Refuting the Dream XOR Lemma via Ideal Obfuscation and Resettable MPC

Saikrishna Badrinarayanan
Snap, Mountain View, USA

Yuval Ishai
Technion, Haifa, Israel

Dakshita Khurana
University of Illinois, Urbana-Champaign, IL, USA

Amit Sahai
University of California Los Angeles, CA, USA
Center for Encrypted Functionalities, Los Angeles, CA, USA

Daniel Wichs
Northeastern University, Boston, MA, USA
NTT Research, Sunnyvale, CA, USA

Abstract

We provide counterexamples to the "dream" version of Yao's XOR Lemma. In particular, we put forward explicit candidates for hard predicates, such that the advantage of predicting the XOR of many independent copies does not decrease beyond some fixed negligible function, even as the number of copies gets arbitrarily large.

We provide two such constructions:

- Our first construction is in the ideal obfuscation model (alternatively, assuming virtual black-box obfuscation for a concrete class of circuits). It develops a general framework that may be of broader interest, and allows us to embed an instance of a *resettably*-secure multiparty computation protocol into a one-way function. Along the way, we design the first resettably-secure multiparty computation protocol for general functionalities in the plain model with super-polynomial simulation, under standard assumptions.
- The second construction relies on public-coin differing-inputs obfuscation (PCdiO) along with a certain form of hash-function security called extended second-preimage resistance (ESPR). It starts with a previously known counterexample to the dream direct-product hardness amplification based on ESPR, and uses PCdiO to upgrade it into a counterexample for the XOR lemma.

Prior to our work, even completely heuristic counterexamples of this type were not known.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography; Theory of computation → Cryptographic protocols

Keywords and phrases XOR Lemma, Resettable MPC, Obfuscation

Digital Object Identifier 10.4230/LIPIcs.ITC.2022.10

Funding *Saikrishna Badrinarayanan*: Work done while at UCLA.
Yuval Ishai: Supported in part by ERC Project NTSC (742754), BSF grant 2018393, and ISF grant 2774/20.
Dakshita Khurana: Supported in part by DARPA SIEVE project contract No. #HR00112020024, a gift from Visa Research, and a C3AI DTI award.
Amit Sahai: Supported in part from a Simons Investigator Award, DARPA SIEVE award, NTT Research, NSF Frontier Award 1413955, BSF grant 2012378, a Xerox Faculty Research Award, a Google Faculty Research Award, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through Award HR00112020024.
Daniel Wichs: Partially supported by NSF grants CNS-1750795, CNS- 2055510 and the Alfred P. Sloan Research Fellowship.

1 Introduction

Consider the following standard information-theoretic technique for hardness amplification. Suppose we have a joint distribution (X, B) such that the bit B is "weakly unpredictable" given X, in the sense that B has positive entropy conditioned on X. Then, for t independent samples (x_i, b_i) from (X, B), the XOR of all b_i can only be predicted from $(x_1, \ldots, x_t)$ with $2^{-\Omega(t)}$ advantage over a random guess. The question we address in this work is whether the same holds also in the computational setting.

The computational variant of the above XOR lemma is a central tool in complexity theory that first appeared in presentations of Yao's work [76]. It postulates that if a predicate P of an input $x \in \{0,1\}^\lambda$ is weakly unpredictable by algorithms of a certain complexity, with respect to some input distribution $X = X(\lambda)$, then $P(x_1, \ldots, x_t) = \bigoplus_{i=1}^{t} P(x_i)$ for large enough t is strongly unpredictable by algorithms within a related complexity bound. Yao stated this in the context of one-way functions, where the predicate P can be any "hard-core" bit of a one-way function f: in other words, P is an easy-to-compute Boolean function of the input to f that is hard to predict given the output of f. Here the distribution X is the output distribution of f on a uniformly random input.

Since it was first introduced, different versions of this lemma were proved in the literature, starting with Levin's proof [63], an alternate proof by Impagliazzo [55], and a third one by Goldreich et al. [44]. Unfortunately, all existing proofs of the XOR lemma are stuck at the following barrier: for *any fixed* negligible function $\mu(\cdot)$, no matter how large the polynomial $t(\cdot)$ is, we cannot prove that for large enough λ, the adversary's advantage drops to $\mu(\lambda)$. More concretely, we have no evidence that *any* polynomial $t(\lambda)$ number of repetitions bring the adversary's advantage down even to $\lambda^{-\log \lambda}$, if the original hardness of f was assumed to hold only against all polynomial-sized adversaries.

It is unclear why this barrier exists, and whether it is just an artifact of known proof techniques. Intuitively, it appears that the adversary's advantage should reduce arbitrarily if we perform sufficiently many repetitions. This was previously conjectured and termed the "dream version" of Yao's XOR lemma. It was formalized in [44], and used by [18] to obtain (a stronger flavor of) weak public-key cryptography from strong one-way functions. The dream XOR lemma, if true, would fundamentally change our understanding of how intractability works. It would help arbitrarily bring down errors in security arguments with sufficiently many repetitions; leading to exciting new constructions of primitives like non-interactive non-malleable commitments following [19], or easily obtain multi-instance security [16] with good parameters from standard hardness assumptions.

Previous Explanations. Over the years, there have been some explanations for why the dream XOR lemma eludes a proof. Black-box reduction based proofs of the dream XOR lemma are likely to fail for the following folklore reason. In order to prove that XOR parallel repetition brings the adversary's advantage down to some small probability $\mu(\lambda)$, we would need an efficient reduction that uses an adversary breaking security of the parallel repetition with advantage $\mu(\lambda)$, to break the security of a single instance with significantly larger advantage. But such a reduction cannot obtain any useful information from an attacker unless it succeeds at least once, and this could require $\mathsf{poly}(1/\mu(\lambda))$ attempts. Therefore, this reduction could be efficient for every inverse polynomial $\mu(\cdot)$, but would become inefficient as soon as $\mu(\cdot)$ is a fixed negligible function. This has been attributed to Rudich [44], who also proved that the dream XOR lemma does not hold in a relativized world, by introducing an oracle that inverts every tuple of instances with probability $\mu(\lambda)$.

Shaltiel and Viola [71] initiated a line of work on the impossibility of better black-box hardness amplification results, including the XOR lemma. A recent work by Shaltiel [70], improving on [49], rules out dream XOR lemmas with proofs by so-called "class reductions," which can exploit the efficiency of their oracle. Despite this progress, it is not clear that ruling out (even relaxed forms of) black-box reductions gives a strong evidence against the dream XOR lemma. There are quite a few examples for the surprising power of non-black-box techniques, both in cryptography and in complexity theory.

Another partial explanation was offered by Dodis et al. [34], who gave a counterexample to the "dream version" of the "direct-product lemma". In particular, they showed how to construct a *weak one-way function* that no polynomial-time adversary can invert with probability better than (e.g.,) $\frac{1}{2}$, but any arbitrary polynomial number of independent copies can be simultaneously inverted in polynomial time with advantage greater than $\lambda^{-\log \lambda}$. Their construction relies on an ad-hoc assumption on an un-keyed hash function, which was justified in the random-oracle model with auxiliary input. Since the direct-product lemma implies the XOR lemma without much loss in parameters, a dream version of the former would have implied the latter. Therefore, their work closes off one potential avenue toward proving the dream XOR lemma. Had their weak one-way function also been injective, then their counter-example to the dream direct-product lemma would have also immediately yielded a counter-example to the XOR lemma by taking the hard-core bits of the function. However, their weak one-way function was not injective, in which case the ability to find *some* pre-images and break one-wayness, does not imply the ability to find *the* correct hard-core bits and break the XOR lemma.

In this work, we ask the following question, explicitly left as an open problem in [34]:

Can we build an explicit counterexample to the dream version of the XOR lemma: a one-way function with a (weakly) hardcore bit, for which predicting the XOR of many hardcore bits does not reduce an adversary's advantage beyond a specific negligible function?

We note that, using a trusted setup that generates a trapdoor permutation (and can therefore invert it), one could heuristically obfuscate Rudich's oracle to obtain an explicit counterexample in the structured reference string model; however, beyond the need for trusted setup, this would induce a strong correlation between different instances. Such a correlation is inherently at odds with the idea of the dream XOR lemma, which conjectures hardness of *completely independent and uncorrelated instances*.

1.0.0.1 Are There Simple Heuristic Counterexamples?

To our knowledge, prior to our work, there were no candidate counterexamples to the dream XOR lemma *under any assumption*. In other words, there was not even a *heuristic* construction of an explicit predicate that can be plausibly conjectured to violate the dream XOR lemma. The difficulty of coming up with such heuristic counterexamples is arguably why the conjecture was put forward in the first place.

1.1 Overview of Results

We give two kinds of explicit counterexamples to the dream XOR lemma.

