}$ breaks adaptive $\ell(n)$-key unpredictability, then $\mathcal{R}^{*\mathcal{A}}$ will break adaptive $\ell(n)$-key pseudorandomness with at most a constant-factor loss in time and success probability.

Hence, to define the reduction $\mathcal{R}'$, given an adversary $\mathcal{A}$ against adaptive $\ell(n)$-key unforgeability of Π', we can simply let $\mathcal{R}'^{\mathcal{A}} = \mathcal{R}^{\mathcal{R}^{*\mathcal{A}}}$—since $\mathcal{R}$ breaks $(\mathcal{C}, t(\cdot))$ given an adaptive $\ell(n)$-key pseudorandomness adversary against $\mathcal{U}$, and $\mathcal{R}^{*\mathcal{A}}$ is an adaptive $\ell(n)$-key pseudorandomness adversary against $\mathcal{U}$ when $\mathcal{A}$ is an adaptive $\ell(n)$-key unforgeability adversary against Π', then $\mathcal{R}'^{\mathcal{A}}$ will break $(\mathcal{C}, t(\cdot))$. Also, our observations about the reduction $\mathcal{R}^*$ imply that, for any $\mathcal{A}$ and sufficiently large n, $\mathsf{Time}_{\mathcal{R}'^{\mathcal{A}}}(n) \le 2\mathsf{Time}_{\mathcal{R}^{\mathcal{A}}}(n)$ and $\mathsf{Success}_{\mathcal{R}'^{\mathcal{A}}}(n) \ge 1/3\mathsf{Success}_{\mathcal{R}^{\mathcal{A}}}(n)$, so $\lambda_{\mathcal{R}'}(n) \le 6\lambda_{\mathcal{R}}(n)$. □

Notice that, if $\mathcal{R}$ is linear-preserving, then $\mathcal{R}'$ must be as well, since its security loss is within a constant factor of $\mathcal{R}$'s. Hence, combining the above claim with Theorem 2 gives us:

Theorem 4. Let $\mathcal{U} = \{f_k : \{0,1\}^n \to \{0,1\}^n\}_{k\in\{0,1\}^n}$ be a family of efficiently computable functions. If there exists some fixed-parameter black-box reduction $\mathcal{R}$ for basing pseudorandomness of $\mathcal{U}$ on a $r(\cdot)$-round intractability assumption $(\mathcal{C}, t(\cdot))$ (for polynomial $r(\cdot)$), then either:

(1) $\mathcal{R}$ is not a linear-preserving reduction, or

(2) there exists a polynomial-time adversary $\mathcal{B}$ breaking the assumption $(\mathcal{C}, t(\cdot))$.

C Proof of Claim 8

Here we present the central "rewinding argument" used to prove our failure probability bound, which, aside from minor adaptations and slight reorganization for clarity, is reproduced from [MP18].

Proof. First, let us assume that other simulated instances (besides J) of $\mathcal{A}$ in the experiment $\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{-J},\vec{m}_J,\mathcal{O}_{ext}}(1^n)$ will never return Fail (i.e., that they will "magically" produce a correct forgery instead of returning Fail). This can only increase the probability that instance J will return Fail, as it guarantees that the experiment never aborts early.

Next, consider the permutation $\pi_J = (\pi_J^1, \ldots, \pi_J^{\ell(n)})$ in instance J; note that by the definition of $\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{-J},\vec{m}_J,\mathcal{O}_{ext}}(1^n)$ the execution is fully determined by π_J. Let us also define for $i \in [\ell(n)-1]$ the "rewound" sequence

$$\rho(\pi_J, i) \triangleq (\pi_J^1, \ldots, \pi_J^{i-1}, \pi_J^{\ell(n)})$$

corresponding to the rewound execution where the key-opening query in the i^{th} position is replaced by $\pi_J^{\ell(n)}$ to attempt to extract a correct key for that instance.

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

- $E_1(\rho(\pi_J, i))$: $\mathcal{R}$ fails to return a valid key (i.e., one consistent with all tag queries $m_{(\cdot)}$) for instance $\pi_{\ell(n)}$ in the respective rewinding, including the case where $\mathcal{R}$ attempts to rewind $\mathcal{A}$ before responding.
- $E_2(\rho(\pi_J, i))$: During the respective rewinding, $\mathcal{R}$ asks for a forgery for another instance $I \neq J$ (for which $\mathcal{B}'$ would be required to rewind) before returning a key for instance $\pi_J^{\ell(n)}$, *or* $\mathcal{R}$ requires external communication with $\mathcal{C}$ (which cannot be rewound) before returning a response.

