Electronic Colloquium on Computational Complexity, Report No. 20 (2010)

Interactive Locking, Zero-Knowledge PCPs, and Unconditional Cryptography

Vipul Goyal* Yuval Ishai† Mohammad Mahmoody‡ Amit Sahai§

February 18, 2010

Abstract

Motivated by the question of basing cryptographic protocols on stateless tamper-proof hardware tokens, we revisit the question of unconditional two-prover zero-knowledge proofs for **NP**. We show that such protocols exist in the *interactive PCP* model of Kalai and Raz (ICALP '08), where one of the provers is replaced by a PCP oracle. This strengthens the feasibility result of Ben-Or, Goldwasser, Kilian, and Wigderson (STOC '88) which requires two stateful provers. In contrast to previous zero-knowledge PCPs of Kilian, Petrank, and Tardos (STOC '97), in our protocol both the prover and the PCP oracle are efficient given an **NP** witness.

Our main technical tool is a new primitive that we call *interactive locking*, an efficient realization of an unconditionally secure commitment scheme in the interactive PCP model. We implement interactive locking by adapting previous constructions of *interactive hashing* protocols to our setting, and also provide a direct construction which uses a minimal amount of interaction and improves over our interactive hashing based constructions.

Finally, we apply the above results towards showing the feasibility of basing unconditional cryptography on *stateless* tamper-proof hardware tokens, and obtain the following results:

- We show that if tokens can be used to encapsulate other tokens, then there exist unconditional and statistically secure (in fact, UC secure) protocols for general secure computation.
- Even if token encapsulation is not possible, there are unconditional and statistically secure commitment protocols and zero-knowledge proofs for **NP**.
- Finally, if token encapsulation is not possible, then no protocol can realize statistically secure oblivious transfer.

Keywords: Cryptography, Statistical Security, Zero Knowledge, Interactive Proof, Interactive PCP, Commitment Scheme, Oblivious Transfer, Interactive Hashing

*Microsoft Research, India, vipul@microsoft.com

†Technion and UCLA, yuvali@cs.technion.ac.il. Supported in part by ISF grant 1310/06, BSF grants 2004361, 2008411, and NSF grants 0716835, 0716389, 0830803, 0916574.

‡Princeton University, mohammad@cs.princeton.edu. Supported by NSF grants CNS-0627526, CCF-0426582 and CCF-0832797.

§UCLA, sahai@cs.ucla.edu.

ISSN 1433-8092

1 Introduction

What is the minimal amount of trust required for unconditionally secure cryptography? Unconditional cryptography can be based on trusted two-party functionalities such as oblivious transfer [Rab81, Kil88] or noisy channels [CK88], on bounded storage assumptions [Mau92], on the presence of an honest majority [BGW88, CCD88, RB89], or even on the presence of a dishonest majority of *non-communicating* parties [BGKW88]. More recently, there has been a considerable amount of work on cryptographic protocols in which parties can generate and exchange tamper-proof hardware tokens. In this model it was shown that unconditionally secure commitments [MS08] or even general secure two-party computation [GIS$^+$10] are possible, provided that the tokens can be *stateful*. In particular, stateful tokens can erase their secrets after being invoked. The present work is motivated by the goal of establishing unconditional feasibility results for cryptography using *stateless* hardware tokens. This question turns out to be related to the classical question of unconditional multi-prover zero-knowledge proofs, which we revisit in this work. We start with some relevant background.

1.1 Multi-Prover Zero-Knowledge

Since the introduction of zero-knowledge proofs in the seminal work of Goldwasser, Micali, and Rackoff [GMR89], a large body of work has been devoted to understanding the capabilities and limitations of such proofs. A particularly successful line of research studied the power of *statistical zero-knowledge* (SZK) proofs — ones which guarantee that even computationally unbounded verifiers can learn nothing from the interaction with the prover. In contrast to computational zero-knowledge proofs [GMW91], a major limitation of SZK proofs which restricts their usefulness in cryptography is that they seem unlikely to cover the entire class of **NP** [For89, AH91]. The related goal of obtaining any kind of *unconditional* zero-knowledge proofs for **NP**, which do not rely on unproven intractability assumptions, seems as unlikely to be achieved (cf. [OW93]) at least until the elusive **P** vs. **NP** question is resolved.

Motivated by the above goals, Ben-Or, Goldwasser, Kilian, and Wigderson [BGKW88] introduced in 1988 the model of *multi-prover interactive proofs* (MIPs), a natural extension of the standard model of interactive proofs which allows the verifier to interact with two or more non-communicating provers. The main result of [BGKW88] is an unconditional two-prover SZK proof for any language in **NP** (see [LS95, BFL90, DFK$^+$92] for subsequent improvements). A direct cryptographic application suggested in [BGKW88] is that of proving one's identity using a pair of bank cards. We will further discuss these types of applications later.

In a very surprising turn of events, the initial work on zero-knowledge in the MIP model led to a rapid sequence of developments that have literally transformed the theory of computer science. This line of research culminated in the first proof of the PCP Theorem [AS98, ALM$^+$98].

The notion of probabilistically checkable proofs (PCPs) is very relevant to our work. In 1988, Fortnow, Rompel, and Sipser [FRS88] suggested an alternative model for MIPs in which multiple provers are replaced by a single oracle, subsequently called a *PCP oracle* or just a PCP. The difference between an oracle and a prover is that an oracle, like a classical proof, cannot keep an internal state. When a prover is asked multiple queries, the answer to each query can depend on all previous queries, whereas the answer of an oracle to each query must depend on that query alone. The latter difference makes soundness against PCP oracles easier to achieve than soundness against provers, which explains the extra power of PCPs over traditional interactive proofs. However,

as already observed in [BGKW88], the *zero-knowledge* property becomes harder to achieve when converting provers into oracles because oracles have no control over the number of queries made by a dishonest verifier. In particular, if the verifier may query the entire domain of the oracle (as in the case of traditional polynomial-length PCPs) then the oracle can no longer hide any secrets.

The question of replacing zero-knowledge provers by stateless oracles is motivated by practical scenarios in which verifiers can "reset" provers to their initial state, say by cutting off their power supply. (Note that similarly to zero-knowledge provers, zero-knowledge PCP oracles should be *randomized* in the sense that their answer depends both on the query and on a secret source of randomness which is picked once and for all when the oracle is initialized.) This motivation led to a recent line of work on *resettable zero-knowledge*, initiated by Canetti, Goldreich, Goldwasser, and Micali [CGGM00]. The main results from [CGGM00] show that, under standard cryptographic assumptions, there exist resettable (computational) zero-knowledge proofs for **NP**. However, results along this line do not seem relevant to the case of *unconditional* (and statistical) zero-knowledge proofs, which are the focus of the present work.

Zero-knowledge PCPs. The question of *unconditional* zero-knowledge PCPs was studied by Kilian, Petrank and Tardos [KPT97] (improving over previous results implicit in [DFK$^+$92]). Specifically, it is shown in [KPT97] that any language in **NEXP** admits a proof system with a *single* PCP which is statistical zero-knowledge against verifiers that can make any polynomial number of PCP queries (but are otherwise computationally unbounded). However, as expected from proof systems for **NEXP**, the answers of the PCP oracle cannot be computed in polynomial time. This still leaves hope for scaling down the result to **NP** and making the PCP oracle efficient given an **NP** witness. Unfortunately, such a scaled down version presented in [KPT97] has the undesirable side effect of scaling down the zero-knowledge property as well, effectively restricting the number of queries made by a cheating verifier to be much smaller than the (fixed polynomial) entropy of the oracle. Thus, compared to typical feasibility results in cryptography, the results of [KPT97] for **NP** require us to either make an unreasonable assumption about the computational capability of the (stateless) prover, or to make an unreasonable assumption about the limitations of a cheating verifier.

Interactive PCPs. The above state of affairs motivates us to consider the *Interactive PCP* (IPCP) model, which was recently put forward by Kalai and Raz [KR08] and further studied in [GKR08b]. This model can be seen as lying in between the pure PCP model and the pure MIP model, thus aiding us in our quest for a "minimal" model for efficient unconditional zero-knowledge proofs for **NP**. In the IPCP model there is one interactive prover as in the MIP model and one PCP as in the PCP model. The study of IPCPs in [KR08] was motivated by the efficiency goal of allowing *shorter* PCPs for certain **NP** languages than in the traditional PCP model, at the price of a small amount of interaction with a prover. In contrast, our use of the IPCP model is motivated by the *feasibility* goal of obtaining unconditional zero-knowledge proofs for **NP** with polynomial-time prover and PCP oracle. Another difference is that while in the context of [KR08] a PCP is at least as helpful as a prover, the zero-knowledge property we consider is harder to satisfy with a PCP oracle than with a prover (as discussed above). The IPCP model can be made strictly stronger than the MIP model by requiring soundness to hold also with respect to *stateful* PCP oracles. We tackle this stronger variant as well, but we stick to the basic IPCP model by default.

To meaningfully capture zero-knowledge proofs with polynomial-time provers in the IPCP

model, we extend the original IPCP model from [KR08] in two natural ways. First, we allow the PCP to be randomized. Concretely, we assume that both the prover and the PCP are implemented by polynomial-time algorithms with three common inputs: an instance x, a witness w, and a random input r. (This is analogous to earlier models for efficient multi-prover zero-knowledge proofs for **NP**.) The length of both w and r is polynomial in $|x|$. Second, as discussed above, in order to allow the PCP oracle to hide secrets from the verifier we need to use PCP oracles with a super-polynomial query domain, and we restrict cheating verifiers to make (an arbitrary) polynomial number of queries to the oracle, but otherwise allow them to be computationally unbounded. Note, however, that in contrast to the solutions from [KPT97] we cannot use PCP oracles with a super-polynomial entropy since we want our PCP to be efficiently implementable.

This gives rise to the following feasibility question:

> *Are there (efficient-prover) statistical zero-knowledge proofs for* **NP** *in the interactive PCP model?*

1.2 Our Results

We answer the above question affirmatively, presenting an *unconditional* SZK proof for **NP** in the interactive PCP model with efficient prover and PCP oracle. Zero-knowledge holds against cheating verifiers which can make any polynomial (in fact, even sub-exponential) number of PCP queries, but are otherwise computationally unbounded. Our protocol can be implemented in a constant number of rounds. We also show how to get a similar protocol (with a non-constant number of rounds) in the stronger variant of the IPCP model in which a cheating PCP oracle may be stateful, thus strengthening the previous feasibility result from [BGKW88].

Interactive locking. The main technical tool we use to obtain the above results (as well as additional applications discussed below) is a new primitive which we call an *interactive locking scheme* (ILS). This primitive extends in a natural way the notion of non-interactive locking schemes which were defined and implemented in [KPT97]. The original locking primitive can be viewed as a PCP-based implementation of a non-interactive commitment with statistical hiding and binding. Roughly speaking, a locking scheme is an oracle which hides a secret that can later be revealed to the receiver by sending it a decommitment key. Given access to the oracle alone, it is hard for the receiver to learn anything about the secret. However, it is easy for the receiver to become convinced that at most one secret can be successfully decommitted even when the oracle is badly formed.

The locking scheme from [KPT97] requires the oracle to have bigger entropy than the number of queries against which the hiding property should hold. We prove the intuitive fact that such a limitation is inherent, and therefore there is no efficient-oracle non-interactive locking scheme which resists an arbitrary polynomial number of queries. This is because intuitively if the entropy of the oracle is bounded, then either: (1) the receiver is able to learn all the entropy by making a polynomial number of queries, and therefore break the hiding property; or (2) if some entropy is hidden no matter what queries the receiver makes, then a cheating sender is able to create a "fake" oracle that can cheat on this entropy and therefore be opened to any value, breaking the binding property.

This motivates our notion of an *interactive* locking scheme. An ILS is a locking scheme in the IPCP model: the commitment phase can involve, in addition to oracle queries by the receiver, interaction with the sender from whom the secret originated. Here the sender and the oracle play

the roles of the prover and PCP oracle in the IPCP model, respectively. Decommitment still involves a single message from the sender to the receiver. Somewhat surprisingly (and counter to our own initial intuition), we show that interaction can be used to disrupt the intuitive argument above.