Our first counterexample develops a general framework that allows us to embed an instance of multiparty computation with *resettable* security into a non-interactive cryptosystem, such as a one-way function. It relies on ideal obfuscation, or alternatively, virtual-black-box (VBB)

obfuscation for some specific class of circuits, along with other standard hardness assumptions. Using this framework, the counterexample to the dream XOR lemma is extremely simple. We believe this framework may be of broader interest and may potentially be useful in designing counterexamples to other conjectures in cryptography. Along the way, we also develop the first resettably-secure multiparty computation protocol for general functionalities in the plain model with super-polynomial simulation, under standard assumptions. This protocol requires three rounds and assumes the existence of two-round sub-exponentially secure statistically sender-private OT. This result is of independent interest and achieves general feasibility for resettably-secure multiparty computation in the plain model; we evade prior impossibility results [47] by aiming for super-polynomial simulation.

Our second counterexample is a tailor-made construction whose sole purpose is to defy the dream XOR lemma. However, it avoids the need for ideal or VBB obfuscation. Instead, we can rely on a concrete obfuscation security property called *public-coins differing inputs obfuscation (PCdiO)* [12, 5, 22, 56], along with a hash function security property called *extended second-preimage resistance* (ESPR) [34] and injective one-way functions. The construction uses PCdiO to "upgrade" the counterexample of [34] for the dream direct-product lemma based on ESPR, into a counterexample for the dream XOR lemma.

PcdiO is a clean assumption which, as of today, all existing iO constructions can be conjectured to satisfy. While diO (without the public coin restriction) is known to be implausible [39], all known implausibility results crucially rely on "contrived" auxiliary information. All such negative results therefore do not generalize to PcdiO (we refer the reader to [56] for further discussion of PcdiO). Indeed, given recent progress on constructing iO (e.g. [58]), it is plausible that the PcdiO, that we rely on, may be reduced to well-studied assumptions.

We stress that even with ideal obfuscation, Rudich's oracle does not directly give rise to such counterexamples because it is not efficient. We believe that our techniques for obfuscation-based counterexamples will find other applications.

1.1.0.1 Are the Counterexamples Explicit?

Both of our counterexamples can be made fully explicit by making the same kind of leap of faith one makes when instantiating standard idealized models in cryptography (such as the random oracle model [17] or the generic group model [72]). For the first counterexample, one can use any existing iO construction (such as the one from [58]) as a heuristic substitute for special-purpose obfuscation. A similar heuristic has been suggested in prior works (see, e.g., [39, 42]). For the second, use iO construction as the PC-diO and use SHA-3 as the ESPR. Either way, one gets fully explicit constructions of one-way functions and hard-core bits for which the XOR lemma *provably* does not amplify hardness. Assuming that the function is actually one-way to begin with relies on a strong but explicit assumption. Nevertheless, if one wanted to prove that the dream XOR lemma holds, one would now have to show an attack against the one-wayness of these explicit one-way function candidates. This is very different from the previous oracle-based separations or (generalized) black-box impossibility results, which could be potentially circumvented by finding a novel non-black-box proof technique. We refer the reader to the end of Section 1.2 for additional discussions about the special-purpose obfuscation assumption.

1.2 First Counterexample

A New Paradigm. We suggest a new paradigm for obtaining counterexamples to parallel repetition. We use this paradigm to obtain an explicit counterexample to the dream version of the XOR lemma. Our paradigm can be thought of as implementing Rudich's oracle in a distributed manner between completely independent instances of a one-way function. Such distributed protocols are often cryptographically achievable via secure multi-party computation (MPC). Specifically, one could treat each instance of a one-way function as a participant in an MPC protocol, and implement an ideal functionality that with probability $\mu(\lambda)$, outputs the inverse of all the instances of the one-way function. Clearly, this would allow the XOR of the hardcore bits of individual instances of the one-way function to be predicted with an advantage of at least $\mu(\lambda)$. Indeed, a similar idea was employed by [34] in the context of a counterexample for the direct-product hardness amplification of signature schemes, by having the attacker leverage interaction with the signing oracle to run the protocol.

But in our case there is a crucial type mismatch: we would like to build one-way functions, an inherently non-interactive primitive, by relying on secure MPC, which is an inherently interactive protocol. We resolve this mismatch by relying on obfuscation to *non-interactively implement* the next message function of each party in a secure MPC protocol[1]. Specifically, the output of our one-way function consists of an obfuscation of the next-message function for the appropriate MPC.

When sufficiently many one-way functions are combined, an adversary can execute an MPC protocol between them by appropriately querying their next message functions, as a result, inverting all one-way functions simultaneously with probability $\mu(\lambda)$.

But obfuscating the next message function in this manner exposes individual one-way functions (i.e., "participants" in the MPC protocol) to a new threat model. An adversary can query an obfuscated program repeatedly and in an arbitrary order, amounting to what are called "resetting" attacks [27]. This requires us to confront the need for MPC protocols that are secure against such strong resetting attacks.[2]

Resettable MPC. The question of whether secure MPC can be achieved in a setting where participants can be simultaneously reset has previously been studied by [48, 47] and for the specific setting of zero knowledge in [27, 11, 20, 29, 28, 21, 30]. In particular, the work of [48] considered resetting attacks on only one party and obtained positive results. In the general setting where more than one party can be reset, [47] provided negative results for certain functionalities thereby ruling out a general purpose protocol for all functionalities. In addition, they obtained positive results for a limited class of entropic functionalities.

We observe that this negative result can be side-stepped by allowing our MPC simulator to run in super-polynomial time. Technically, we build on the recent concurrent secure MPC protocols with super-polynomial simulation in [10], which are themselves based on the notion of super-polynomial strong simulation [60]. MPC security with super-polynomial simulation [66, 68] turns out to be sufficient for many scenarios, including for building our counterexamples.

[1] This approach has previously been studied in multiple other contexts [38, 33, 7]. However, in all those cases, the inputs to the MPC are fixed apriori and (implicitly) hardwired into the obfuscated programs. Looking ahead, in our protocol, the inputs cannot be fixed apriori and we resort to using resettable MPC to overcome this crucial issue. Note that this issue is also the reason why we do not know how to use iO to implement our obfuscated next-message function, whereas the earlier works mentioned above were able to use only iO.

[2] A similar issue also came up in [34] when the MPC was embedded in a signing oracle, which is interactive but stateless. It was resolved similarly by relying on some form of resettable MPC.

As a contribution of independent interest, we obtain the first resettable MPC protocol for general functionalities, admitting super-polynomial simulation. Our protocol requires only three rounds of interaction, and assumes the existence of two-round sub-exponentially receiver-private and statistically sender-private OT, which can in turn be based on a variety of standard assumptions including the (sub-exponential) hardness of DDH, LWE, QRA, or DCRA [65, 3, 51, 9, 23, 35]. We state this result in the form of the following informal theorem.

▶ **Theorem 1** (Informal). *Assuming the existence of sub-exponentially receiver-private and statistically sender-private two-round OT, there exists a three-round resettably-secure MPC protocol for general functionalities, admitting a super-polynomial simulator.*

We stress that our resettable MPC protocol does not assume any form of obfuscation.

Our Counterexample to the Dream XOR Lemma. Armed with resettable MPC, we show that one-way functions with hard-core predicates to which the Dream XOR lemma provably does not apply can be obtained in the ideal obfuscation model, as follows: on input $x = (x_1||x_2) \in \{0,1\}^\lambda$, the one-way function f simply outputs an injective one-way function g applied to x_1, the first $\lambda/2$ bits of x, and uses the remaining $\lambda/2$ bits to obfuscate the next-message function of a participant in an MPC protocol. The ideal functionality for this MPC protocol obtains inputs (i.e., the x_1 values) from several participants, and with probability exactly $\mu(\lambda)$, outputs all these x_1 values in the clear. The hardcore bit of f on input $x = (x_1||x_2)$ is defined as the hardcore bit of g on input x_1.

We rely on security of the obfuscation scheme, the resettable MPC protocol and the one-way function g to argue that it is hard to recover x_1 (or predict the hard-core bit) of a single instance of this one-way function with probability significantly larger than $\mu(\lambda) \cdot \mathsf{poly}(\lambda) + \mathsf{negl}(\lambda)$. On the other hand, no matter how many times we repeat in parallel, the ability to execute a co-ordinated MPC program between all instances of the one-way function gives rise to an adversarial strategy that efficiently recovers all x_1 values, and therefore hardcore bits from all parallel instances, with probability at least $\mu(\lambda)$. For $\mu(\lambda) = 2^{-\sqrt{\lambda}}$, we prove the following informal theorem.