In addition, for instance J to fail, the non-rewound execution of the instance must succeed (and finish executing), in that the event $E_1(\pi_{J,\leq i})$ (where $\mathcal{R}$ fails to return a valid tag or rewinds) *cannot* occur for any prefix $\pi_{J,\leq i} = (\pi_J^1, \ldots, \pi_J^i)$, where $i \in [\ell(n) - 1]$.

Since, as we have noted, the behavior of J during the hybrid execution $\mathsf{Hyb}_{\{\vec{m}_I,\pi_I\}_{-J},\vec{m}_J,\mathcal{O}_{ext}}(1^n)$ is fully determined by π_J for any fixed $\vec{m}_J$, every sequence π_J will deterministically either result in instance J returning something (either a forgery or $\perp$) or aborting and returning Fail; we shall refer to the former type of sequence (where J 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 lemma, which we note is proven implicitly in [MP18]; however, the complete proof is provided for the interested reader in Appendix C.1. Notationally, let $\leq$ applied to two sequences denote the prefix operator—that is, $s \leq t$ if $s_i = t_i$ for all $i \in |s|$. Then:

Lemma 4. Let S be a set of permutations of $[\ell]$. Let $T = \{s'||n : s||n \in S, s' < s\}$ be the set of "rewound" prefixes including, for any permutation $p \in S$ and integer $i < \ell$, p's first i elements concatenated with its last element. For any $s \in S$, let $P_s = \{p : p \leq s\}$ be the set of prefixes of elements in S. If there exists integer B such that, for all $s \in S$, $|P_s \cap T| \leq B$ (i.e., no $s \in S$ can share more than B prefixes with T), then the size of S is bounded by $|S| \leq (B + 1)(\ell - 1)!$

The relevance of this lemma stems intuitively from the fact that, if there exists a "bad" permutation $\pi_J = (\pi_J^1, \ldots, \pi_J^{\ell(n)})$, then any sequence aside from π_J itself which begins with an element of the set $\{s'||\pi_J^{\ell(n)} : s' \leq \pi_J\}$ will, by the fact that π_J is bad, fail to extract a valid key as a response to $(\mathsf{Open}, \pi_J^{\ell(n)})$ during the rewinding. Of course, this can occur due to either of the events E_1 or E_2 described above; we would like to conclude that these sequences are themselves good, but we can only conclude that a sequence is good if E_1 specifically occurs.

This is where the lemma helps us; letting T be the set of all of the prefixes guaranteed by some bad sequence in S to fail to extract a key in the rewinding, we know that a sequence containing strictly more than $W(n) + r(n)$ of these prefixes (namely, $W(n)$ at most which might contain end messages for other instances because of our earlier assumptions about rewinding, and $r(n)$ which might contain external communication) must be a good sequence, since at that point one of the prefixes is guaranteed to fail due to E_1 and not E_2.

So, applying Lemma 4 with $\ell = \ell(n)$, $B = W(n)+r(n)$, and S being the set of bad permutations, we deduce that S may have size at most

$$(W(n) + r(n) + 1)\,(\ell(n) - 1)!$$

and so at most an

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

fraction of all permutations of $[\ell(n)]$ can be bad (for any setting of the randomness outside of the permutation π_J), and the remainder must be good.

Hence, the chance that a permutation chosen at random is bad, which by definition is equal to the probability that a randomly chosen permutation $\pi_J = (\pi_1, \ldots, \pi_{\ell(n)})$ will result in instance J returning Fail, can be at most the fraction of permutations which are bad, or:

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

As this holds for arbitrary $\vec{m}_J$, it likewise holds over *all* $\vec{m}_J$ and thus over all randomness of the execution $\mathsf{Hyb}_{\{\vec{m}_I, \pi_I\}_{-J}, \vec{m}_J, \mathcal{O}_{ext}}(1^n)$. $\square$

C.1 Proof of Lemma 4

Proof. The proof, which is aside from notation identical to its counterpart in [MP18], begins with two crucial claims:

Claim 13. No permutation π of $[\ell]$ is such that $P_\pi \cap T \geq B + 2$.