We present several constructions of efficient interactive locking schemes. We show how to obtain such schemes from *interactive hashing* — a primitive which was introduced by Naor, Ostrovsky, Venkatesan, and Yung [NOVY98] for the purpose of constructing statistically hiding and *computationally* binding commitment schemes from any one-way permutation (see also [OVY93a, DHRS04, HR07]). The high level idea of the transformation from interactive hashing to ILS is to "implement" a one-way permutation by an oracle which contains a random *point function* (i.e., a function that outputs 0 on all but one random point). To ensure the binding property even when the oracle is badly formed, the receiver should query the oracle on a small number random points to verify that it is not "too far" from a point function. The (black-box) proof of security of the interactive hashing protocol implies (unconditional) proof of security for the ILS.

The above connection allows us to use interactive hashing protocols from the literature for obtaining interactive locking schemes, but leaves open the question of minimizing the amount of interaction with the sender. We resolve this question by presenting a novel direct construction of ILS which requires only a single round of interaction with the sender.

The high level idea behind our single round ILS is as follows. The oracle π constructed by the sender will be the zero function over $\{0,1\}^n$ except for an "interval" of size 2^{cn}: $\pi(x) = 1$ for $a \leq x \leq a + 2^{cn}$ and $\pi(x) = 0$ elsewhere. Depending on whether the sender commits to zero or one, the interval will be planted in the first or second half of the oracle π and reveal a to the receiver in the decommitment phase. By choosing the size of the interval 2^{cn} small: $c < 1$, the receiver will not able to ask any query from the planted interval and find out the committed bit during the commitment phase with a non-negligible probability. But now the sender is able to cheat by planting intervals in both the first and second half of π. The idea is to ask a "challenge" question about the interval in such a way that the prover cannot find a *pair* of planted intervals in the first and second half of π with the same challenge answer. A natural idea is to use a pairwise independent hash function $h\colon \{0,1\}^n \to \{0,1\}^{dn}$ and ask the prover to reveal $h(a)$. The prover is able to plant at most $2^{(1-c)n}$ *separate* intervals in each half of π. Each of the intervals in the first and second half of π will have the same hashes with probability 2^{-dn}. Therefore if $2(1-c) < d$, then with high probability over the choice of h the prover is *not* able to find two intervals with the same hash value $h(a)$ and thus gets committed to a fixed bit. But now the information revealed by $h(a)$ might help the receiver to run a smarter search to find a non-zero point in π and find out the committed bit. We give a specific construction of pairwise independent hash functions where the preimages of any hash value are "scattered" in the domain of the hash function. The scattered-preimages property prevents the receiver to take advantage of knowing $h(a)$ and find out where the interval is planted. Therefore we get both the binding and the hiding properties.

Cryptography using hardware tokens. The above study of zero-knowledge interactive PCPs and interactive locking schemes is motivated by a recent line of research on the capabilities of cryptographic protocols in which parties can generate tamper-proof hardware tokens and send them to each other. Katz [Kat07] shows that, under computational assumptions, general *universally composable* (UC) secure two-party computation [Can01] is possible in this model if the tokens are allowed to be *stateful*, and in particular can erase their secrets after being invoked. It was

subsequently shown that even *unconditional* security is possible in this model, first for the case of commitment [MS08] and then for general tasks [GIS+10]. See [GO96, GKR08a, HL08] and references therein for other applications of stateful tokens in cryptography.

Obtaining similar results using *stateless* tokens turns out to be more challenging. Part of the difficulty stems from the fact that there is no guarantee on the functionality of tokens generated by malicious parties — they may compute arbitrary functions of their inputs and may even carry state information from one invocation to another. It was recently shown in [GIS+10], improving on [CGS08], that any *one-way function* can be used for basing (computationally) UC-secure two-party computation on stateless tokens. More practical protocols which satisfy weaker notions of security were given in [Kol10]. These works leave open the question of obtaining a similar result *unconditionally*, and with *statistical* security. (To get around impossibility results in the plain model, the number of queries to a token should be polynomially bounded, but otherwise malicious parties may be computationally unbounded.) In fact, the constructions from [CGS08, GIS+10, Kol10] may lead to a natural conjecture that achieving statistical security in this setting is impossible, since in these constructions all the "useful information" contained in tokens can be learned by a computationally unbounded adversary using a polynomial number of queries.

However, similar to the case of ILS discussed above, the combination of stateless tokens and interaction turns out to be surprisingly powerful. As already alluded to in [BGKW88], MIP protocols can naturally give rise to protocols in the hardware token model. In our case, we implement the ILS (or IPCP) by having a single sender (prover) create a stateless tamper-proof hardware token which implements the PCP oracle and send it to the receiver (verifier). Applying this to our results, this directly gives rise to the first unconditionally secure commitment protocols and SZK proofs for **NP** using stateless tokens.

We show how this can be extended to general unconditionally secure (in fact, UC-secure) two-party computation if parties are allowed to build tokens which encapsulate other tokens: namely, the receiver of a token A is allowed to build another token B which internally invokes A. The high level idea is the following. By the completeness of oblivious transfer (OT) [Kil88, IPS08], it suffices to realize OT using stateless tokens. This is done as follows. The OT sender's input is a pair of strings (s_0, s_1) and the OT receiver's input is a selection bit b. The OT receiver commits b using an ILS. Applying our best construction, this involves sending a token A to the OT sender and responding to a random challenge message received from the OT sender. The OT sender now prepares and sends to the receiver a token B with the following functionality. Token B accepts a selection bit b along with a corresponding decommitment message. It checks that the decommitment is valid (this involves invocations of the token A, which token B encapsulates) and then returns the string s_b if decommitment was successful. The binding property of the ILS guarantees that the OT receiver can learn at most one string s_b. The hiding property of the ILS guarantees that the sender cannot learn b.

Interestingly, we also show a matching negative result: if token encapsulation is not allowed, then statistically secure OT is impossible (regardless of UC-security). The proof of this negative result employs the recent notion of accessible entropy from [HRVW09] and the high level idea is as follows. One way to explain why unconditional OT (even over random inputs) is not possible in the standard model of interaction is that at any time during the interaction a computationally unbounded party (e.g. the OT receiver R) is able to sample from the space of its randomness r_R conditioned on what the other party (i.e. OT sender S) knows about R's computation. The latter information is captured by the transcript τ of the protocol. Therefore if at the end of

the interaction the distribution of r_R conditioned on τ is not revealing the receiver's bit b, the receiver can sample from $(r_R \mid \tau)$ and find out both of the strings (s_0, s_1). If such sampling is done efficiently [HRVW09] says that the protocol has accessible entropy. In the token model, however, the exchanged information about Alice and Bob is not symmetric and Bob might not know which queries Alice has asked from Bob's tokens. Similar to the standard model we define a protocol (A, B) in the token model to have accessible entropy if the parties can (information theoretically) sample their history of computation so far only conditioned on the *other* party's view. Similar to the standard model, accessing the entropy of a protocol for OT in the token model can be used to break its security for the benefit of either of the parties. We prove the following technical lemma: For any protocol (A, B) in the stateless token model, there is another protocol (A', B') where **(1):** the parties in (A', B') emulate the original protocol (A, B), **(2)** (A' and B' also learn more information about the tokens they hold along the emulation of (A, B) but still ask $\text{poly}(n)$ number of queries, and more importantly **(3):** almost all the entropy of A' and B' in (A', B') is accessible. This lemma, together with the necessity of inaccessible entropy for the possibility of OT proves that unconditional OT does not exist in the stateless token.

Organization. In Section 2, we define the notions of zero-knowledge IPCPs and ILS, and show how to use ILS to build unconditional zero-knowledge IPCPs for **NP**. We also show that interaction is required for efficient ILS. In Section 3, we show how to construct ILS. In Section 6, we show the implications of our work on (unconditionally secure) cryptography with tamper-proof hardware tokens.

2 Preliminaries

2.1 Basics about Probabilities

By $\mathsf{sup}(X)$ we mean the support set of the random variable X. By $x \leftarrow X$ we mean the process of sampling x according to the distribution of the random variable X. By $(X \mid Y)$ we denote the random variable X conditioned on the random variable Y.

Definition 2.1. For the random variables X and Y defined over the set U (i.e. $\mathsf{sup}(X) \bigcup \mathsf{sup}(Y) \subseteq U$) we let $\mathsf{SD}(X, Y)$ denote their *statistical distance* defined as: $\mathsf{SD}(X, Y) = \max_{E \subseteq U} |\Pr[X \in E] - \Pr[Y \in E]|$. We also use $X \approx_\alpha Y$ and call X and Y α-close whenever $\mathsf{SD}(X, Y) \leq \alpha$.

It can be verified that the the statistical distance is the maximum advantage by which a statistical test (i.e., an algorithm A which outputs in $\{0, 1\}$) can distinguish between two random variables X and Y: $\mathsf{SD}(X, Y) = \max_A |\Pr[A(X) = 1] - \Pr[A(Y) = 1]|$. Also, the triangle inequality holds for the statistical distance: $\mathsf{SD}(X, Y) + \mathsf{SD}(Y, Z) \geq \mathsf{SD}(X, Z)$.

Lemma 2.2. *Let $X = (X_1, X_2)$ and $Y = (Y_1, Y_2)$ be random variables defined over the set $U_1 \times U_2$ and suppose $\mathsf{SD}(X, Y) \leq \epsilon$. Let $Z = (Z_1, Z_2)$ be the random variable defined over $U_1 \times U_2$ as follows: Z_1 is sampled according to X_1 (defined by the distribution of X), and then Z_2 is sampled according to $(Y_2 \mid Y_1)$ conditioned on $Z_1 = Y_1$. Then it holds that $\mathsf{SD}(Y, Z) \leq \epsilon$ and therefore by the triangle inequality $\mathsf{SD}(X, Z) \leq 2\epsilon$.*

Proof. Let $\mathsf{SD}(Y, Z) \geq \epsilon$ and let A be the algorithm that distinguishes between Y and Z with advantage $\geq \epsilon$. Then one can distinguish between X and Y with advantage more than ϵ as well

by the following algorithm. Given the input $W = (W_1, W_2)$ (which is either sampled according to X or Y) we take the first component W_1 and sample Y_2 conditioned on $Y_1 = W_1$ according to the distribution of Y. We then apply the test A over (W_1, Y_2). It is easy to see that the new test distinguishes between Y and X as well as A does between Y and Z. ■

Lemma 2.3 (Lemma 6.4 of [IR89]). *Let $Z_1, \ldots, Z_i, \ldots$ be any sequence of random variables determined by a finite underlying random variable X, let E be any event for random variable X, and let $0 \leq p \leq 1$. Let B_j be the event that $\Pr_X[E(X) \mid Z_1, \ldots, Z_j] \geq p$, and let $B = \bigvee_j B_j$. Then it holds that $\Pr_X[E[X] \mid B] \geq p$.*

2.2 Interactive Algorithms

We assume that the reader is familiar with the notion of interactive Turing Machines introduced in [GMR89] which here we call *interactive algorithms*. An interactive *protocol* is defined by tow interactive algorithms A and B where each of A and B may have its own private input, private random tape, and output. By $\langle A, B\rangle(x)$ we denote the output of B when A and B interact on the common input x. We take the output 1 to denote an "accept" and the output 0 to denote a "reject". When x is clear from the context we might omit it from the notation (e.g., letting $\langle A, B\rangle$ denote $\langle A, B\rangle(x)$).

By an *efficient* algorithm we mean one which runs in polynomial time over its input length. We do not assume the interactive algorithms to be necessarily efficient unless explicitly mentioned. We always let n denote the length of the input x which will also serve as the security parameter when A and B are efficient.

A *round* consists of a message from A followed by another message from B assuming that A starts the protocol. The round complexity of the protocol (A, B) is the maximum number of rounds that A and B interact as a function of the input length n.

The *view* of an interactive algorithm consists of its input, its randomness and the answers received from the other interactive algorithm participating in the protocol.

We use the notation $\widehat{A}$ to denote a possibly malicious interactive algorithm participating in a protocol instead of the honest interactive algorithm A.

Although one of the main applications of the interactive protocols is to use them as a "proof system" [GMR89], here we define their basic properties in a more abstract level so that we can use them in the next section in the extended model of Interactive PCPs.

Definition 2.4 (Properties of interactive protocols). Let (P, V) be an interactive protocol composed of a prover P and a verifier V. We define the following properties for (P, V).