▶ **Theorem 2** (Informal). *Assuming resettably-secure MPC with super-polynomial simulation for general functionalities, for target negligible function $\mu(\lambda) = 2^{-\sqrt{\lambda}}$, there exists an explicit counterexample to the dream XOR lemma in the ideal obfuscation model. Furthermore, such a counterexample exists in the plain model under a plausible special-purpose obfuscation assumption.*

We choose to set $\mu(\lambda) = 2^{-\sqrt{\lambda}}$ primarily for the sake of simplicity in exposition, but our technique also generalizes to rule out arbitrary negligible $\mu(\lambda)$. While ideal obfuscation does not exist in the plain model [50, 12], the theorem applies relative to any world in which ideal obfuscation exists. This can refer to any *oracle* (in the complexity-theoretic sense) that enables ideal obfuscation, or given the ability to obfuscate functions using ideal trusted hardware. This counterexample is meaningful even when given ideal obfuscation, because all algorithms are given free access to the obfuscated function, and moreover the model does not introduce any shared randomness; as a result, instances of our one-way function remain *truly independent*.

Replacing Ideal Obfuscation with a Concrete Obfuscation Conjecture. In fact we go one step further and we postulate that a very specific functionality can be obfuscated, in a *virtual black-box* [12] (VBB) manner, *without auxiliary input*. As a result, assuming VBB or special-purpose obfuscation without auxiliary input for a specific class of circuits, we obtain counterexamples to the XOR lemma in the plain model.

How meaningful or plausible is this concrete assumption? While there are known impossibility results for VBB obfuscation of several functionalities, including PRFs *in the presence of auxiliary input*, the only known meaningful negative results on VBB without auxiliary input are the highly contrived "self-eating" programs developed by Barak et al. [12].

Despite it being plausible to embed the circuit family of [12] into *some* specific instantiations of resettable MPC and signatures, it appears extremely unlikely that *every* instantation of resettable MPC and signatures will have a [12]-style counterexample embedded into it. All we need for our counterexample is the existence of *one* VBB-obfuscatable family, which is compatible with all known evidence regarding VBB obfuscation. We stress again that our VBB assumption does not require security with respect to any auxiliary information. Indeed, such special-purpose obfuscation assumptions were used by Garg et al. [39] to prove negative results for differing-inputs obfuscation, and to this date, there are no known refutations of these types of conjectured assumptions for non-contrived circuits without auxiliary input.

Finally, we note that the recent work of [58] has shown how to construct indistinguishability obfuscation (iO) from standard assumptions. In addition, several other constructions of iO from new assumptions that look plausible and are quite simple to state have appeared [1, 57, 6, 2, 40, 41, 24, 25, 74, 31]. While we do not know how to use indistinguishability obfuscation to achieve our result, this recent progress suggests that perhaps achieving VBB obfuscation for circuit families such as ours may also be possible from standard assumptions. Our work offers further motivation for this important line of study.

1.3 Second Counterexample

Our second counterexample begins with the work of [34], which constructs a counterexample to a dream version of *direct-product* hardness amplification. In particular, they construct a hard relation R such that, given a uniformly random instance $\widetilde{x}$, no polynomial time adversary can find a witness w such that $(\widetilde{x}, w) \in R$ except with negligible probability. However, given t independent copies $\widetilde{x}_1, \ldots, \widetilde{x}_t$, the adversarial advantage of finding all t witnesses w_i such that $(\widetilde{x}_i, w_i) \in R$ does not decrease much as t gets large. Concretely, there is a polynomial time adversary that can find all t witnesses with probability (say) $2^{-\sqrt{\lambda}}$, no matter how large t is. This counterexample is based on a non-standard hash function security property called extended second-preimage resistance (ESPR), which is weaker than collision resistance, but is assumed to hold for a fixed (un-keyed) hash function against non-uniform attackers. The work of [34] justifies this assumption by showing that ESPR security holds in the random-oracle model with auxiliary input [73, 32], which models security properties for un-keyed hash functions with respect to non-uniform attackers.

As an initial idea to get a counterexample for the dream XOR lemma, one may hope to simply take a hard-core predicate for the relation R. The main issue is that the witness w for $\widetilde{x}$ is not unique, and so even if we are able to find *some* witness for $\widetilde{x}$, it does not mean we can compute *the* correct hard-core predicate. We resolve this by relying on an injective one way function $\hat{f}$ and a public-coins differing-inputs obfuscation (PCdiO) [5, 22, 56]. PCdiO is a strengthening of indistinguishability obfuscation (iO). All current constructions of iO can be conjectured to also satisfy PCdiO security, although no proofs of security for achieving PCdiO are as-yet known.

For the counterexample, we define a one-way function f that gets as input $x = (\hat{x}, \tilde{x}, r)$ and outputs $y = (\hat{y} = \hat{f}(\hat{x}), \tilde{x}, \tilde{C})$ where $\tilde{C}$ is an obfuscated circuit that takes as input a witness w, and if $(\tilde{x}, w) \in R$ it outputs $\hat{x}$, else $\perp$; we use r as the randomness for the obfuscation. We show that the function f is one-way and moreover, given the output y, it is hard to find $\hat{x}$. Intuitively, this follows because it is hard to find a valid witness w for $\tilde{x}$ that will make the obfuscated circuit output anything useful, and therefore the obfuscated circuit is indistinguishable from one that does not contain $\hat{x}$ and always outputs $\perp$; on the other hand the one-wayness of $\hat{f}$ says that it is hard to compute $\hat{x}$ from $\hat{y}$. We define a predicate $P(x)$ to be Goldreich-Levin hardcore bit of $\hat{x}$, and the above shows that $P(x)$ is a hard-core predicate of $f(x)$. On the other hand, it is easy to see that the dream XOR lemma does not hold for f, P. Given many values $y_i = f(x_i)$ for $i = 1, \ldots, t$, we can find all the witnesses $w_1, \ldots, w_t$ for $\tilde{x}_1, \ldots, \tilde{x}_t$ with probability $2^{-\sqrt{\lambda}}$. We then input these witnesses w_i to the respective obfuscated programs contained in y_i to recover $\hat{x}_i$, which allows us to recover all the hardcore-predicates $P(x_i)$ and therefore also $\bigoplus P(x_i)$.

We obtain the following informal theorem.

▶ **Theorem 3** (Informal). *Assuming the existence of public-coins differing-inputs obfuscation (PCdiO), extended second-preimage resistant (ESPR) hash functions, and injective one-way functions there exists an explicit counterexample to the dream XOR lemma.*

We observe that we do not actually even need PCdiO for the above counterexample, and (public-coins) extractable witness encryption suffices; instead of obfuscating the circuit that takes as input a witness w, and if $(\tilde{x}, w) \in R$ it outputs $\hat{x}$, we use witness encryption to encrypt the message $\hat{x}$ with respect to the statement $\tilde{x}$ for the relation R.

1.4 Counterexamples for the Goldreich-Levin Predicate

We note that, in both our counterexamples, only the choice of the one-way function is "artificial", but the hardcore predicate is just the Goldreich-Levin (GL) predicate, albeit only applied to one of the components of the input, rather than the entire input as a whole. One may ask whether it is possible to get a counterexample where the hardcore predicate is GL applied to the entire input, or whether there is hope that the dream XOR lemma would hold in this case. Although we do not know how to get a counterexample for the GL predicate applied to the pre-image of a one-way function, we can get a counterexample if we generalize to *one-way puzzles* and consider the GL predicate applied to the solution of such a puzzle. In a one-way puzzle, there is a randomized algorithm that generates random hard puzzles together with a solution that can be verified in polynomial time. Security says that no polynomial time attacker can solve a random hard puzzle with better than negligible probability. In this case, we can take the one-way functions from our counterexamples and define a hard puzzle consisting of the one-way function output, while the solution is just the small component of the one-way function's input that we apply GL to. In both examples, we can efficiently verify the solution. To summarize, the above shows that the dream XOR lemma fails, even when restricted to the specific GL predicate, at least when applied to general one-way puzzles.

1.5 Related Work

We note that there is much work in cryptography on designing counter-examples to statements that "should intuitively hold" but don't. Even when it becomes clear that a proof for such statements is lacking, a counter-example provides some tangible understanding of how things

can go wrong. Some such works that provide interesting counterexamples include: parallel repetition of multi-player games [37, 36, 54] and cryptographic protocols [15, 67], hardness amplification [34], circular security of encryption schemes [69, 61, 4, 62, 46, 75, 45], selective opening attack security of encryption and commitments [14, 53, 52], leakage amplification via parallel repetition [64, 59, 34], hardness of prediction via obfuscation [26]. Such counter-examples can point us to refined versions of the statement that may potentially still hold. Furthermore, exactly because such counter-examples "defy intuition", they often capture interesting ideas and techniques that turn out to be of greater value down the line. For example, the techniques developed in the context of circular security counter-examples [46] lead to positive results on obfuscation from LWE [75, 45].