Proof. Assume for the sake of contradiction that there exists some such π—i.e., that there are $B+2$ 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 $(\pi_1^j, \ldots, \pi_{i_j - 1}^j, \pi_\ell^j)$ 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 $\pi^{B+2} = (\pi_1^{B+2}, \ldots, \pi_\ell^{B+2})$; we shall show that $P_{\pi^{B+2}} \cap T \geq B + 1$ and thus that $\pi^{B+2} \notin S$, a contradiction because $\pi^{B+2} \in S$ by assumption.

We know that, since by assumption the first i_{B+2} elements of π are equivalent to

$$(\pi_1^{B+2}, \ldots, \pi_{i_{B+2}-1}^{B+2}, \pi_*^{B+2})$$

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 $(\pi_1^j, \ldots, \pi_{i_j-1}^j, \pi_\ell^j)$ is identical to π in the first i_j elements, which, since by assumption $i_j \leq i_{B+2} - 1$, also indicates that $(\pi_1^j, \ldots, \pi_{i_j-1}^j, \pi_\ell^j)$ is identical to π^{B+2} in the first i_j elements.

This in turn implies that $(\pi_1^j, \ldots, \pi_{i_j-1}^j, \pi_\ell^j) \in P_{\pi^{B+2}}$ for each $j \in [B+1]$, and, since the sequences $\pi^1, \ldots, \pi^{B+1}$ are also in S (meaning $(\pi_1^j, \ldots, \pi_{i_j-1}^j, \pi_\ell^j) \in T$ for each j), the claim is proven by contradiction, as the statement of Lemma 4 contradicts our assumption that $\pi^{B+2} \in S$. $\square$

So, we know that any member of S must have $P_\pi \cap T \leq B$, while any non-member still has $P_\pi \cap T \leq B + 1$. We will combine this fact with the subsequent claim to derive our final bound on $|S|$.

Claim 14. For each $i \in [\ell - 1]$, there exist at least $|S|$ distinct permutations π such that there exists an $s \in S$ for which

$$(s_1, \ldots, s_{i-1}, s_\ell) \in P_\pi$$

Proof. Beginning with $i = 1$, we observe that permutations with at least $|S|/(\ell - 1)!$ different last elements s_ℓ must occur in S (as there are only $(\ell - 1)!$ sequences with any given last element). Furthermore, any permutation in S with a certain last element s_ℓ must be such that $(s_\ell) \in P_\pi$ for a total of $(\ell-1)!$ different permutations π (i.e., anything beginning with s_ℓ), and different s_ℓ will of course correspond to disjoint sets of sequences. Thus, we conclude that the permutations in S will in total correspond to at least $(|S|/(\ell-1)!)(\ell-1)! = |S|$ distinct permutations.

For the remaining $i > 1$, we can apply the same logic to the distinct arrangements of the elements $(s_1, ..., s_{i-1})$ and s_ℓ . Among the permutations in S there must be a minimum of $|S|/(\ell-i)!$ such arrangements (since, given a fixed $(s_1, ..., s_{i-1}, s_\ell)$, there are $(\ell-i)!$ permutations possible), and sequences with each arrangement will have $(s_1, ..., s_{i-1}, s_\ell) \in P_\pi$ for a total of $(\ell-i)!$ distinct permutations π (i.e., any permutation beginning with $(s_1, ..., s_{i-1}, s_\ell)$). Hence, by the same logic as before, the permutations in S will once again correspond to at least $(|S|/(\ell-i)!)(\ell-i)! = |S|$ distinct permutations. $\square$

In total, we notice that, since at least $|S|$ distinct π have the property that there exists $s \in S$ such that $(s_1, ..., s_{i-1}, s_\ell) \in P_\pi$ for any $i \in [\ell-1]$, there are at least $|S|(\ell-1)$ distinct pairs (π, i) for which there exists $s \in S$ such that $(s_1, ..., s_{i-1}, s_\ell) \in P_\pi \cap T$. Furthermore, we recall that the permutations in S have $P_\pi \cap T \leq B$, while the remaining $\ell! - |S|$ permutations have $P_\pi \cap T \leq B+1$. This provides an upper bound of $B|S| + (\ell! - |S|)(B+1)$ on the number of pairs (π, i) with this property. 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 $|S|$:

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

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

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

$\square$