- **Soundness:** (P, V) is $(1 - \delta)$-sound for the input x, if for every prover $\widehat{P}$ it holds that $\Pr[\langle \widehat{P}, V\rangle(x) = 1] \leq \delta$.
- **Statistical zero-knowledge (SZK):** (P, V) is $\epsilon(n)$-SZK for the language L if there is a simulator Sim that gets *oracle access*[1] to an arbitrary (unbounded) verifier $\widehat{V}$, runs in time $\mathrm{poly}(n)$, and *if* $x \in L$ then $\mathsf{Sim}^{\widehat{V}}$ produces a view for $\widehat{V}$ which is $\epsilon(n)$-close to the view of $\widehat{V}$ when interacting with P on the input x. A query $q_{\mathsf{sim}} = (r_V, a_1, \ldots, a_i)$ of the simulator Sim to the oracle $\widehat{V}$ consists of the randomness r_V for $\widehat{V}$ and a set of first i messages of the prover $a_1, \ldots, a_k$ to the verifier. The answer $a_{\mathsf{ver}} = (q_1, \ldots, q_{i+1})$ returned by the oracle $\widehat{V}$ consists of the first $i+1$

[1] All the zero-knowledge constructions in this paper use *black-box* simulators.

messages of the verifier $\widehat{V}$ when r_V is used as the randomness and $a_1, \ldots, a_k$ are received from the prover. Thus the simulator Sim, in general, can "rewind" $\widehat{V}$ by asking queries of the form $q_{\mathsf{sim}} = (r_V, a_1, \ldots, a_i)$ and $q'_{\mathsf{sim}} = (r_V, a_1, \ldots, a_{i-1}, a'_i)$. The simulator Sim is called *straight-line* if its queries are of the following "incremental" form: $(r_V), (r_V, a_1), \ldots, (r_V, a_1, \ldots, a_i)$ and the output of $\mathsf{Sim}^{\widehat{V}}$ is the last query of Sim: $(r_V, a_1, \ldots, a_j)$. It is easier to think of a straight-line simulator as an interactive algorithm which simply interacts with the verifier $\widehat{V}$ and outputs the view of $\widehat{V}$ in this interaction.

Note that the soundness is only a property of the verifier V while zero-knowledge is only a property of the prover P.

Inefficient verifiers with large randomness. To prove the zero-knowledge for inefficient verifiers $\widehat{V}$ with super-polynomially long randomness, we slightly change the simulator's behavior of Definition 2.4 as follows. We let the oracle $\widehat{V}$ to choose the randomness r_V uniformly at random (hidden from the simulator) and the simulator's queires would only consist of prover's messages $q_{\mathsf{sim}} = (a_1, \ldots, a_i)$. Also the simulator's output will be the last query of the simulator $q_{\mathsf{sim}} = (a_1, \ldots, a_j)$ joint with the randomness r_V chosen by the verifier (and yet $(r_V, a_1, \ldots, a_j)$ should be ϵ-close to the view of $\widehat{V}$ when interacting with P). All of our simulators in this paper will be of this form to let us handling arbitrarily inefficient verifiers.

Definition 2.5 (Composition of interactive protocols)**.** Let $(P_1, V_1), \ldots, (P_k, V_k)$ be k interactive protocols. By $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ we mean the *sequential composition* of $(P_1, V_1), \ldots, (P_k, V_k)$ defined as follows.

- Let x be the common input.
- V_{seq} uses independent random coins $r_{V_1}, \ldots, r_{V_k}$ for $V_1, \ldots, V_k$ and P_{seq} uses independent coins $r_{P_1}, \ldots, r_{P_k}$ for $P_1, \ldots, P_k$.
- P_{seq} and V_{seq} first interact according to the protocol (P_1, V_1) over the input x, and after that interact according to the protocol (P_2, V_2) (over the asme input x) and so on.
- At the end V_{seq} rejects if any of V_i's reject.

By $(P_{\mathsf{par}}, V_{\mathsf{par}})$ we mean the *parallel composition* of $(P_1, V_1), \ldots, (P_k, V_k)$, which is defined similar to $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ with the difference that $V_1, \ldots, V_k$ interact with the prover P_{par} in a *round-synchronized* way. Namely in the j'th round of the interaction, V_{par} sends the queries $q_1^j, \ldots, q_k^j$ to the prover P_{par} where q_i^j is the j'th query of V_i (and $q_i^j = \perp$ if V_j's interaction finishes before the j'th round). Then V_{seq} receives k answers $a_1^j, \ldots, a_k^j$ and V_i uses a_i as its answer to continue its execution.[2]

Both $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ and $(P_{\mathsf{par}}, V_{\mathsf{par}})$ are special cases of *concurrent composition* $(P_{\mathsf{con}}, V_{\mathsf{con}})$ in which the verifier runs $V_1, \ldots, V_k$ and the prover runs $P_1, \ldots, P_k$ in a way that V_i will be interacting with P_i in the i'th "sessions". There is not any round synchronization enforced across different sessions. The honest verifier V_{con} will send the next query q_i^j (i.e. the j'th query of V_i) when she receives the answer a_i^{j-1} from P_i in the i'th session, but a cheating verifier $\widehat{V_{\mathsf{con}}}$ might *delay* sending such query

[2]To define the parallel composition, we do not necessarily assume $(P_1, V_1), \ldots, (P_k, V_k)$ to have equal round complexity, and the round complexity of $(P_{\mathsf{par}}, V_{\mathsf{par}})$ will be the maximum of the round complexity of $(P_1, V_1), \ldots, (P_k, V_k)$.

to gather more answers in different sessions before continuing. On the other hand the honest prover P_{con} will send the answer a_i^j back whenever he receives the query q_i^j from V_{con}, but a cheating prover $\widehat{P_{\mathsf{con}}}$ might delay sending such answer to gather more queries in different sessions before continuing.

By using simple "delay policies", both the prover and the verifier (when the other party is honest) are able to force a concurrent composition to behave like a parallel or sequential composition.

The inputs in composed protocols. In all variants of compositions defined above (including the concurrent composition), the same input x is used in all the sub-protocols. More general definitions of concurrent composition allow different inputs to be used in different instances of the protocol, but our definition still generalizes the parallel and sequential definitions and is enough for our purposes. Moreover, the properties of the concurrent composition described in Lemma 2.13 below (in particular the SZK property) hold in the more general form of concurrent composition with different inputs (as long the inputs are fixed at the beginning of the execution of the system.)

Lemma 2.6 (Properties of composition of interactive protocols)**.** *Let $(P_1, V_1), \ldots (P_k, V_k)$ be k interactive protocols, and let $(P_{\mathsf{seq}}, V_{\mathsf{seq}}), (P_{\mathsf{par}}, V_{\mathsf{par}}), (P_{\mathsf{con}}, V_{\mathsf{con}})$ be, in order, their sequential, parallel, and concurrent compositions. It holds that:*

1. **Soundness [BM88]:** *If (P_i, V_i) is $(1 - \delta_i)$-sound over the input x for all $i \in [k]$, then $(P_{\mathsf{con}}, V_{\mathsf{con}})$ (and in particular $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ and $(P_{\mathsf{par}}, V_{\mathsf{par}})$) will have soundness $1 - \prod_i \delta_i$.*
2. **SZK [KLR06]:** *If (P_i, V_i) is ϵ_i-SZK for the language L with a straight-line simulator for all $i \in [k]$, then $(P_{\mathsf{con}}, V_{\mathsf{con}})$ (and in particular $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ and $(P_{\mathsf{par}}, V_{\mathsf{seq}})$) will be $(\sum_i \epsilon_i)$-SZK for L with a straight-line simulator.*

Part 1 of Lemma 2.6 is proved in [BM88] for the case of parallel composition, but the proof extends to the concurrent composition as well. The proof of Part 2 of Lemma 2.13 follows by a standard hybrid argument similar to the proof of Part 2 in Lemma 2.17 below.

2.3 Oracle Algorithms

We use the standard notation A^{π} to denote an *oracle algorithm* A accessing the oracle π. For the oracles $\pi_1, \ldots, \pi_k$, by $\pi = (\pi_1 | \ldots | \pi_k)$ we mean their *combined* oracle defined as follows: Given the query (i, q), π answers as $\pi(i, q) = \pi_i(q)$. We allow an oracle algorithm A^{π} to ask multiple oracle queries from π in one *round* of queries.

By a malicious *stateful* oracle $\widehat{\pi}$ we mean an interactive algorithm whose honest behavior is like an oracle. In other words, given a round of queries $q_1, \ldots, q_k$, π returns $\pi(q_1), \ldots, \pi(q_k)$, but a malicious stateful oracle $\widehat{\pi}$ accessed by the oracle-algorithm A can respond the queries similar to a malicious interactive algorithm interacting with A. That is, $\widehat{\pi}$'s answer to a query q can depend on the common input x, or the previous queries of A asked from $\widehat{\pi}$, or other queries being asked in the same round as q.

Definition 2.7 (Parallel composition of oracle algorithms)**.** Let $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ be k oracle algorithms with access to k oracles $\pi_1, \ldots, \pi_k$ where for all $i \in [k]$ A_i asks at most t rounds of oracle queries from π_i. We define $A_{\mathsf{par}}^{\pi_{\mathsf{par}}}$ the *parallel composition* of $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ as follows.

- The oracle $\pi_{\mathsf{par}} = (\pi_1 | \ldots | \pi_k)$ is the combined oracle of $\pi_1, \ldots, \pi_k$.

- $A_{\mathsf{par}}^{\pi_{\mathsf{par}}}$ emulates $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ in parallel as follows. Let q be one of the queries of A_i in its j'th round of queries to π_i. A_{par} asks (i, q) from π_{par} in its j'th round of queries and returns the answer to A_i.
- $A_{\mathsf{par}}^{\pi_{\mathsf{par}}}$ rejects if any of the emulated algorithms $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ reject.

2.4 Interactive PCPs

Interactive PCPs (Definition 2.8 below) were first introduced in [KR08] and combine the notion of oracle algorithms with interactive algorithms. Similar to the previous section, here also we define IPCPs in a general way, not only for the purpose of a proof system, but rather as a model of interaction consisting of interactive algorithms and a prover.

Definition 2.8. (Adapted from [KR08]) An *interactive probabilistically checkable proof* (IPCP) $\Gamma = (P, \pi, V)$ consists of an interactive algorithm P (the *prover*), an oracle π (the *PCP oracle*), and an interactive algorithm V (the *verifier*) such that:

- P and π share common randomness r_P, and V is given the randomness r_V.
- P, π, and V will be given an input x of length $|x| = n$. P and π may also receive a common private input w.[3]
- The PCP oracle π is a function of (r_P, x, w, q) where q is a query of the verifier V. Since (r_P, x, w) is fixed at the beginning of the protocol, we might simply use $\pi(q)$ to denote the answer to the query q.
- P and V^{π} engage in an interactive protocol during which V can query the PCP oracle π and at the end V accepts or rejects.

By an *efficient* IPCP we mean one in which the prover P, the PCP oracle π, and the verifier V run in polynomial time over the input length $|x| = n$.

Definition 2.9 (Properties of IPCPs)**.** Let $\Gamma = (P, \pi, V)$ be an IPCP. We define the following properties for Γ.

- **Soundness:** Γ is $(1-\delta)$-sound for input x, if for every prover and oracle $(\widehat{P}, \widehat{\pi})$ it holds that $\Pr[\langle \widehat{P}, V^{\widehat{\pi}} \rangle(x) = 1] \leq \delta$.
- **Adaptive-soundness:** Γ is $(1-\delta)$-adaptively-sound for input x, if for every prover $\widehat{P}$ and every *stateful* oracle $\widehat{\pi}$ it holds that $\Pr[\langle \widehat{P}, V^{\widehat{\pi}} \rangle(x) = 1] \leq \delta$.
- **Statistical zero-knowledge (SZK):** The SZK property is defined as an extension to the SZK property of interactive protocols (Definition 2.4) with respect to a language L. Since all of our simulators in this paper are straight-line, for sake of simplicity here we only describe how to extend the definition to SZK of IPCPs for straight-line simulators.
  - The straight-line simulator interacts with a (potentially malicious) verifier $\widehat{V}$, while the simulator Sim receives *all* the queries of the the verifier (including both the queries asked from the prover and from the oracle) and responds to them.

[3] For example when (P, π) are efficient and $L \in \mathbf{NP}$, w could be a witness for $x \in L$.