2 Detailed Technical Overview for First Counterexample

As discussed in the introduction, we obtain our counterexample by implementing a variant of Rudich's oracle in a completely decentralized manner, without introducing any correlations between instances of our counterexample. In this section, we outline our construction and give an overview of our proof technique.

A central cryptographic primitive that enables decentralized computation is secure multi-party computation (MPC). MPC enables several mutually distrusting participants to jointly compute a function f of their private inputs while only revealing the output y of f applied to their joint inputs, and revealing no information to each player beyond their own input and the output y.

2.1 XOR lemma counterexample

We design a one-way function $\mathcal{G}$ as our counterexample to the XOR lemma. $\mathcal{G}$ on input uniform randomness $x = (\alpha||\beta)$, outputs $f(\alpha)$ for an injective one-way function f, and uses randomness β to build a "proxy" that participates in an MPC protocol. This proxy is simply the next-message function of a participant in the appropriate MPC protocol. We define a hardcore predicate for $\mathcal{G}$ on input $x = (\alpha||\beta)$ as the output of a hardcore predicate for $f(\alpha)$.

The MPC protocol will allow multiple such proxies to jointly emulate a randomized ideal functionality that obtains the value α as input (from each participating proxy), and with probability $\mu(\lambda)$ for some fixed negligible function $\mu(\cdot)$, outputs all the α values it obtained as input from all proxies, and otherwise outputs $\perp$. If we are able to implement such an MPC protocol, it is clear that no matter how many times we repeat, an adversary would be able to run the MPC protocol between all instances of the one-way function and obtain the α values, and therefore the hardcore predicates of all instances, simultaneously, with probability at least $\mu(\lambda)$.

But in order for this to be a valid counterexample, we also need to ensure that the hard-core predicate for a *single* instance of this one-way function is secure: that is, it cannot be predicted with probability close to 1. In fact, we prove that the hard-core bit cannot be predicted with advantage better than $\mathsf{negl}(\lambda)$. For this, we must ensure that even given a next-message function that has the value α hardwired in it, it is not possible to extract α except with probability $\mathsf{negl}(\lambda)$. As a first step, we *obfuscate* the next-message function instead of releasing it in the clear. Assuming that the next-message circuit can be obfuscated with virtual black-box security, this restricts all information that can be learned from the obfuscated program to information that can be obtained via input-output access alone. Note, however, that we are not done yet, since this still allows an adversary, who has access to the obfuscated circuit, to query the next-message function multiple times on the same partial

transcript, and potentially out of order. We use signatures to fix the latter issue and ensure that the adversary cannot make any meaningful queries on round $(i+1)$ of an MPC execution without querying on round i, for any $i \geq 1$. Unfortunately, this still leaves the obfuscated next-message circuit vulnerable to a "resetting" attack: where an adversary can query such a circuit to obtain multiple executions with the same partial transcript. Therefore, we need to harden our MPC protocol to obtain security against resetting attacks.

Finally, we point out one remaining (subtle) issue. Note that our counterexample is based on VBB-obfuscating the next-message function of a participant in an MPC protocol. In our setting, the adversary – given a single instance of our one-way function – can query this obfuscated program by generating his own next-message functions for imaginary MPC participants. Importantly, the adversary gets to *choose* the identity of these participants. But giving the adversary the freedom to choose the identity of a participant enables it to invoke the same participant (with the same input and randomness), many times using multiple identities in the same protocol. This could, for instance, allow the adversary to make multiple copies of the honest one-way function, and instantiate n parties, all having the same input and random tape, and potentially use this to manipulate the randomness of ideal functionality. More generally, when running an MPC protocol it is assumed that all players have different identities, and the adversary *does not* have the freedom to manipulate the identities of honest parties (or make multiple copies of any given honest identity).

In our setting, since every party is an obfuscated program, we must ensure that messages generated by each program are properly authenticated *under distinct verification keys*. We therefore add an explicit check to our VBB obfuscated program: the program will check that all verification keys are different and all messages in the input transcript are correctly signed. Ensuring that all verification keys are different guarantees that even in the ideal world, every participant of the MPC protocol has a different identity. Once distinct identities are carefully enforced in this way, the underlying MPC protocol (via underlying tools such as non-malleable commitments) ensures that the secret inputs of players to the MPC protocol are all independent of each other. We describe our formal construction assuming the existence of resettable MPC.

Next, we turn to building MPC secure against resetting attacks. As discussed in the introduction, such resettably-secure MPC protocols are only known for ideal functionalities that satisfy a specific property [47][3]. Since our ideal functionality does not satisfy this property, we develop an appropriate resettable MPC protocol for use in our setting. We relax security to allow superpolynomial simulation, because that suffices for our applications. In what follows, we provide an overview of this protocol.

2.2 Resettable MPC

Our construction of MPC with superpolynomial simulation (SPS), secure against resetting attacks, proceeds in two steps.

- **Resettably-Secure SPS MPC against Semi-malicious adversaries.** As a first step, we build resettably secure MPC against semi-malicious adversaries. A semi-malicious adversary always produces messages in the support of honestly generated messages, and

[3] We point out that [47] give resettably secure protocols with polynomial simulation, but only for (very limited) functionalities that satisfy a special property. Informally, they require that there exist compression and decompression algorithms such that with overwhelming probability, for all possible inputs, the compression algorithm generates a small suggestion string s which helps the decompression algorithm correctly predict the output of the adversary in any given session (given the adversary's input-output pairs in prior sessions and its current input, without knowledge of the honest party's input). We refer the reader to [47] for details and a formal description of this property.

writes the input and randomness used to generate these messages on a special witness tape. We compile a semi-malicious MPC protocol that is *not necessarily secure against resetting attacks* into one that is resettably secure against semi-malicious adversaries. Our compiler simply appends a round to the beginning of the protocol; in this round, each party P_i must commit to its input and to a uniformly sampled PRF key K_i. Next, all participants execute the semi-malicious resettably-insecure MPC protocol, except in every round, each party P_i uses randomness that is generated by evaluating the PRF with key K_i on the transcript so far.

The effect of this is that all protocol messages sent by P_i become a deterministic function of the values committed in the first round and the transcript so far. In fact, *any* protocol transcript generated by any combination of honest and semi-malicious adversarial participants is a deterministic function of the first round.

Therefore, any adversary that resets to a previous point in the protocol after round 1 and behaves semi-maliciously, must send exactly the same messages each time. On the other hand, any adversary that resets to the middle of round 1 sees an entirely new execution of the semi-malicious protocol. As a result, we are able to prove that the resulting resettably-secure semi-malicious MPC protocol admits a polynomial time simulator. Next, we compile to obtain an MPC protocol resettably-secure against fully malicious adversaries.

- **Resettably-Secure SPS MPC against Malicious Adversaries.** Our compiler is identical to the one in [10], which builds three round concurrent MPC with superpolynomial simulation. Here, in addition, we prove resettable security. The only difference is that while the compiler in [10] uses as subroutine an underlying stand-alone secure MPC against semi-malicious adversaries; we instead use a *resettably* secure MPC against semi-malicious adversaries (as described above). As ingredients, beyond the resettable semi-malicious protocol, this compiler relies on two round non-malleable commitments and two round zero knowledge arguments with super-polynomial strong simulation [60]. We note that all of these ingredients are secure (or can be easily modified to be secure) under resetting attacks. Furthermore, following [10], we show that the resulting protocol admits a straight-line superpolynomial time simulator. Resettable security of the resulting protocol against malicious adversaries follows by a careful analysis of this simulator and makes use of the resettable security of all ingredients. We formalize these ideas in the full version of the paper.

3 Paper Organization

We describe the first counterexample, assuming resettable MPC, in Appendix B. We first describe some relevant preliminaries in Appendix A. Due to lack of space, we defer the construction of resettable MPC and the second counterexample to the full version of the paper.

References

1 Shweta Agrawal. Indistinguishability obfuscation without multilinear maps: New methods for bootstrapping and instantiation. In *EUROCRYPT*, 2019.

2 Shweta Agrawal and Alice Pellet-Mary. Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear FE. In *EUROCRYPT*, 2020.

3 William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In *EUROCRYPT*, 2001.

4 Navid Alamati and Chris Peikert. Three's compromised too: Circular insecurity for any cycle length from (ring-)lwe. In *CRYPTO*, 2016.

5 Prabhanjan Ananth, Dan Boneh, Sanjam Garg, Amit Sahai, and Mark Zhandry. Differing-inputs obfuscation and applications. *IACR Cryptology ePrint Archive*, 2013. URL: `http://eprint.iacr.org/2013/689`.

6 Prabhanjan Ananth, Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. Indistinguishability obfuscation without multilinear maps: New paradigms via low degree weak pseudorandomness and security amplification. In *CRYPTO*, 2019.

7 Prabhanjan Ananth, Aayush Jain, Moni Naor, Amit Sahai, and Eylon Yogev. Universal constructions and robust combiners for indistinguishability obfuscation and witness encryption. In *CRYPTO*, 2016.

8 Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In *EUROCRYPT*, 2012.

9 Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In *ASIACRYPT*, 2017.

10 Saikrishna Badrinarayanan, Vipul Goyal, Abhishek Jain, Dakshita Khurana, and Amit Sahai. Round optimal concurrent MPC via strong simulation. In *TCC*, 2017.

11 Boaz Barak, Oded Goldreich, Shafi Goldwasser, and Yehuda Lindell. Resettably-sound zero-knowledge and its applications. *FOCS*, 2001.

12 Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *CRYPTO*, 2001.

13 Boaz Barak and Amit Sahai. How to play almost any mental game over the net - concurrent composition via super-polynomial simulation. In *FOCS*, 2005.

14 Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In *EUROCRYPT*, 2009.

15 Mihir Bellare, Russell Impagliazzo, and Moni Naor. Does parallel repetition lower the error in computationally sound protocols? In *FOCS*, 1997.

16 Mihir Bellare, Thomas Ristenpart, and Stefano Tessaro. Multi-instance security and its application to password-based cryptography. In *CRYPTO*, 2012.

17 Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS*, 1993.

18 Eli Biham, Yaron J. Goren, and Yuval Ishai. Basing weak public-key cryptography on strong one-way functions. In *TCC*, 2008.

19 Nir Bitansky and Huijia Lin. One-message zero knowledge and non-malleable commitments. In *TCC*, 2018.

20 Nir Bitansky and Omer Paneth. On the impossibility of approximate obfuscation and applications to resettable cryptography. In *STOC*, 2013.

21 Nir Bitansky and Omer Paneth. On non-black-box simulation and the impossibility of approximate obfuscation. *SIAM J. Comput.*, 2015.

22 Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 52–73. Springer, Heidelberg, February 2014. `doi:10.1007/978-3-642-54242-8_3`.

23 Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In *TCC 2018*, 2018.

24 Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate io from homomorphic encryption schemes. In *EUROCRYPT*, 2020.

25 Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and pairings are not necessary for io: Circular-secure LWE suffices. *IACR Cryptol. ePrint Arch.*, 2020. URL: `https://eprint.iacr.org/2020/1024`.

26 Christina Brzuska, Pooya Farshim, and Arno Mittelbach. Indistinguishability obfuscation and uces: The case of computationally unpredictable sources. In *CRYPTO*, 2014.

27 Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *STOC*, 2000.

28 Kai-Min Chung, Rafail Ostrovsky, Rafael Pass, Muthuramakrishnan Venkitasubramaniam, and Ivan Visconti. 4-round resettably-sound zero knowledge. In *TCC*, 2014.

29 Kai-Min Chung, Rafail Ostrovsky, Rafael Pass, and Ivan Visconti. Simultaneous resettability from one-way functions. In *FOCS*, 2013.

30 Kai-Min Chung, Rafael Pass, and Karn Seth. Non-black-box simulation from one-way functions and applications to resettable security. *SIAM J. Comput.*, 2016.

31 Lalita Devadas, Willy Quach, Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Succinct LWE sampling, random polynomials, and obfuscation. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part II*, volume 13043 of *Lecture Notes in Computer Science*, pages 256–287. Springer, 2021. doi:10.1007/978-3-030-90453-1_9.

32 Yevgeniy Dodis, Siyao Guo, and Jonathan Katz. Fixing cracks in the concrete: Random oracles with auxiliary input, revisited. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 473–495. Springer, Heidelberg, April / May 2017. doi:10.1007/978-3-319-56614-6_16.

33 Yevgeniy Dodis, Shai Halevi, Ron D. Rothblum, and Daniel Wichs. Spooky encryption and its applications. In *CRYPTO*, 2016.

34 Yevgeniy Dodis, Abhishek Jain, Tal Moran, and Daniel Wichs. Counterexamples to hardness amplification beyond negligible. In Ronald Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 476–493. Springer, Heidelberg, March 2012. doi:10.1007/978-3-642-28914-9_27.

35 Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 3–32. Springer, Heidelberg, August 2019. doi:10.1007/978-3-030-26954-8_1.

36 Uriel Feige. On the success probability of the two provers in one-round proof systems. In *Structure in Complexity Theory Conference*. IEEE Computer Society, 1991.

37 Lance Fortnow. The complexity of perfect zero-knowledge. *Advances in Computing Research*, 1989.

38 Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure MPC from indistinguishability obfuscation. In *TCC*, 2014.

39 Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In *CRYPTO*, 2014.

40 Romain Gay, Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from simple-to-state hard problems: New assumptions, new techniques, and simplification. In *EUROCRYPT*, 2021.

41 Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. In *STOC*, 2021.

42 Craig Gentry, Shai Halevi, Mariana Raykova, and Daniel Wichs. Outsourcing private RAM computation. In *FOCS*, 2014.

43 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, 1989.

44 Oded Goldreich, Noam Nisan, and Avi Wigderson. *On Yao's XOR-Lemma*, pages 273–301. Springer Berlin Heidelberg, 2011. doi:10.1007/978-3-642-22670-0_23.

45 Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In *FOCS*, 2017.

46 Rishab Goyal, Venkata Koppula, and Brent Waters. Separating semantic and circular security for symmetric-key bit encryption from the learning with errors assumption. In *EUROCRYPT*, 2017.

47 Vipul Goyal and Hemanta K. Maji. Stateless cryptographic protocols. In *FOCS*, 2011.

48 Vipul Goyal and Amit Sahai. Resettably secure computation. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 54–71. Springer, Heidelberg, April 2009. doi:10.1007/978-3-642-01001-9_3.

49 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In *FOCS*, 2018.

50 Satoshi Hada. Zero-knowledge and code obfuscation. In *ASIACRYPT*, 2000.

51 Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *J. Cryptol.*, 2012.

52 Dennis Hofheinz, Vanishree Rao, and Daniel Wichs. Standard security does not imply indistinguishability under selective opening. In *TCC*, 2016.

53 Dennis Hofheinz and Andy Rupp. Standard versus selective opening security: Separation and equivalence results. In *TCC*, 2014.

54 Justin Holmgren and Lisa Yang. The parallel repetition of non-signaling games: counter-examples and dichotomy. In Moses Charikar and Edith Cohen, editors, *51st Annual ACM Symposium on Theory of Computing*, pages 185–192. ACM Press, June 2019. doi:10.1145/3313276.3316367.

55 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *FOCS*, 1995.

56 Yuval Ishai, Omkant Pandey, and Amit Sahai. Public-coin differing-inputs obfuscation and its applications. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 668–697. Springer, Heidelberg, March 2015. doi:10.1007/978-3-662-46497-7_26.

57 Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. How to leverage hardness of constant-degree expanding polynomials over r to build io. In *EUROCRYPT*, 2019.

58 Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *STOC*, 2021.

59 Abhishek Jain and Krzysztof Pietrzak. Parallel repetition for leakage resilience amplification revisited. In *TCC*, 2011.

60 Dakshita Khurana and Amit Sahai. How to achieve non-malleability in one or two rounds. In *FOCS*, 2017.

61 Venkata Koppula, Kim Ramchen, and Brent Waters. Separations in circular security for arbitrary length key cycles. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 378–400. Springer, Heidelberg, March 2015. doi:10.1007/978-3-662-46497-7_15.

62 Venkata Koppula and Brent Waters. Circular security separations for arbitrary length cycles from LWE. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 681–700. Springer, Heidelberg, August 2016. doi:10.1007/978-3-662-53008-5_24.

63 Leonid A. Levin. One-way functions and pseudorandom generators. In *STOC*, 1985.

64 Allison B. Lewko and Brent Waters. On the insecurity of parallel repetition for leakage resilience. In *51st Annual Symposium on Foundations of Computer Science*, pages 521–530. IEEE Computer Society Press, October 2010. doi:10.1109/FOCS.2010.57.

65 Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *SODA*, 2001.

66 Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *EUROCRYPT*, 2003.

67 Krzysztof Pietrzak and Douglas Wikström. Parallel repetition of computationally sound protocols revisited. In *TCC*, 2007.

68 Manoj Prabhakaran and Amit Sahai. New notions of security: achieving universal composability without trusted setup. In László Babai, editor, *STOC*, 2004.

69 Ron Rothblum. On the circular security of bit-encryption. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 579–598. Springer, Heidelberg, March 2013. doi:10.1007/978-3-642-36594-2_32.

70 Ronen Shaltiel. Is it possible to improve yao's XOR lemma using reductions that exploit the efficiency of their oracle? In *APPROX/RANDOM*, 2020.

71 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 2010.

72 Victor Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT*, 1997.

73 Dominique Unruh. Random oracles and auxiliary input. In *CRYPTO*, 2007.

74 Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious LWE sampling. In *EUROCRYPT*, 2021.