– Since an unbounded verifier can ask arbitrary number of queries from its oracle, here we put a bound u on the number of oracle queries asked by $\widehat{V}$. More formally we say that Γ is $(u(n), \epsilon(n))$-SZK (with a straight-line simulator), if there is a simulator described as above such that for any $v \leq u$, if $\widehat{V}$ asks at most v oracle queries, then Sim runs in time $\mathrm{poly}(n, v)$ and produces a view for $\widehat{V}$ which is ϵ-close to the view of $\widehat{V}$ when interacting with (P, π).

Note that when $u(n)$ is super-polynomial, Definition 2.10 implies the standard notion of zero-knowledge against polynomial-time verifiers.

Definition 2.10 (SZK-IPCP for languages)**.** We say that $\Gamma = (P, \pi, V)$ is an SZK-IPCP for the language L with SZK $(u(n), \epsilon(n))$ and (adaptive) soundness $1 - \delta(n)$ if the following holds:

- **Completeness:** If $x \in L$, then $\Pr[\langle P, V^{\pi} \rangle(x) = 1] = 1$.
- **(Adaptive) soundness:** Γ has (adaptive) soundness $1 - \delta$ if for all $x \notin L$, the verifier V is $(1 - \delta(n))$-(adaptively)-sound. Namely for all provers $\widehat{P}$ and (stateful) oracles $\widehat{\pi}$ it holds that $\Pr[\langle \widehat{P}, V^{\widehat{\pi}} \rangle(x) = 1] \leq \delta(n)$.
- **Statistical zero-knowledge (SZK):** Defined according to the definition of SZK in Definition 2.9.

We simply call Γ an SZK-IPCP for L with (adaptive) security u, if Γ is $(1-1/u)$-(adaptively)-sound and $(u, 1/u)$-SZK.

Round complexity of IPCPs. By the *round complexity* of an IPCP we mean the number of rounds of interaction between the verifier and the *prover* (and not the PCP oracle) where each round consists of a message from the verifier followed by a message from the prover. In particular, the round i for the verifier starts *after* the $i-1$'th message of the prover is sent and ends when the i'th query of the verifier to the prover is sent (which will be followed be a message from the prover). Thus in each round the verifier behaves like an oracle-algorithm. The reasons to only consider the interaction with the prover as a factor in round complexity are as follows.

1. In our constructions of SZK-IPCPs, the number of rounds of oracle queries is either equal to one (Theorem 3.1) or is bounded by the number of rounds of interaction with the prover (Theorem 3.2). On the other hand our negative results (Part 2 of Theorem 4.1) is regardless of number of rounds of oracle queries.
2. Regarding the application of IPCPs to the stateless hardware token model, the queries from the PCP oracle are asked *locally* (as opposed to the queries from the prover which are sent to a remote party). Therefore the number of rounds of interaction with the prover is a more meaningful measure of the efficiency of the system.

Let Γ be an IPCP with j rounds. We can always decompose the verifier's randomness into $r_V = (r_V^1, \ldots, r_V^j)$ so that for the first i rounds only the first i parts $(r_V^1, \ldots, r_V^i)$ are used by the verifier. A trivial decomposition is to let $r_V^1 = r_V$ and r_V^i be the empty string for $i \geq 2$, but that might not be the "natural" decomposition of the randomness.

Definition 2.11 (Public-coin IPCPs)**.** By a public-coin IPCP we mean an IPCP where the prover gets to see the coin tosses of the verifier all along the interactions. Namely the verifier's randomness is decomposable into $r_V = (r_V^1, \ldots, r_V^j)$ such that r_V^i is publicly reviled (to the prover) at the beginning of round i and $(r_V^1, \ldots, r_V^i)$ is used by the verifier in round i.

Comments about public-coin IPCPs.

- As a mental experiment we can always assume that the whole randomness r_V is chosen at the beginning and it is only reviled part by part to the prover and is used by the verifier.
- Without loss of generality we can always assume that r_V^i is the i'th message of the verifier to the prover. That is because the prover knows the content of the oracle π and thus can determine the verifier's query at the end of round i by only knowing $(r_V^1, \ldots, r_V^i)$. Despite this fact, it might still be conceptually simpler to describe the verifier's query in other ways.
- Being public-coin is part of the definition of the model and it might change the soundness of an IPCP. Namely, and IPCP might be 0.99-sound when considered as a (regular) IPCP over an input, but only 0.01-sound when considered as public-coin IPCP.
- When considering adaptive soundness (against stateful oracles), the public-coin IPCP model is only as strong as a two-party (i.e. single prover) interactive protocol. The reason is that the prover and the oracle both get full information about the verifier's messages to both of them and thus can play as a unified party. (For the same reason fully-public-coin multi-prover proof systems are only a special case of single prover proof systems.)

Now we define several forms of composition of IPCPs by extending the corresponding compositions of interactive algorithms (Definition 2.5) to the IPCP model. The new component in the IPCP model compared to the model of interactive algorithms is the oracle. The oracle in all the compositions of IPCPs discussed here is simply the combined oracle of the IPCPs being composed. This way the i'th emulated verifier will ask its oracle queries from the i'th oracle by adding a prefix i to its queries and asking them from the combined oracle.

Definition 2.12 (Composition of IPCPs)**.** Let $\Gamma_1 = (P_1, \pi_1, V_1), \ldots, \Gamma_k = (P_k, \pi_k, V_k)$ be k IPCPs. Let $\pi = (\pi_1 | \ldots | \pi_k)$ be the oracle combination of $\pi_1, \ldots, \pi_k$, and let U_i be the modified version of V_i which asks its oracle queries from π rather than π_i by adding the prefix i and asking (i, q) from π. We define the sequential $\Gamma_{\mathsf{seq}} = (P_{\mathsf{seq}}, \pi_{\mathsf{seq}}, V_{\mathsf{seq}})$, parallel $\Gamma_{\mathsf{par}} = (P_{\mathsf{par}}, \pi_{\mathsf{par}}, V_{\mathsf{par}})$ and concurrent $\Gamma_{\mathsf{con}} = (P_{\mathsf{con}}, \pi_{\mathsf{con}}, V_{\mathsf{con}})$ composition of $\Gamma_1, \ldots, \Gamma_k$, in order, as the sequential, parallel, and concurrent composition of $(P_1, U_1^\pi), \ldots, (P_k, U_k^\pi)$ (note that $\pi_{\mathsf{seq}} = \pi_{\mathsf{par}} = \pi_{\mathsf{con}} = \pi$). In case of parallel composition the emulated verifiers ask their oracles queries also in parallel. Namely, let U_i^j denote the j'th round of the oracle algorithm U_i (i.e., *after* it asks its $(j-1)$'th query and *before* it asks its j'th query from the prover P_i). Similarly let V_{par}^j denote the j'th round of the oracle algorithm V_{par}. The oracle algorithm V_{par}^j will be the parallel composition of the oracle algorithms $U_1^j, \ldots, U_k^j$ according to Definition 2.7.

The definition above describes the *honest* behavior of the prover, oracle, and the verifier in various forms of composition. The way malicious parties can behave in such compositions is determined both by the definition of the composition and the IPCP model. In particular, although in all forms of composition the emulated V_i asks its oracle queries from π by adding a prefix i (simulating

the query being asked from π_i), since the oracle π is accessible by the verifier all along, a malicious verifier is allowed to ask *any* query from the oracle π. Also in the case of concurrent composition, it might be useful for a malicious *stateful* oracle to gather more queries from the verifier before responding to them.

Note that similar to the case of interactive algorithms, the soundness and SZK of the concurrent composition imply those properties (with the same parameters) for the sequential and parallel compositions.

Lemma 2.13 (Properties of composition of IPCPs)**.** *Let $\Gamma_1 = (P_1, \pi_1, V_1), \ldots, \Gamma_k = (P_k, \pi_k, V_k)$ be k IPCPs, and let $\Gamma_{\mathsf{seq}}, \Gamma_{\mathsf{par}}$ and Γ_{con} be in order their sequential, parallel, and concurrent compositions. It holds that:*

1. **Soundness:** *If Γ_i is $(1-\delta_i)$-sound for input x for all $i \in [k]$, then Γ_{con} (and in particular Γ_{seq} and Γ_{par}) will have soundness $1 - \prod_i \delta_i$ over the input x.*
2. **Adaptive-soundness:** *If Γ_i is $(1-\delta_i)$-adaptively-sound for input x for all $i \in [k]$, then Γ_{seq} will have adaptive-soundness $1 - \prod_i \delta_i$ over the input x.*
3. **SZK:** *If (P_i, π_i, V_i) is (u, ϵ_i)-SZK for the language L with a straight-line simulator for all $i \in [k]$, then Γ_{con} (and in particular Γ_{seq} and Γ_{par}) will be $(u, \sum_i \epsilon_i)$-SZK for the language L with a straight-line simulator.*

In the case of sequential composition of interactive protocols it was not necessary for the simulator to be straight-line to get Part 2 of Lemma 2.6, but as we will see in the proof of Lemma 2.13, in the case of IPCPs since the verifier is allowed to query the oracle whenever she wants, we need the simulator to be straight-line even in the case of sequential composition.

Proof. (of Lemma 2.13)

Soundness. Let $\widehat{P}$ be an arbitrary prover and $\widehat{\pi} = (\widehat{\pi_1}| \ldots |\widehat{\pi_k})$ an arbitrary oracle where $\widehat{\pi_i}$ is the oracle accessed by the emulation of V_i in V_{con}. We claim that $\Pr[\langle \widehat{P}, V^{\widehat{\pi}} \rangle = 1] \leq \prod_i p_i$ and the reason is as follows. Let $V_i' = V_i^{\widehat{\pi_i}}$ be the interactive algorithm where the oracle $\widehat{\pi_i}$ is hardwired into the algorithm V_i (note that V_i' is inherently inefficient). Now the interactive algorithm $V' = V_{\mathsf{con}}^{\widehat{\pi}}$ is the same as the concurrent composition of $V_1', \ldots, V_k'$. The crucial point is that the interactive algorithm V_i' is well-defined regardless of which other interactive algorithms are run in parallel. That is because π_i is a fixed *oracle* and its answers only depend on the queries that V_i is asking from it (i.e., π_i's answers do not change depending on other queries asked by other V_j's). Let $\delta_i' = \max_P \Pr[\langle P, V_i' \rangle = 1]$, then by the soundness hypothesis it holds that

$$\delta_i' = \max_P \Pr[\langle P, V_i' \rangle = 1] \leq \max_{P,\pi} \Pr[\langle P, V_i^{\pi} \rangle = 1] \leq \delta_i.$$

Therefore by Part 1 of Lemma 2.6, it holds that

$$\Pr[\langle \widehat{P}, V_{\mathsf{con}}^{\widehat{\pi}} \rangle = 1] \leq \max_P \Pr[\langle P, V' \rangle = 1] \leq \prod_i \delta_i' \leq \prod_i \delta_i.$$

Adaptive-soundness. Suppose the sequential executions of $\Gamma_1, \ldots, \Gamma_k$ do not lead to a reject. Then by the $(1-\delta_i)$-adaptive-soundness of Γ_{i+1}, V_{i+1} will accept with probability at most δ_i regardless of what has happened in the previous interactions and even if $\widehat{\pi}_{i+1}$ is stateful. Thus the probability that *all* of V_i's accept is bounded by $\prod_i \delta_i$.

SZK. For an arbitrary IPCP $\Gamma = (P, \pi, V)$ let (P^u, V) be the *interactive protocol* in which P^u answers both the queries of V from the prover P and up to u oracle queries of V asked from π. If P, π and P^u are honest, then from the point of view of a verifier V who asks at most u oracle queries there is no difference between interacting with P^u or (P, π). Therefore (P^u, V) is ϵ-SZK for any language L (with a straight-line simulator) if and only if Γ is (u, ϵ)-SZK for L (with a straight-line simulator). Now by Part 2 of Lemma 2.6 and using the fact that Γ_i is (u, ϵ_i)-SZK, it follows that (P_i^u, V_i) is ϵ_i-SZK for L. Let $(P_{\mathsf{con}}^u, V_{\mathsf{con}})$ be the concurrent composition of (P_i^u, V_i)'s for $i \in [k]$. By Part 2 of Lemma 2.6 $(P_{\mathsf{con}}^u, V_{\mathsf{con}})$ is $(\sum_i \epsilon_i)$-SZK for L. On the other hand, if we restrict a verifier V to ask at most u queries from a combined oracle $\pi = (\pi_1, \ldots, \pi_k)$, the number of queries that V can ask from the i'th sub-oracle π_i (by adding the prefix i to the query) is also bounded by u. Therefore the $(\sum_i \epsilon_i)$-SZK property of $(P_{\mathsf{con}}^u, V_{\mathsf{con}})$ implies that Γ_{con} is $(u, \sum_i \epsilon_i)$-SZK for L.