75 Daniel Wichs and Giorgos Zirdelis. Obfuscating compute-and-compare programs under LWE. In Chris Umans, editor, *58th Annual Symposium on Foundations of Computer Science*, pages 600–611. IEEE Computer Society Press, October 2017. doi:10.1109/FOCS.2017.61.

76 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *FOCS*, 1982.

A Preliminaries

Let λ denote the security parameter.

A.1 Virtual Black Box Obfuscation

We recall the definition of Virtual Black-Box (VBB) obfuscation from Barak et al.[12]. In this definition, we don't allow any auxiliary inputs (therefore making our assumptions weaker).

▶ **Definition 4** (Virtual Black-Box Obfuscation). *For any polynomial $t(\cdot)$, circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$, a uniform PPT oracle machine* Obf *is a "Virtual Black-Box" Obfuscator for $\mathcal{C}$ if the following conditions are satisfied:*

- *Functionality: For every $\lambda \in \mathbb{N}$ and every input x:*

$$\Pr[(\mathsf{Obf}(C))(x) \neq C(x)] \leq \mathsf{negl}(|C|),$$

  where the probability is over the coins of $C \leftarrow \mathcal{C}_\lambda$.
- *Polynomial Slowdown: There exists a polynomial $p(\cdot)$ such that for every $\lambda \in \mathbb{N}$ and every $C \in \mathcal{C}_\lambda$, $|\mathsf{Obf}(C)| \leq p(|C|)$.*
- *Virtual Black-Box: For every PPT adversary $\mathcal{A}$ there exists a PPT simulator* $\mathsf{Obf.Sim}$*, and a negligible function μ such that for every $\ell \in \mathbb{N}$ and every $C \in \mathcal{C}_\ell$:*

$$\left|\Pr\left[\mathcal{A}\left(\mathsf{Obf}(C)\right) = 1\right] - \Pr\left[\mathsf{Obf.Sim}^C\left(1^{|C|}\right) = 1\right]\right| \leq \mu(|C|) \ ,$$

  where the probabilities are over the coins of $\mathsf{Obf.Sim}$ *and* Obf*.*

A.2 Resettable MPC

We define the notion of Multiparty Computation (MPC) that is resettably secure similar to the definition in Goyal and Maji [47]. The difference in that, in our setting, we consider a simulator that can run in super-polynomial time [66, 13].

Remark. We note that security holds only when the identities of all the parties in the MPC protocol are distinct. Consider n parties $P_1, \ldots, P_n$. An n-party functionality f is a (possibly randomized) mapping of n inputs to n outputs. A secure multi-party computation protocol π with security parameter λ for computing a functionality f is a protocol running in time $\mathsf{poly}(\lambda)$ with the goal of computing the output $f(x_1, \ldots, x_n)$.

The security of a protocol π (with respect to a functionality f) is defined by comparing the real-world execution of the protocol with an ideal-world evaluation of f by a trusted party. More concretely, it is required that for every adversary $\mathcal{A}$, which attacks the real execution of the protocol, there exist an adversary Sim, also referred to as a simulator, which can *achieve the same effect* in the ideal-world. We denote the set of all inputs by $\vec{x} = (x_1, \ldots, x_n)$. The adversary controls a set of parties and at any point during the execution of the protocol it can reset any of the honest parties. We shall consider computational security against parties which have been statically corrupted by the adversary. All honest parties have their random tape independently chosen but when an adversary resets a party, it reuses the same random tape (and the same input).

A.2.0.1 The real execution

In the real execution of the n-party protocol π for computing f in the presence of an adversary $\mathcal{A}$, the honest parties follow the instructions of π. The adversary $\mathcal{A}$, on input λ, outputs the identities of the honest parties, the indices I and identities in 2^λ of corrupted parties, the inputs of the corrupted parties, and an auxiliary input z. $\mathcal{A}$ sends all messages in place of corrupted parties and may follow an arbitrary polynomial-time strategy. The adversary can reset any honest party at any point of time during the execution of the protocol (and potentially even bring them out of sync). Recall that when an adversary resets a party, the reset party reuses the same random tape (and the same input).
The interaction of $\mathcal{A}$ with protocol π defines a random variable $\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(\lambda, \vec{x})$ whose value is determined by the coin tosses of the adversary and the honest players. This random variable contains the output of the adversary (which may be an arbitrary function of its view) as well as the outputs of the uncorrupted parties at the end of the protocol. We let $\mathsf{REAL}_{\pi,\mathcal{A}(z),I}$ denote the distribution ensemble $\{\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(\lambda, \vec{x})\}_{\lambda\in\mathsf{N},\langle\vec{x},z\rangle\in\{0,1\}^*}$.

A.2.0.2 The ideal execution – security with abort

In the ideal world, there is a mutually trusted party which can aggregate the inputs provided to it by the various parties, perform the computation $f(\cdot)$ on their behalf and provide them their respective outputs. An ideal execution for a function f in the presence of an ideal-world adversary (simulator) Sim proceeds as follows:

- **Send inputs to the trusted party:** Honest parties send their inputs to the trusted party; but corrupted parties may decide to send modified inputs to the trusted party, as instructed by Sim. Let x'_i denote the value sent by P_i.
- **Trusted party sends output to the adversary:** The trusted party computes $f(x'_1, \ldots, x'_n) = (y_1, \ldots, y_n)$ and sends $\{y_i\}_{i\in I}$ to the adversary.
- **Adversary instructs trusted party to abort or continue:** This is formalized by having the adversary send either a continue or abort message to the trusted party. In the latter case, the trusted party sends to each uncorrupted party P_i its output value y_i. In the former case, the trusted party sends the special symbol $\perp$ to each uncorrupted party.

- **Resets:** The adversary can reset the ideal world at any point of time. When the adversary decides to reset the ideal world, it requests the trusted party to reset all honest parties and the trusted party sends a reset signal to all honest parties. At this point, the ideal world returns to the first stage where honest parties feed their inputs to the ideal functionality. The honest party inputs do not change between resets.
- **Outputs:** Sim outputs an arbitrary function of its view, and the honest parties output the values obtained from the trusted party.

Sim's interaction with the trusted party defines a random variable $\mathsf{IDEAL}_{f_\perp,\mathcal{A}(z),I}(\lambda,\vec{x})$ that denotes the distribution ensemble $\{\mathsf{IDEAL}_{f_\perp,\mathsf{Sim}(z),I}(\lambda,\vec{x})\}_{\lambda\in\mathsf{N},\langle\vec{x},z\rangle\in\{0,1\}^*}$ where the subscript "$\perp$" indicates that the adversary can abort computation of f. Having defined the real and the ideal worlds, we now proceed to define our notion of security.

▶ **Definition 5** (Resettable MPC with Straight-Line, Black-Box Superpolynomial Simulation)**.** *Let λ be the security parameter. Let f be an n-party randomized functionality, and π be an n-party protocol for $n \in \mathsf{N}$.*

We say that π computes f with resettable straight-line, black-box superpolynomial simulation in the presence of malicious adversaries if for every PPT adversary $\mathcal{A}$ there exists a super-polynomial time simulator Sim that interacts with the adversary via straight-line black-box queries, such that for any $I \subset [n]$:

$$|\Pr[\mathsf{REAL}_{\pi,\mathcal{A}(z),I}(\lambda,\vec{x})=1] - \Pr[\mathsf{IDEAL}_{f_\perp,\mathsf{Sim}(z),I}(\lambda,\vec{x})=1]| = \mathsf{negl}(\lambda)$$

where $\vec{x} = \{x_i\}_{i\in[n]} \in \{0,1\}^*$ and $z \in \{0,1\}^*$.

The simulator Sim is allowed to reset the functionality in the ideal world several times and the number of times the ideal functionality is reset by the simulator could possibly be more than the number of resets performed by the adversary in the real world, though this number must be a polynomial in the security parameter.

A.3 Security Against Semi-Malicious Adversaries

We take this definition verbatim from [8]. A semi-malicious adversary is modeled as an interactive Turing machine (ITM) which, in addition to the standard tapes, has a special witness tape. In each round of the protocol, whenever the adversary produces a new protocol message msg on behalf of some party P_k, it must also write to its special witness tape some pair (x, r) of input x and randomness r that explains its behavior. More specifically, all of the protocol messages sent by the adversary on behalf of P_k up to that point, including the new message m, must exactly match the honest protocol specification for P_k when executed with input x and randomness r.

Also, we assume that the attacker is rushing and hence may choose the message m and the witness (x, r) in each round adaptively, after seeing the protocol messages of the honest parties in that round (and all prior rounds). The adversary may also choose to abort the execution on behalf of P_k in any step of the interaction.