We shall point out that since in the sequential composition Γ_{seq} a cheating verifier $\widehat{V_{\mathsf{seq}}}$ can ask arbitrary queries to the oracle π_{seq} at any time during the interaction, therefore the behavior of $\widehat{V_{\mathsf{seq}}}$ in Γ_{seq} does *not* necessarily correspond to a cheating verifier interacting in the sequential composition of (P_i^u, V_i)'s. However any cheating verifier $\widehat{V_{\mathsf{con}}}$ participating in Γ_{con} (and in particular participating in Γ_{seq} and Γ_{par}) always corresponds to a cheating verifier interacting in a concurrent composition of (P_i^u, V_i)'s. ■

2.5 Interactive Locking Schemes

An *Interactive locking scheme* is a commitment scheme implemented in the IPCP model. A similar definition appeared in [KPT97] without the interaction (i.e. only with an oracle), but as we will see in Theorem 4.1 *non-interactive* locking schemes are inherently inefficient and therefore not as applicable in cryptographic settings.

Definition 2.14 (Interactive locking scheme)**.** Let $\Lambda = (S, \sigma, R)$ be an *efficient* IPCP (where we call S the sender, σ the locking oracle and R the receiver) of the following form:

- The common input is of the form 1^n where n is the security parameter.
- (S, σ) receive a private input $w \in W_n$ which is called the committed message as well as the private randomness r_S. The receiver R gets the randomness r_R.
- The receiver R gets oracle access to the locking oracle σ and R^σ interacts with S in two phases: **(1)** commitment phase and **(2)** decommitment phase. The decommitment phase consists of only one message from the sender S to the receiver R which includes the committed message w and the private randomness r_S used by S. Following this message the receiver R (perhaps after asking more queries from the oracle σ) accepts or rejects.
- Completeness: For any $w \in W_n$ if all parties are honest the receiver accepts with probability one:
$$\Pr[\langle S(w), R^{\sigma(w)} \rangle(1^n) = 1] = 1$$
where the probability is over the random seeds r_S and r_R.

Then Λ is called an *interactive locking scheme* (ILS) for the message space W_n and if $W = \{0, 1\}$, we call Λ a *bit*-ILS. When n is clear from the context we might simply use W rather than W_n to denote the message space.

Definition 2.15 (Properties of ILS's)**.** Let $\Lambda = (S, \sigma, R)$ be an ILS. We define the following properties for Λ.

- **Binding:** We define Λ to be $(1-\delta)$-binding if for any sender $\widehat{S}$ and any oracle $\widehat{\sigma}$, with probability at least $1-\delta$ over the interaction of the commitment phase there is at most one possible w such that $\widehat{S}$ can decommit to successfully. More formally, let R' be the interactive algorithm which does the following.

  1. Choose a randomness r_R for the receiver R.
  2. Run the commitment phase of R in Λ.
  3. R' reads all the queries of the oracle $\widehat{\sigma}$ that are of length bounded by the running time of R (this step is inherently inefficient).
  4. Knowing the content of the oracle $\widehat{\sigma}$, R' *accepts* if there exist (w_1, r_1) and (w_1, r_2) such that $w_1 \neq w_2$ and R would accept both of (w_1, r_1) and (w_1, r_2) in the decommitment phase.

  Λ is $(1-\delta)$-binding over the message space W_n iff (S, σ, R') is $(1-\delta)$-sound over the common input $x = 1^n$.

- **Hiding:** Let $\widehat{R}$ be any malicious receiver who asks at most u oracle queries from σ, and let τ_w be the random variable which consists of the transcript of the interaction of R with (S, σ) till the end of the commitment phase when the committed message is $w \in W$. Λ is (u, ϵ)-hiding if for every such malicious receiver $\widehat{R}$ and every $\{w_1, w_2\} \subseteq W$ it holds that $\mathsf{SD}(\tau_{w_1}, \tau_{w_2}) \leq \epsilon$.

- **Equivocability:** Λ is equivocable if there is an efficient sampling algorithm Sam that given (τ, w) where τ is the transcript (including the oracle queries) of the commitment phase of $\langle S, \widehat{R}^\sigma \rangle$ (for an arbitrary receiver $\widehat{R}$) and any $w \in W$, if

  $$\Pr[w \text{ is the committed message and } \tau \text{ is the transcript of } \langle S, \widehat{R}^\sigma \rangle] \neq 0$$

  then $\mathsf{Sam}(\tau, w)$ outputs r according to the distribution $(r_S \mid \tau, w)$. Namely r is sampled according to the distribution of the private randomness r_S of (S, σ) conditioned on w being the committed message and τ being the transcript of the commitment phase.

We simply call the ILS Λ u-secure if it is $(1-1/u)$-binding and $(u, 1/u)$-hiding.

Public-coin ILS. Since ILS's are IPCPs by definition, we define an ILS to be public-coin similar to the way we defined public-coin IPCPs. Note that the soundness of an ILS might decrease when considered as a public-coin system.

Round-complexity of ILS. Since the interactive part of the decommiemnt phase of any ILS consists only of one message from the sender S to the receiver R, by the round-complexity of any ILS we only refer to the number of rounds during its commitment phase.

Composition of ILS's. Different forms of composition of IPCPs can be applied to ILS's just as well, but here we are interested in their parallel composition and a specific variant of it. In any k-fold composition of the ILS's the sender receives k private messages $w_1, \ldots, w_k$ where each $w_i \in W$ is used as the private message in the i'th ILS. Therefore the message space of the composed ILS potentially increases from W to W^k. We use parallel composition to expand the message space. We use a variant of parallel composition which uses the *same* message in all composed ILS's to to amplify the soundness of an ILS.

Definition 2.16 (Parallel Compositions of ILS's)**.** Let $\Lambda_1 = (S_1, \sigma_1, R_1), \ldots, \Lambda_k = (S_k, \sigma_k, R_k)$ be k ILS's. By $\Lambda_{\mathsf{par}} = (S_{\mathsf{par}}, \sigma_{\mathsf{par}}, R_{\mathsf{par}})$ we mean the the parallel composition of $\Lambda_1, \ldots, \Lambda_k$ as IPCPs according to Definition 2.12. Namely, during the decommitment phase the sender S_{par} sends the message $((w_1, r_{P_1}), \ldots, (w_k, r_{P_k}))$ to the receiver R_{par} who emulates the interactive algorithm R_i over the message (w_i, r_{P_i}) for $i \in [k]$ (and accepts if all of them accept).

By a *same-message* (parallel) composition $\Lambda_{\mathsf{smp}} = (S_{\mathsf{smp}}, \sigma_{\mathsf{smp}}, R_{\mathsf{smp}})$ of $\Lambda_1, \ldots, \Lambda_k$ we mean their parallel composition where the receiver enforces the condition that the same $w_1 = w_2 = \cdots = w_k \in W$ is used as the commmitted message in all of $\Lambda_1, \ldots, \Lambda_k$. Namely, in the decommitment phase of Λ_{smp}, the sender S_{smp} sends $(w, r_{S_1}, \ldots, r_{S_k})$ to the receiver R_{smp} and the receiver runs the verification algorithm similar to R_{par} but over the message $((w, r_{P_1}), \ldots, (w, r_{P_k}))$.

Note that the message space of the same-message composition Λ_{smp} stays the same as W (as opposed to the regular parallel composition Λ_{par} which expands the message space to W^k).

Lemma 2.17 (Properties of parallel compositions of ILS's)**.** *Let $\Lambda_1 = (S_1, \sigma_1, R_1), \ldots, \Lambda_k = (S_k, \sigma_k, R_k)$ be k ILS's with the same round complexity and the same message space W, and let $\Lambda_{\mathsf{par}} = (S_{\mathsf{par}}, \sigma_{\mathsf{par}}, R_{\mathsf{par}})$ and $\Lambda_{\mathsf{smp}} = (S_{\mathsf{smp}}, \sigma_{\mathsf{smp}}, R_{\mathsf{smp}})$ be their parallel and same-message compositions. Then we have:*

1. **Binding:** *If Λ_i is $(1 - \delta_i)$-binding for all $i \in [k]$, then Λ_{par} is $(1 - \sum_i \delta_i)$-binding and Λ_{smp} is $(1 - \prod_i \delta_i)$-binding.*
2. **Hiding:** *If Λ_i is (u, ϵ_i)-hiding for all $i \in [k]$, then Λ_{par} and Λ_{smp} are both $(u, \sum_i \epsilon_i)$-hiding.*
3. **Equivocability:** *If Λ_i is equivocable for all $i \in [k]$, then both of Λ_{par} and Λ_{smp} are also equivocable.*

Proof. .

Binding. S_{par} of Λ_{par} gets bound to a message $(w_1, \ldots, w_k)$ if all of S_i's for $i \in [k]$ get bound to a message w_i. By the union bound and the $(1 - \delta_i)$-binding of Λ_i with probability at most $\sum_i \delta_i$ by the end of the commitment phase there exist an $i \in [k]$ where S_i is not bound to a fixed w_i. Thus S_{par} has binding at least $(1 - \sum_i \delta_i)$.

On the other hand, since S_{smp} is forced to use the same message w in all of R_i's, S_{smp} gets bound to a message w if for *at least* one $i \in [k]$ R_i gets bound to a message w. Recall that the latter happens whenever the inefficient modification of the receiver R'_i defined in binding part of Definition 2.15 rejects and by the binding of Λ_i this rejection happens with probability at least $1 - \delta_i$. Therefore by the soundness amplification of parallel composition of IPCPs (Part 1 of Lemma 2.13), with probability $1 - \prod_i \delta_i$ at least one of R'_i's reject in the execution of S_{smp} in which case the sender S_{smp} gets committed to at most one possible message w.

Hiding. We use a standard hybrid argument (which works the same for both Λ_{par} and Λ_{smp}). Fix any two messages $w^1 = (w_1^1, \ldots, w_k^1), w^2 = (w_1^2, \ldots, w_k^2)$ and consider $k+1$ experiments as follows. In the i'th experiment for $0 \leq i \leq k$, we use a parallel composition of $\Lambda_1, \ldots, \Lambda_k$ while for $0 \leq j \leq i$ the committed message in Λ_j is w_j^1 and for $i+1 \leq j \leq k$, the committed message in Λ_j is w_j^2. Fix any malicious receiver $\widehat{R_{\mathsf{par}}}$ who asks at most u queries from its locking oracle σ. Let τ_i be the random variable which consists of the transcript of the commitment phase in the i'th experiment. If $\mathsf{SD}(\tau_0, \tau_k) > \sum_i \epsilon_i$, then there should exist $i \in [k]$ such that $\mathsf{SD}(\tau_{i-1}, \tau_i) > \epsilon_i$. Therefore $\widehat{R_{\mathsf{par}}}$ can distinguish between the experiments $i-1$ and i with advantage ϵ_i. Now we claim that there exists a malicious receiver $\widehat{R}$ that can break the ϵ_i-hiding of Λ_i by ϵ_i-distinguishing the two experiments where in the first one w_i^1 and in the second one w_i^2 is used as the committed message in Λ_i. $\widehat{R}$ will run $\widehat{R_{\mathsf{par}}}$ while simulating the $k-1$ other senders and oracles of $\widehat{R_{\mathsf{par}}}$ as follows. For $\Lambda_0, \ldots, \Lambda_{i-1}$ $\widehat{R}$ simulates their senders and locking oracles while w_j^1 is used as the committed string in Λ_j and in $k-i$ other games $\Lambda_{i+1}, \ldots, \Lambda_k$ it simulates their senders and locking oracles while w_j^2 is used as the committed string in Λ_j. Then it is easy to see that $\widehat{R}$ will ϵ_i-distinguish between the two experiments because $\widehat{R_{\mathsf{con}}}$ ϵ_i-distinguishes τ_{i-1} from τ_i. But this contradicts the ϵ_i hiding of Λ_i.

Equivocability. Let $\tau_{\mathsf{par}} = (\tau_1, \ldots, \tau_k)$ be the transcript of the interaction of R_{par} with $(S_{\mathsf{par}}, \sigma_{\mathsf{par}})$ where τ_i consists of the part of the transcript for Λ_i and let $w = (w_1, \ldots, w_k)$ be the message that has a non-zero chance of being the private message conditioned on τ_{par} being the transcript. To sample a randomness $(r_{P_1}, r_{P_2}, \ldots, r_{P_k})$ consistent with τ_{par} and w one has to sample r_{P_i} consistent with (τ_i, w_i) for all $i \in [k]$ which is possible by the equivocability of Λ_i's. ■

Therefore we can use Λ_{par} to expand the message space at the cost of slight loss in hiding and binding, while Λ_{smp} can be used to amplify the soundness at the cost of slight loss in hiding.

3 Statistically Zero-Knowledge IPCP for NP

In this section we show how to construct a $2^{\Omega(n)}$-secure constant-round SZK-IPCP for any language $L \in \mathbf{NP}$ where both the prover and the PCP oracle in our construction can be implemented efficiently given a witness w for $x \in L$. We also show how to achieve a $2^{\Omega(n)}$-*adaptively*-secure SZK-IPCP for any $L \in \mathbf{NP}$ at the cost of poly(n) round-complexity. More formally we prove the following two theorems.

Theorem 3.1 (Constant-round SZK-IPCP for **NP**)**.** *For any language $L \in \mathbf{NP}$ there exists a 2-round efficient public-coin SZK-IPCP Γ_{2R} for L with security $2^{\Omega(n)}$. Moreover, the simulator of Γ_{2R} is straight-line and therefore by Lemma 2.13 for a small enough constant c, a 2^{cn}-fold concurrent composition of Γ_{2R} remains $(2^{\Omega(n)}, 2^{-\Omega(n)})$-SZK.*

Theorem 3.2 (Adaptively-secure SZK-IPCP for **NP**)**.** *There exists a (*poly(n)*-round) efficient SZK-IPCP Γ_{adap} for L with* adaptive-*security $2^{\Omega(n)}$.*

The oracle is inherent for SZK. If we only want to achieve computational zero-knowledge, we can remove the PCP oracle from the IPCP model and only the interaction would suffice to achieve this goal (under the computational assumption that one-way functions exist [GMW91]). On the other hand, if the domain of the PCP oracle π were only of size poly(n) (rather than

super-polynomial), then a malicious polynomial-time verifier could read all of the information in π and so the language L would be in the class $\mathbf{SZK} \subset \mathbf{AM} \cap \mathbf{co\text{-}AM}$ which does not contain the class **NP** unless the polynomial hierarchy collapses. This means that the existence of a oracle π with a super-polynomial domain is inherent to achieve unconditional zero-knowledge for **NP**.

Intuition behind Theorem 3.1. Our main step to prove Theorems 3.1 is to construct an interactive (ILS) (Definition 2.14), a primitive corresponding to commitment schemes in the IPCP model. In Theorem 4.1 we present an ILS with optimal round complexity (i.e. one round). Then we feed our ILS (as a commitment scheme) into the well-known construction of [GMW91] to achieve zero-knowledge for **NP** with non-negligible soundness. A classical way to amplify the soundness of proof systems (while keeping the round-complexity) in the standard model of interaction is to use parallel composition. We use the fact (Part 1) that parallel composition of IPCP's decreases the soundness error exponentially. The latter result (i.e. Part 1 of Lemma 2.17) is interesting on its own since the IPCP model lies in between the single-prover and the multi-prover models and it is known [FRS88] that the parallel repetition does *not* amplify the soundness in a simple exponential form (as one would wish). Secondly, we show that although the parallel composition might hurt the zero-knowledge in general, by crucially using equivocability feature of our ILS (see Definition 2.14) one can prove that SZK is preserved under parallel composition.

Lemma 3.3. *(Followed by Part 1 of Theorem 4.1) There exist an efficient ILS $\Lambda = (S, \sigma, R)$ for the message space $\{1, 2, 3\}$ with security $2^{\Omega(n)}$ which is public-coin and equivocable. Moreover the commitment phase of Λ has only one round of interaction.*

To use the construction of [GMW91] we also need the following technical lemma.

Lemma 3.4 (Hiding of selective opening of ILS's)**.** *Let $\Lambda_1 = (S_1, \sigma_1, R_1), \ldots, \Lambda_k = (S_k, \sigma_k, R_k)$ be ILS's with the same round complexity and the same message space W and suppose that Λ_i is (u, ϵ_i)-hiding for all $i \in [k]$. Also let D be an arbitrary (perhaps correlated) distribution over W^k. Now consider an arbitrary receiver $\widehat{R}$ who asks at most u oracle queries and interacts with the parallel composition of $(S_1, \sigma_1), \ldots, (S_k, \sigma_k)$ in the following experiments* Real *or* Simul*. Then $\widehat{R}$ can not distinguish between* Real *and* Simul *with an advantage more than $3\sum_i \epsilon_i$. In other words the statistical distance between the view of $\widehat{R}$ in the two experiments is at most $3\sum_i \epsilon_i$.*

Experiment Real**:**

1. *$(w_1, \ldots, w_k)$ is sampled according to the distribution D, $(r_1, \ldots, r_k)$ is sampled uniformly at random and (S_i, σ_i) receives (w_i, r_i) as its private message and randomness.*
2. *$\widehat{R}$ interacts with $(S_1, \sigma_1), \ldots, (S_k, \sigma_k)$ in parallel till the end of the commitment phases.*
3. *$\widehat{R}$ selects an arbitrary set $B \subseteq [k]$ and receives (w_i, r_i) for all $i \in B$.*
4. *$\widehat{R}$ can continue asking up to u oracle queries (including the previous queries asked) from the combined oracle $\sigma = (\sigma_1 | \ldots | \sigma_k)$.*

Experiment Simul**:**

1. *$(r'_1, \ldots, r'_k)$ is sampled uniformly at random and (S_i, σ'_i) receives $(0, r'_i)$ as its private message and randomness.*
2. *$\widehat{R}$ interacts with $(S_1, \sigma'_1), \ldots, (S_k, \sigma'_k)$ in parallel till the end of the commitment phases.*
3. *$\widehat{R}$ selects a subset $B \subseteq [k]$. At this point $(w_1, \ldots, w_k)$ is sampled according to D, and r_i is sampled at random conditioned on being consistent with w_i and the transcript of $\langle S_i, \widehat{R}^{\sigma'_i} \rangle$. Then $\widehat{R}$ receives (w_i, r_i) for all $i \in B$.*
4. *Now for all $i \in B$ let σ_i be the oracle generated with (w_i, r_i) as its private message and randomness, and for all $i \in [k] \setminus B$ let σ_i be the same as σ'_i. Let the combined oracle be defined as $\sigma = (\sigma_1 | \ldots | \sigma_k)$. $\widehat{R}$ can continue asking up to u oracle queries (including the previous queries) from the combined oracle σ.*

Proof. We will use several hybrid arguments. We first define one more experiment:

Experiment Mixed**:**

1. $(r'_1, \ldots, r'_k)$ is sampled uniformly at random and (S_i, σ'_i) receives $(0, r'_i)$ as its private message and randomness.
2. $\widehat{R}$ interacts with $(S_1, \sigma'_1), \ldots, (S_k, \sigma'_k)$ in parallel till the end of the commitment phases.
3. $\widehat{R}$ selects a subset $B \subseteq [k]$. At this point $(w_1, \ldots, w_k)$ is sampled according to D, and r_i is sampled at random conditioned on being consistent with w_i and the transcript of $\langle S_i, \widehat{R}^{\sigma'_i} \rangle$. Then $\widehat{R}$ receives (w_i, r_i) for all $i \in B$.
4. Now for *all* $i \in k$ let σ_i be the oracle generated with (w_i, r_i) as its private message and randomness, and let the combined oracle σ be defined as $\sigma = (\sigma_1 | \ldots | \sigma_k)$. $\widehat{R}$ can continue asking up to u oracle queries (including the previous queries) from the combined oracle σ'.

Claim 3.5. *For any receiver $\widehat{R}$ who asks at most u oracle queries, the statistical distance between the view of $\widehat{R}$ in* Real *and* Mixed *is at most $\sum_i \epsilon_i$.*

Proof. Let τ_{mixed} and τ_{real} be the transcript of the commitment phases in the experiments Mixed and Real. As a mental experiment suppose:

1. In Mixed, the sampling $(w_1, \ldots, w_m) \leftarrow D_k$ is done at the beginning (similar to Real).
2. In both Real and Mixed, the random seeds $(r_1, \ldots, r_k)$ are chosen after the commitment phases are done, conditioned on the transcript of the commitment phases (τ_{mixed} or τ_{real}) and $(w_1, \ldots, w_m)$.

Now the only difference between Real and Mixed is in the statistical distance between $(w_1, \ldots, w_k, \tau_{\mathsf{mixed}})$ and $(w_1, \ldots, w_k, \tau_{\mathsf{real}})$ in the two experiments. The crucial point is that the distribution of r_i's conditioned on $(w_1, \ldots, w_k, \tau)$ are independent and thus the way they are sampled in Real and Mixed (conditioned on $(w_1, \ldots, w_k, \tau)$) is the same. Now suppose the statistical distance between $(w_1, \ldots, w_k, \tau_{\mathsf{mixed}})$ and $(w_1, \ldots, w_k, \tau_{\mathsf{real}})$ is more than $\sum_i \epsilon_i$. Then by an average argument we can

fix the value of $(w_1, \ldots, w_k)$ and still keep the statistical distance to be more than $\sum_i \epsilon_i$. Namely for fixed values of $(w_1, \ldots, w_k)$, the statistical distance between τ_{mixed} and τ_{real} is more than $\sum_i \epsilon_i$. Notice that the difference between Mixed and Real (after fixing $(w_1, \ldots, w_k)$) is that τ_{mixed} is generated with $(0, \ldots, 0)$ as the private messages while τ_{real} is generated with $(w_1, \ldots, w_k)$. But this contradicts the hiding property of parallel composition of ILS's (i.e. Part 2 of Lemma 2.17). ■

The following claim together with Claim 3.5 (and the triangle inequality over the statistical distances) finishes the proof.

Claim 3.6. *For any receiver $\widehat{R}$ who asks at most u oracle queries, the statistical distance between the view of $\widehat{R}$ in* Mixed *and* Simul *is at most $\sum_i 2\epsilon_i$.*

Before proving Claim 3.6 consider the following two experiments.

Experiment Com0Dec0**:**

1. $\widehat{R}$ interacts with (S_i, σ_i) in the commitment phase conditioned on 0 being the private message of (S_i, σ_i).
2. $\widehat{R}$ can continue asking up to u oracle queries.

Experiment Com0Dec$w$**:**

1. $\widehat{R}$ interacts with (S_i, σ_i) in the commitment phase conditioned on 0 being the private message of (S_i, σ_i).
2. A randomness of the sender r_S is sampled conditioned on the view of $\widehat{R}$ and w being the private message.
3. $\widehat{R}$ can continute asking up to u oracle queries, but the oracle answers will be based on (w, r_S) being the private message and the randomness of the oracle.

Claim 3.7. *For any message w, any receiver $\widehat{R}$ who asks at most u oracle queries can distinguish between the* Com0Dec0 *and* Com0Dec$w$ *by an advantage of at most $2\epsilon_i$.*

Proof. Let the experiment Com$w$Dec$w$ be an experiment similar to Com0Dec$w$ where the transcript of the commitment phase was generated conditioned on w being the private message. By the (u, ϵ_i)-hiding of the ILS Λ_i the statistical distance between the view of $\widehat{R}$ in Com0Dec0 and Com$w$Dec$w$ is at most ϵ_i. We can apply Lemma 2.2 (with X, Y, and Z being in order the view of $\widehat{R}$ in Com0Dec0, Com$w$Dec$w$, and Com0Dec$w$, and the first part of each random variable being the view of the commitment phase) to conclude that the statistical distance of view of $\widehat{R}$ in Com0Dec0 and Com0Dec$w$ is at most $2\epsilon_i$. ■

Proof. (of Claim 3.6) Suppose $\widehat{R}$ can distinguish between Simul and Mixed with advantage more than $\sum_i \epsilon_i$. By an average argument we can get fix the sample $(w_1, \ldots, w_k) \leftarrow D_x$ and use it in Simul and Mixed and keep the distinguishing advantage of $\widehat{R}$ more than $\sum_i \epsilon_i$. Now we use a hybrid argument. For $0 \leq j \leq k$, define the j'th hybrid experiment as follows.