B Counterexample via VBB Obfuscation and Resettable MPC

We start this section by defining hard-core predicates. Next, we state the dream version of Yao's XOR lemma, and then we describe our counterexample.

▶ **Definition 6** ($\epsilon(\lambda)$-Hard Core Predicate.). *Let λ denote a security parameter and $m = m(\lambda), n = n(\lambda)$ be polynomials in λ. An $\epsilon(\lambda)$-hard core predicate of a one-way function $f : \{0,1\}^{m(\lambda)} \to \{0,1\}^{n(\lambda)}$ is a predicate $P : \{0,1\}^{m(\lambda)} \to \{0,1\}$ such that for every polynomial-time non-uniform $\mathcal{A}$ and every λ,*

$$\Pr_{x \leftarrow \{0,1\}^{\lambda}}[\mathcal{A}(1^{\lambda}, f(x)) = P(x)] \leq \frac{1}{2} + \epsilon(\lambda)$$

▶ **Lemma 7** (Hard Core Predicate for a One-Way Function). *[43] If f is a one-way function, then $h(x,r) = \langle x, r \rangle (\mathsf{modulo}\ 2)$ is an $\epsilon(\lambda)$-hard core predicate, according to Definition 6, for the one-way function f' defined by $f'(x,r) = (f(x), r)$, for some $\epsilon(\lambda) = \mathsf{negl}(\lambda)$.*

▶ **Conjecture 8** (Dream version of Yao's XOR Lemma). *[18] Fix $\mu(\lambda) = 2^{-\sqrt{\lambda}}$. Let f denote a one-way function and P denote an $\epsilon(\lambda)$-hard core predicate according to Definition 6. Then for any $t = \mathsf{poly}(\lambda)$, $P^{(t)}(x_1, \ldots, x_t) = \bigoplus_{i \in [t]} P(x_i)$ is an $\epsilon'(\lambda)$-hard core predicate for $f'(x_1||x_2|| \ldots ||x_t) \triangleq f(x_1)||f(x_2)|| \ldots ||f(x_t)$, such that $\epsilon'(\lambda) \leq \epsilon(\lambda)^{t(\lambda)} + \mu(\lambda)$.*

We note that this conjecture is a special case of (and is therefore implied by) the dream conjecture first formulated in [44]. Also, we fixed $\mu(\lambda)$ to be an arbitrary negligible function in λ, specifically, we set it to $2^{-\sqrt{\lambda}}$ for ease of exposition. However, we note that our refutation of this conjecture can be generalized to refute arbitrary negligible functions $\mu(\cdot)$ by setting the parameters of our one-way function accordingly. That is, our proof can be generalized to show that for every negligible function $\nu(\cdot)$, there exists a one-way function that refutes Conjecture 8 with $\mu(\cdot) = \nu(\cdot)$. We will now construct a one-way function for which Conjecture 8 does not hold.

B.1 Construction

We define a one-way function $\mathcal{G} : \{0,1\}^{2\lambda} \to \{0,1\}^{p'(\lambda)}$, where $p'(\cdot)$ is a polynomial in λ, the exact value of which will be determined later.

B.1.0.1 Notation and Primitives Used

- Let $g : \{0,1\}^{\lambda} \to \{0,1\}^{p(\lambda)}$ be any injective one way function and h denote the Golreich-Levin [43] hardcore bit for this one way function.
- Let π^{Res} denote any resettably secure MPC protocol with superpolynomial simulation. Let $(\pi^{\mathsf{Res}}.\mathsf{NextMsg}_1, \pi^{\mathsf{Res}}.\mathsf{NextMsg}_2, \ldots, \pi^{\mathsf{Res}}.\mathsf{NextMsg}_{\mathsf{n}})$ denote the algorithms used by each party to compute the messages in each of the rounds and $\pi^{\mathsf{Res}}.\mathsf{OUT}$ denote the algorithm used by each party to compute its final output. Also, let Trans_i denote all messages sent in an execution of π^{Res} up to round i. Let $\mathsf{Res.Sim}$ denote the straight-line (super-polynomial) simulator for this protocol. Let $(\mathsf{Res.Sim.NextMsg}_1, \ldots, \mathsf{Res.Sim.NextMsg}_{\mathsf{n}})$ denote the algorithms used by the simulator to compute the messages in each of the rounds and $\mathsf{Res.Sim.Out}$ denote the algorithm used to compute the final output on behalf of the honest parties. Let $\ell(\lambda)$ denote the length of the randomness required by each party on inputs of length λ. Let $s(\lambda)$ denote the maximum size of the circuit representation of $(\pi^{\mathsf{Res}}.\mathsf{NextMsg}_1, \pi^{\mathsf{Res}}.\mathsf{NextMsg}_2, \ldots, \pi^{\mathsf{Res}}.\mathsf{NextMsg}_{\mathsf{n}}, \pi^{\mathsf{Res}}.\mathsf{OUT})$ in this protocol.
- Let $(\mathsf{Obf}, \mathsf{Eval})$ be a VBB obfuscation scheme and $\mathsf{Obf.Sim}$ denote its simulator. Let $r(\lambda)$ denote the length of the randomness used to obfuscate programs of size $s(\lambda)$, and $s'(\lambda)$ denote the size of the obfuscated program.
- Let $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ be a sub-exponentially unforgeable signature scheme.
- Let $\mathsf{PRG} : \{0,1\}^{\lambda} \to \{0,1\}^{\sqrt{\lambda}+\lambda+r(\lambda)+\ell(\lambda)}$ be a pseudorandom generator.

B.1.0.2 Construction

The one-way function $\mathcal{G}$, on input $x = (a, b)$, where $|a| = |b| = \lambda$, is:

- Compute $y = g(a)$.
- Compute $(\alpha, \beta, \gamma, \delta) \leftarrow \mathsf{PRG}(b)$ where α is of length $\sqrt{\lambda}$, β is of length λ, γ is of length $\ell(\lambda)$ and δ is of length $r(\lambda)$. Set $\mathsf{r}^{\mathsf{Res}} = \gamma$.
- Generate $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda; \beta)$.
- Compute $\widehat{\mathsf{C}} = \mathsf{Obf}(1^\lambda, \mathsf{C})$ using randomness δ where program C is described in Figure 1.
- Output $(\widehat{\mathsf{C}})$. We set $p'(\lambda) = |\widehat{\mathsf{C}}|$.

Hardwired values: $(a, y, \alpha, \mathsf{sk}, \mathsf{vk}, \mathsf{r}^{\mathsf{Res}})$

1. Output y. Additionally, do the following.
2. If input = ("Reveal Identity for MPC"): output vk.
3. If input = ("MPC", "Begin", "Party" $\mathsf{j}, \overrightarrow{\mathsf{vk}}$):
   - Output $\perp$ if any of the following happen:
     - $\mathsf{vk}^j \neq \mathsf{vk}$,
     - $\{\mathsf{vk}^k\}_{k \in [\lambda^2]}$ are not distinct,
   - Output $\sigma = \mathsf{Sign}(\mathsf{sk}, (\overrightarrow{\mathsf{vk}}||j))$
4. If input = ("MPC", "Round" 1, "Party" $\mathsf{j}, \overrightarrow{\mathsf{vk}}, \sigma$):
   - Output $\perp$ if any of the following happen:
     - $\mathsf{vk}^j \neq \mathsf{vk}$,
     - $\{\mathsf{vk}^k\}_{k \in [\lambda^2]}$ are not distinct,
     - $\mathsf{Verify}(\mathsf{vk}, \sigma, (\overrightarrow{\mathsf{vk}}||j)) \neq 1$.
   - Begin resettable MPC protocol π^{Res} amongst λ^2 parties $\mathsf{P}_1, \ldots, \mathsf{P}_{\lambda^2}$ playing the role of P_j with identity vk. Use input α, randomness $\mathsf{r}^{\mathsf{Res}}$ to evaluate functionality f (Figure 2). Let ℓ be the number of rounds of π^{Res}.
   - Compute $\mathsf{msg}_1 = \pi^{\mathsf{Res}}.\mathsf{NextMsg}_1(1^\lambda, \alpha; \mathsf{r}^{\mathsf{Res}})$, $\sigma_1 = \mathsf{Sign}(\mathsf{sk}, (\overrightarrow{\mathsf{vk}}||\mathsf{msg}_1||1||j))$.
   - Output $(\mathsf{msg}_1, \sigma_1)$.
5. If input = ("MPC", "Round" i, "Party" $\mathsf{j}, \overrightarrow{\mathsf{vk}}, \overrightarrow{\mathsf{msg}}_{i-1}, \sigma_{i-1}, \mathsf{Trans}_{i-2}$):
   - Let Trans_{i-2} denote the transcript till round $(i-2)$ and $\overrightarrow{\mathsf{msg}}_{i-1}$ denote the set of messages sent by all parties in round $(i-1)$ of protocol π.
   - Output $\perp$ if any of the following happen:
     - $\mathsf{vk}^j \neq \mathsf{vk}$,
     - $\{\mathsf{vk}^k\}_{k \in [\lambda^2]}$ are not distinct,
     - $\mathsf{Verify}(\mathsf{vk}, \sigma_{i-1}, (\overrightarrow{\mathsf{vk}}||\mathsf{Trans}_{i-2}||\mathsf{msg}_{i-1}||i-1||j)) \neq 1$.
   - If round number $i \leq \ell$, do:
     - Continue π^{Res}. Compute $\mathsf{msg}_i = \pi^{\mathsf{Res}}.\mathsf{NextMsg}_i(\mathsf{Trans}_{i-2}, \overrightarrow{\mathsf{msg}}_{i-1}, \alpha; \mathsf{r}^{\mathsf{Res}})$.
     - Set $\mathsf{Trans}_{i-1} = (\mathsf{Trans}_{i-2}||\overrightarrow{\mathsf{msg}}_{i-1})$, $\sigma_i = \mathsf{Sign}(\mathsf{sk}, (\overrightarrow{\mathsf{vk}}||\mathsf{Trans}_{i-1}||\mathsf{msg}_i||\ i||j))$.
     - Output $(\mathsf{msg}_i, \sigma_i)$.
   - If round number $i = (\ell + 1)$, do:
     - Compute $z = \pi^{\mathsf{Res}}.\mathsf{OUT}(\mathsf{Trans}_{\ell-1}, \overrightarrow{\mathsf{msg}}_\ell, \alpha; \mathsf{r}^{\mathsf{Res}})$.
     - If $z = 0$, output a. Else, output $\perp$.

Figure 1 Description of program C.

Input: For each $i \in [\lambda^2]$, party P_i has input α_i of length $\sqrt{\lambda}$. **Output:** If $(\alpha_1 \oplus \ldots \oplus \alpha_{\lambda^2}) = 0^{\sqrt{\lambda}}$, output 0. Otherwise, output $\perp$.

Figure 2 Description of Functionality f.

B.1.0.3 Hardcore Predicate

Recall that the Goldreich-Levin hardcore predicate [43] for the one-way function g is h. We now define the hardcore predicate H for $\mathcal{G}$ as: $\mathsf{H}(x) = h(a)$, where $x = (a, b)$ such that $|a| = |b|$.

B.2 Security

We conjecture the following about the existence of special-purpose obfuscation.

▶ **Conjecture 9** (Special-purpose obfuscation)**.** *There exists a secure signature scheme and a resettable MPC protocol according to Definition 5 for which, for the class of circuits $\mathsf{C} = \{\mathsf{C}_\lambda\}_{\lambda \in \mathbb{N}}$ where for $\lambda \in \mathbb{N}$, C_λ is depicted in Figure 1, where $a \in \{0,1\}^\lambda, y \in \{0,1\}^{p(\lambda)}, \alpha \in \{0,1\}^{\sqrt{\lambda}}, (\mathsf{SK}, \mathsf{VK}) \in \mathsf{Supp}(\mathsf{Gen}(1^\lambda)), \mathsf{r}^{\mathsf{Res}} \in \{0,1\}^{\ell(\lambda)}$, there exists a virtual black-box obfuscator according to Definition 4.*

▶ **Theorem 10.** *Assuming Conjecture 9 and resettably secure MPC with super-polynomial simulation against malicious adversaries according to Definition 5, Conjecture 8 is false. In particular, assuming Conjecture 9, and either sub-exponential DDH or LWE, Conjecture 8 is false.*

This also implies that the above result holds in the ideal obfuscation model. Here we model obfuscation as an ideal functionality with two interfaces. An "Obfuscate" interface takes as input a program P and outputs a handle $\tilde{P}$. The handle can simply be a counter which is incremented on each invocation. The ideal functionality keeps a list of such tuples $(\tilde{P}, P)$. An "Evaluate" interface takes as input a handle $\tilde{P}$ and an input x, and finds the corresponding tuple $(\tilde{P}, P)$ in the list; if such a tuple exists it outputs $P(x)$ else $\perp$. We rely on the fact that Conjecture 9 holds relative to this oracle and that our counter-example uses the obfuscator in a black-box manner. This gives us the following corollary.

▶ **Corollary 11.** *Assuming sub-exponential DDH or LWE, Conjecture 8 is false in the ideal obfuscation model.*

We now describe the proof of this theorem. First, we prove that H defined above is indeed a hardcore predicate for function $\mathcal{G}$. We observe that for our construction of $\mathcal{G}$ and H, this also automatically proves that $\mathcal{G}$ is a one way function. This is for the following reason: suppose for contradiction that $\mathcal{G}$ is not a one way function. Then, there exists a PPT adversary that, given a value $\mathcal{G}(x = (a, b)) = (\widehat{\mathsf{C}})$, can compute an inverse $x' = (a', b')$ with non-negligible probability. However, observe that since g is an injective one way function, $a' = a$. Further, since $\mathsf{H}(x = (a, b)) = h(a)$, $\mathcal{A}$ can compute $\mathsf{H}(x') = \mathsf{H}(x)$. Thus, given just $\mathcal{G}(x)$, $\mathcal{A}$ can compute $\mathsf{H}(x)$ with non-negligible probability which contradicts the fact that H is a hardcore predicate for function $\mathcal{G}$. Therefore, it suffices to formally show that for every PPT adversary $\mathcal{A}$,

$$\mathsf{Pr}[\mathcal{A}(\mathcal{G}(x)) = \mathsf{H}(x)] = \frac{1}{2} + \mathsf{negl}(\lambda)$$

where the probability is over the randomness of x and $\mathcal{A}$. We do this via the following series of computationally indistinguishable hybrids $\mathsf{Hyb}_0, \ldots, \mathsf{Hyb}_6$. We defer the rest of the proof to the full version of the paper.

B.3 Parallel Repetition Attack

We now describe the counterexample to the dream XOR lemma by setting $t = \lambda^2$. That is, we will construct a PPT adversary $\mathcal{A}$ that, given λ^2 samples of the outputs of the one way function $\mathcal{G}$, can compute the XOR of their respective hardcore bits with respect to predicate H with probability greater than or equal to $2^{-\sqrt{\lambda}}$ thus disproving Conjecture 8. Formally, we construct an adversary $\mathcal{A}$ such that : $\mathsf{Pr}[\mathcal{A}(\mathcal{G}(x_1), \ldots, \mathcal{G}(x_{\lambda^2})) = \mathsf{H}(x_1) \oplus \ldots \oplus \mathsf{H}(x_{\lambda^2})] \geq 2^{-\sqrt{\lambda}}$ where the probability is over the random choices of the values $(x_1, \ldots, x_{\lambda^2})$ and the randomness of the adversary.

B.3.0.1 Adversary's Strategy

Adversary $\mathcal{A}$, given $\{\mathcal{G}(x_i) = (y_i, \widehat{\mathsf{C}}_i)\}_{i \in [\lambda^2]}$, does:

1. Run an execution of the resettable MPC protocol π^{Res} amongst (λ^2) parties for functionality f (Figure 2) using obfuscated programs $\widehat{\mathsf{C}}_1, \ldots, \widehat{\mathsf{C}}_{\lambda^2}$. The protocol messages are forwarded appropriately to all the obfuscations.
2. Abort if any obfuscated program outputs $\perp$. Else, let a_i be the value output by $\widehat{\mathsf{C}}_i$ at the end of the protocol. For each $i \in [\lambda^2]$, compute $h(a_i)$.
3. Output $h(a_1) \oplus \ldots \oplus h(a_{\lambda^2})$.

B.3.0.2 Analysis

For each input $x_i = (a_i, b_i)$, recall that α_i is the first $\sqrt{\lambda}$ bits of $\mathsf{PRG}(b_i)$. For randomly chosen inputs $x_1, \ldots, x_{\lambda^2}$, $Pr[(\alpha_1 \oplus \ldots \oplus \alpha_{\lambda^2}) = 0^{\sqrt{\lambda}}] \geq 2^{-\sqrt{\lambda}}$. Therefore, by the correctness of the obfuscation scheme, the resettable MPC protocol π^{Res} and the signature scheme, it is easy to see that for randomly chosen inputs $x_1, \ldots, x_{\lambda^2}$, for every $j \in [\lambda^2]$, the adversary learns the pre-image a_i with probability $\geq 2^{-\sqrt{\lambda}}$. Thus, $\mathsf{Pr}[\mathcal{A}(\mathcal{G}(x_1), \ldots, \mathcal{G}(x_{\lambda^2})) = \mathsf{H}(x_1) \oplus \ldots \oplus \mathsf{H}(x_{\lambda^2})] \geq 2^{-\sqrt{\lambda}}$ and this completes the proof.