Experiment Hybrid_j

1. $(r'_1, \ldots, r'_k)$ is sampled uniformly at random and (S_i, σ'_i) receives $(0, r'_i)$ as its private message and randomness.

2. $\widehat{R}$ interacts with $(S_1, \sigma'_1), \ldots, (S_k, \sigma'_k)$ in parallel till the end of the commitment phases.

3. $\widehat{R}$ selects a subset $B \subseteq [k]$. At this point $(w_1, \ldots, w_k)$ is sampled according to D, and r_i is sampled at random conditioned on being consistent with w_i and the transcript of $\langle S_i, \widehat{R}^{\sigma'_i} \rangle$. Then $\widehat{R}$ receives (w_i, r_i) for all $i \in B$.

4. Now for all $i \in B \cup [j]$ let σ_i be the oracle generated with (w_i, r_i) as its private message and randomness, and for all $i \in [k] \setminus (B \cup [j])$ let σ_i be the same as σ'_i. Let the combined oracle be defined as $\sigma = (\sigma_1 | \ldots | \sigma_k)$. $\widehat{R}$ can continue asking up to u oracle queries (including the previous queries) from the combined oracle σ.

Note that $\mathsf{Hybrid}_0 = \mathsf{Simul}$ and $\mathsf{Hybrid}_k = \mathsf{Mixed}$ (with the mentioned difference that $(w_1, \ldots, w_k)$ is fixed now). Therefore if $\widehat{R}$ can distinguish between Simul and Real with advantage more than $\sum_i 2\epsilon_i$, then there exists $i \in [k]$ such that $\widehat{R}$ can distinguish between Hybrid_{i-1} and Hybrid_i with advantage more than $2\epsilon_i$. It is easy to see (similar to the proof of Part 2 of Lemma 2.17) that $\widehat{R}$ can be converted into another malicious receiver $\widehat{R}_i$ who asks at most u oracle queries and is able to distinguish between $\mathsf{Com0Dec0}$ and $\mathsf{Com0Dec}w_i$ with advantage more than $2\epsilon_i$. But the latter is not possible because of Claim 3.7. ■

Therefore Lemma 3.4 follows from Claims 3.6 and 3.5 and the triangle inequality over the statistical distance.[4] ■

Now we prove Theorem 3.1 using Lemma 3.3 and Lemma 3.4.

Proof. (of Theorem 3.1) Let $L \in \mathbf{NP}$, and let x be an input to an IPCP for L. By using the Cook-Levin reduction one can efficiently compute (from x) a graph G where $x \in L$ iff $G \in \mathsf{3COL}$, and moreover any witness for $x \in L$ can be efficiently converted into a witness for $G \in \mathsf{3COL}$ and vice versa. So we would assume that the parties always will run the Goldreich-Levin reduction first and then will work with a graph G as their input where G has $n = |N|$ vertices and $m = |M|$ edges. By a padding argument we can always assume that $n \geq |x|$ and so any $u(n)$-secure IPCP for $\mathsf{3COL}$ will yield a $u(n)$-secure IPCP for L as well. Therefore without loss of generality will work with $L = \mathsf{3COL}$ directly.

We use the zero-knowledge construction of [GMW91] which in turn uses a commitment scheme. For the needed commitment scheme we use the ILS $\Lambda = (S, \sigma, R)$ of Lemma 3.3.

Construction 3.8 (Weakly-sound SZK-IPCP for $\mathsf{3COL}$)**.** Let $w \colon [n] \to \{1, 2, 3\}$ be a proper 3-coloring for G where $w(i) \neq w(j)$ if $i \neq j$. Let $\Lambda = (S, \sigma, R)$ be the ILS of Lemma 3.3 and let w be the secret input given to (S, σ). The IPCP $\Gamma = (P, \pi, V)$ is as follows.

[4] Note that the reductions in various hybrid arguments in the proof of Lemma 3.4 are *not* efficient, but they keep the number of queries asked by $\widehat{R}$ bounded (which was sufficient). The reductions can be done efficiently if Λ_i's are equivocable.

1. The prover P chooses $s \leftarrow S_3$ at random where S_3 is the set of all permutations over the set $\{1,2,3\}$. P uses s to "randomize" the coloring w to get a new 3-coloring of G as $c(i) = s(w(i))$ for all $i \in N$. Note that c is also a proper 3-coloring of G. Moreover for any edge $(i,j) \in M$ it holds that $(c(i), c(j))$ is uniformly distributed over $\{1,2,3\}^2$ conditioned on $c(i) \neq c(j)$.

2. The prover P and the verifier V will engage in an n-fold parallel composition of the commitment phase of the ILS Λ where $c(i)$ and r_i are, in order, the committed message and the private randomness used by (S_i, σ_i) (in the i'th instance of Λ). The interaction of this step stops right before the decommitment phases start.

3. The verifier V chooses an edge $(i,j) \leftarrow M$ at random and sends (i,j) to the prover P.

4. The prover decommits the i'th and j'th instances of Λ by sending $(c(i), r_i)$ and $(c(j), r_j)$ to the verifier V.

5. The verifier V verifies $(c(i), r_i), (c(j), r_j)$ as decommitments of the i'th and j'th and rejects if either of them fail or if $c(i) = c(j)$.

Claim 3.9. *The IPCP $\Gamma = (P, \pi, V)$ of the Construction 3.8 has the following properties.*

1. **Completeness:** *If G is a 3-colorable graph, then $\Pr[\langle P, V^\pi \rangle(G) = 1] = 1$.*
2. **Soundness:** *Γ is $\frac{1}{2m}$-sound. Namely, if G is not 3-colorable, then $\Pr[\langle P, V^\pi \rangle(G) = 0] \geq \frac{1}{2m}$.*
3. **SZK:** *Γ is $(2^{\Omega(n)}, 2^{-\Omega(n)})$-SZK with a straight-line simulator.*
4. **Efficiency:** *Γ can be implemented efficiently (when (P, π) are given a proper coloring w of G as their private input).*
5. **Round-complexity:** *If the commitment phase of Λ has t rounds, then Γ has round complexity $t + 1$.*
6. **Public-coin:** *Γ is public-coin if Λ is public-coin.*

Proof. .

Completeness. The completeness follows from the completeness of Λ and the fact that w is a proper coloring.

Soundness. Suppose G is not 3-colorable. Since Γ runs n instances of Λ in parallel, by Part 1 of Lemma 2.17, with probability at least $1 - n\delta$, by the end of the commitment phases, for each $i \in [n]$ there is at most one possible $c(i)$ that the prover can decommit to (together with some randomness r_i) successfully. In the following we assume that such event has happened. Now for each $i \in [n]$ let $c(i)$ be the possible color that the prover can decommit to and let $c(i) = *$ if such color does not exist. Since G is not 3-colorable, with probability at least $1/m$ the verifier chooses an edge (i,j) such that either $c(i) = *$, or $c(j) = *$, or $c(i) \neq c(j)$. But in all these cases the verifier will reject. Therefore the verifier will reject at some point during he interaction with probability at least $(1 - \delta n) \cdot \frac{1}{m}$. Recall that Λ had hiding $1 - \delta \geq 1 - 2^{\Omega(n)}$, which means that Γ has soundness $(1 - n\delta) \cdot \frac{1}{m} \geq \frac{1 - n2^{-\Omega(n)}}{m} \geq \frac{1}{2m}$.

SZK. Suppose $\widehat{V}$ is a malicious verifier which asks up to u queries from π where Λ (of Lemma 3.3) is (u,ϵ)-hiding. The simulator Sim starts by plugging in a random seed for $\widehat{V}$ and interacting with $\widehat{V}$ similar to the way that (P,π) would do. But the problem is that Sim is not given the coloring w and cannot get the randomized proper coloring c. Instead Sim does the following.

1. Sim uses the trivial (and most probably improper) coloring $c(i)=0$ for all $i\in N$ and interacts with $\widehat{V}$.
2. After receiving the edge $(i,j)\in M$ from $\widehat{V}$, the simulator chooses $(c(i),c(j)) \leftarrow \{1,2,3\}^2$ conditioned on $c(i)\neq c(j)$ and chooses r'_i (resp. r'_j) at random conditioned on being consistent with $c(i)$ (resp. $c(j)$) and the transcript of $\langle S_i, \widehat{V}^{\sigma_i}\rangle$ (resp. $\langle S_j, \widehat{V}^{\sigma_j}\rangle$). The latter is possible because of the equivocability of Λ.
3. Sim answers the remaining oracle queries of $\widehat{V}$, but it pretends that $(c(i),r'_i)$ and $(c(j),r'_j)$ are the private message and randomness used in order by σ_i and σ_j. (For queries to other oracles π_t where $t\notin\{i,j\}$, Sim continues using $(0,r_t)$ as the message and the randomness used for the oracle π_t.)

When one is interacting with the honest prover P and the honest oracle π, for any fixed edge $(i,j)\in M$ the distribution of the colors of its vertices $(c(i),c(j))$ is uniformly distributed over $\{1,2,3\}^2$ conditioned on $c(i)\neq c(j)$. Therefore by Lemma 3.4, the simulator Sim generates a view for $\widehat{V}$ which is $3\sum_{i\in[n]}\epsilon = (3n\epsilon)$-close to the view of $\widehat{V}$ when interacting with the honest prover and the oracle (P,π). Since Λ is $(u,\epsilon)=(2^{\Omega(n)},2^{-\Omega(n)})$-hiding, it follows that Γ has SZK $(2^{\Omega(n)},3n2^{-\Omega(n)})=(2^{\Omega(n)},2^{-\Omega(n)})$ with a straight-line simulator.

The efficiency, round-complexity, and public-coin properties are immediate. ■

Finally let $\Gamma_{2\mathsf{R}}$ be an mn-fold parallel composition of Γ. By Part 1 of Lemma 2.13, $\Gamma_{2\mathsf{R}}$ has soundness $1-(1-1/2m)^{mn}=1-2^{\Omega(n)}$. Also by Part 3 of Lemma 2.13 $\Gamma_{2\mathsf{R}}$ has SZK $(2^{\Omega(n)},mn2^{-\Omega(n)})=(2^{\Omega(n)},2^{-\Omega(n)})$ with a straight-line simulator. Clearly $\Gamma_{2\mathsf{R}}$ remains complete, efficient, and 2-round and this finishes the proof of Theorem 3.1. ■

Proof. (of Theorem 3.2) To prove Theorem 3.2 it is enough to get an SZK-IPCP with non-negligible adaptive soundness. Then by Part 2 of Lemma 2.13 we can use the sequential composition to amplify the adaptive-soundness.

Note that if the verifier asks only one query from the oracle, then there is no difference between soundness and adaptive-soundness.

A construction of [BGKW88] takes any multi-prover interactive proof system and compiles it into a 2-prover system with non-negligible soundness where the verifier asks only one query from the oracle. The construction of [BGKW88] is as follows. The verifier sends its randomness r_V to the first prover and gets a full simulation of its interaction with the original multi-prover system. Then, in order to make sure that the first prover did not lie, one of the query/answer pairs claimed by the first prover is chosen at random and is verified by the second prover. If the answers were different the verifier rejects. It is easy to see that if the first prover does not simulate the game honestly as a simulation of the multi-prover case, then it will be caught with probability roughly $1/v$ where v is the total number of queries of the interaction being simulated. (We refer the reader to [BGKW88] for more details).

As a first try, we use the construction of [BGKW88] to compile our SZK-IPCP of Theorem 3.1 into one with non-negligible adaptive-soundness and only one oracle query. Namely, we starts ask the verifier to send its randomness to the prover who simulates the whole execution of the original system. Then we choose one of the query/answer pairs claimed by the prover and verify its correctness with the oracle (who is supposed to know the answer to such queries). Unfortunately, the problem with this construction is that the verifier is also able to "reset" the original prover through the new oracle. That is because the new oracle is supposed to know the answer to all the queries to the original prover and there is no restriction on which queries the verifier will ask from the oracle. Therefore we loose the SZK property.

To keep the SZK property, as a second try, we change the construction above as follows. The verifier will choose only one of the *oracle* queries, simulated by the new prover, and verifies that query with the new oracle. Now if the prover lies about any of the oracle-type simulated queries, it still will be caught by the verifier with non-negligible probability. Unfortunately, this time we might loose the soundness. The reason is that if the soundness of the original system relies on the oracle queries being hidden from the prover, since in the new simulation we are revealing the oracle queries of the original system to the prover, the prover might honestly simulate the oracle queries but use this knowledge to lie about the simulated queries of the original prover. But the good news is that if the original construction of SZK-IPCP was public-coin, then the soundness is still guaranteed even if the prover gets to see the oracle queries, and so in this case we do *not* loose the soundness! More formally we use the following construction.

Construction 3.10 (From soundness to weak adaptive-soundness)**.** Let $\Gamma = (P, \pi, V)$ be an IPCP. We construct new IPCP $\Gamma' = (P', \pi', V')$ as follows.

- **PCP oracle:** The (honest) oracle π' will be the same as π.
- **The interaction of P' and V':**
  1. V' chooses a randomness r_V for V and interacts with P' similar to the interaction of V and P. The main difference is that any time that V needs to ask a query from π, V' will ask this query from the *prover* P' instead.
  2. At this time V' has a full description of a possible interaction between V^{π} and P and rejects if V would reject in this interaction.
  3. Suppose $q_1, \ldots, q_v$ are the oracle queries that were asked from the prover P' and let a_i be the answer claimed by P' for the oracle query q_i. If V' didn't reject in the previous step, it will chooses a random $i \in [v]$ and queries q_i from the oracle π'. Let $a'_i = \pi'(q_i)$ be the answer. V' rejects if $a_i \neq a'_i$.

Claim 3.11 (Properties of Construction 3.10)**.** *Let $\Gamma = (P, \pi, V)$ be an (efficient)* public-coin *IPCP for the language L which is $(1-\delta)$-sound, (u, ϵ)-SZK and public-coin where the honest verifier V asks at most v oracle queries. When Γ is used in Construction 3.10 the result is an (efficient) IPCP $\Gamma' = (P', \pi', V')$ for the language L which is $(u - v, \epsilon)$-SZK, and is $(1-\delta)/v$-*adaptively-*sound.*

Proof. .

SZK. Any simulator Sim for Γ with parameters (u, ϵ) can be used to get a simulator Sim' for Γ' with parameters $(u - v, \epsilon)$. The reason is that the view of any malicious verifier $\widehat{V'}$ in Γ' who asks at most $u - v$ can be simulated by another malicious verifier $\widehat{V}$ in Γ who uses $\widehat{V'}$ as a black box (and in a straight-line manner). The verifier $\widehat{V}$ runs $\widehat{V'}$ and whenever $\widehat{V'}$ asks an *oracle* query from the prover P', $\widehat{V}$ also asks this query from the oracle π. Thus any (straight-line) simulator for $\widehat{V}$ can be used to simulate the view of $\widehat{V'}$ with the same statistical distance ϵ.

Adaptive soundness. Since the verifier asks only one query from π', it does not matter whether π' is stateful or not. More formally, for any potentially stateful π' we define a (stateless) oracle π'' which given the query x answers according to π' when π' is asked *only* the query x. Note that from the point of view of V' there is no difference between π' and π'', so we might as well assume that V' asks its query from the *stateless* oracle π''.

Let Rej be the event that V' rejects. By the soundness of Γ as a *public-coin* IPCP if P' answers all of $q_1, \ldots, q_v$ honestly according to π'', then it would hold that $\Pr[\mathsf{Rej}] \geq 1-\delta$. It is crucial that Γ is public-coin, because in the interaction of V' and P', P' gets to know the oracle queries of V and the soundness should hold even in this case. But a malicious prover $\widehat{P'}$ can change $\Pr[\mathsf{Rej}]$ by lying about the answer to some of q_i's. Let Lie be the event that there exists $i \in [v]$ such that $a_i \neq \pi''(q_i)$, and let NoLie be the complement event that $a_i = \pi''(q_i)$ for all $i \in [v]$. By the $(1-\delta)$-soundness of the public-coin IPCP Γ it still holds that:

$$\Pr[\mathsf{Rej} \wedge \mathsf{NoLie}] \geq (1 - \delta) - \Pr[\mathsf{Lie}]. \tag{3.1}$$

On the other hand if Lie happens then P' will be caught with probability at least $1/v$, and so

$$\Pr[\mathsf{Rej} \mid \mathsf{Lie}] \geq 1/v. \tag{3.2}$$

Therefore we conclude that

$$\begin{aligned}
\Pr[\mathsf{Rej}] &= \Pr[\mathsf{Rej} \wedge \mathsf{NoLie}] + \Pr[\mathsf{Rej} \wedge \mathsf{Lie}] \\
&\geq \max(0, 1 - \delta - \Pr[\mathsf{Lie}]) + \Pr[\mathsf{Lie}] \cdot \Pr[\mathsf{Rej} \mid \mathsf{Lie}] && \text{by Inequality 3.1} \\
&\geq \max(0, 1 - \delta - \Pr[\mathsf{Lie}]) + \Pr[\mathsf{Lie}] \cdot \frac{1}{v} && \text{by Inequality 3.2} \\
&\geq (1 - \delta)/v. && \text{Achieved by } \Pr[\mathsf{Lie}] = 1 - \delta
\end{aligned}$$

■

We use Construction 3.10 over the IPCP $\Gamma = (P, \pi, V)$ of Theorem 3.1 to get a new IPCP $\Gamma' = (P', \pi', V')$. Let $v = \mathrm{poly}(n)$ be the number of oracle queries of V in Γ. Since Γ is public-coin and is $2^{\Omega(n)}$-secure, therefore Γ' will have adaptive-soundness $1/(2mv) = 1/\mathrm{poly}(n)$ and SZK $(2^{\Omega(n)} - v, 2^{-\Omega(n)}) = (2^{\Omega(n)}, 2^{-\Omega(n)})$.

Finally let Γ_{adap} be a mvn-fold sequential composition of Γ'. By Part 2 of Lemma 2.13, Γ_{adap} has adaptive-soundness $1 - (1 - 1/(2mv))^{mvn} = 1 - 2^{\Omega(n)}$. Also by Part 3 of Lemma 2.13 Γ_{adap} has SZK $(2^{\Omega(n)}, mvn2^{-\Omega(n)}) = (2^{\Omega(n)}, 2^{-\Omega(n)})$ and this finishes the proof of Theorem 3.2. ■

4 Interactive Locking Schemes

In this section we study ILS's and will construct an ILS with optimal round complexity.

Theorem 4.1. *(A round-optimal ILS) Let $\ell(n) = \text{poly}(n)$, then*

1. *There exist an efficient ILS $\Lambda = (S, \sigma, R)$ for the message space $\{0,1\}^\ell$ with security $2^{\Omega(n)}$ which has a commitment phase of only one round and is public-coin.*
2. *Any ILS with a noninteractive commitment phase needs an inefficient oracle σ and thus Λ has optimal round-complexity (as an efficient ILS).*

First, in Section 4.1 we present a general construction for ILS's from any interactive hashing scheme (IHS) (see Definition 4.2) where the round-complexity of the used IHS equals the round-complexity of the commitment phase of the constructed ILS. Using known constructions for IHS we get an ILS with a 2-round commitment phase. We show that this is the best one can get through this approach by showing that any IHS with the parameters needed for our construction requires at least 2 rounds (see Proposition 4.5).

Then in Section 4.2 we present a direct construction for ILS with only one round of interaction in the commitment phase. Both constructions of Sections 4.1 and 4.2 are $2^{\Omega(n)}$-secure and public-coin. Thus the 1-round construction of Section 4.2 proves Part 1 of Theorem 4.1.

Finally in Section 4.3 we will show that at least one round of interaction is needed in the commitment phase of any *efficient* ILS (proving Part 2 of Theorem 4.1).

4.1 ILS from IHS

Interactive hashing schemes (IHS — Definition 4.2) and their variants have been of great use in designing cryptographic protocols. Although the core central properties needed for IHS's in all these applications is of an information theoretic flavor, in the applications of IHS's in the computational regime [OVY93a, OVY93b, HHK$^+$05, HR07], compared to the applications in the information theoretic regime [CCM98, CCM98, DHRS04]) some extra properties are required for the IHS used. Here we use a minimal definition whose properties hold in all existing forms of IHS in the literature and show that the existence of any IHS according to this definition implies the existence of ILS.

Definition 4.2 (Interactive hashing)**.** An *interactive hashing scheme* (IHS) $\Psi = (S, R)$ is a two party protocol between a sender S and a receiver R where both are given 1^n as the security parameter and S is also given $w \in M$ as the private message. By the end of the protocol the transcript of the protocol determines two outputs $\{w_0 \neq w_1\} \subseteq \{0,1\}^n$ for which the following properties hold.

- **Completeness:** If S and R are honest, then $w \in \{w_0, w_1\}$.
- **Binding:** The protocol is ρ-binding, if for every *fixed* set $T \subset W$ of size $|T| \leq \rho \cdot |W|$ and any malicious sender $\widehat{S}$ the probability that both of w_0 and w_1 lie in T is at most $1/2$; namely $\Pr_{\langle \widehat{S}, R\rangle = (w_0, w_1)}[\{w_0, w_1\} \subset T] \leq 1/2$.
- **Hiding:** If the sender S is honest, then for any transcript τ of the protocol which determines the outputs (w_0, w_1) it holds that $\Pr_{w \leftarrow \{w_0,w_1\}}[w = w_0 \mid \tau] = \Pr_{w \leftarrow \{w_0,w_1\}}[w = w_1 \mid \tau] = 1/2$. Therefore if, $w \leftarrow W$ is chosen uniformly at random, w_0 and w_1 will have the same chance of being equal to w.

We call Ψ *efficient* if both S and R are efficient and *public-coin* if the messages of the receiver R only consist of her public coin tosses.

It is easy to see that if R simply sends a 2-to-1 pairwise-independent hash function $h\colon \{0,1\}^n \to \{0,1\}^{n-1}$ to S and S sends back $h(w)$, it gives a ρ-binding IHS for $\rho \approx 2^{-n/2}$. But, as we will see in Lemma 4.3 for our construction of ILS from IHS (Lemma 4.3) we need IHS's with non-negligible binding $\rho = \Omega(1/\text{poly}(n))$.

Lemma 4.3 (ILS from IHS)**.** *Let $\Psi = (S_\Psi, R_\Psi)$ be a k-round IHS with binding ρ over the message space $\{0,1\}^n$ with a deterministic sender S_Ψ. Then there exists a bit-ILS $\Lambda = (S, \sigma, R)$ with a k-round commitment phase which is $1/2$-binding and $(2^{n/2}, 2^{-n/2})$-hiding. Moreover if Ψ is efficient, then (S, σ) are efficient and the receiver R runs in time $\text{poly}(n, 1/\rho)$ (and so if in addition $\rho = 1/\text{poly}(n)$, then Λ is efficient as well).*

[OVY93a] presented an IHS with binding $\rho = \Omega(1)$ at the cost of $\text{poly}(n)$ rounds of interaction. [DHRS04] showed how to achieve a constant-round IHS with $1/\text{poly}(n)$ binding. In all the known constructions of IHS the sender is deterministic, and in our construction of ILS from IHS (Lemma 4.3) we use this property. The determinism of the sender is not crucial to us and if the sender is randomized, then we only need the following sampling property to hold: Given any transcript τ of the protocol determining (w_0, w_1) and any $i \in \{0,1\}$ one can efficiently sample r_i from the sender's space of randomness according to $r_i \leftarrow (r_S \mid \tau, w = w_i)$.

By using any known k-round efficient IHS with $1/\text{poly}(n)$-hiding and Lemma 4.3, we get an efficient ILS Λ which is $1/2$-binding and $(2^{\Omega(n)}, 2^{-\Omega(n)})$-hiding. Now Part 1 of Lemma 2.17 yields that an n-fold parallel composition of Λ is $2^{\Omega(n)}$-secure and finally an ℓ-fold same-message parallel composition expands the message space to $\{0,1\}^\ell$ while we keep the $2^{\Omega(n)}$-security. Therefore by using the 2-round IHS scheme of [DHRS04] and Lemma 4.3 we derive Part 1 of Theorem 4.1 (but still without the optimal round-complexity).

Intuition. Our approach in the proof of Lemma 4.3 is to use the well-known reduction from statistically hiding bit commitments to IHS in the *computational* setting [NOVY98] and its black-box proof of security. Roughly speaking the idea of [NOVY98] is to run the IHS over the message $y \in \{0,1\}^n$ where $y = f(x)$ and f is a permutation. Now if f is a one-way permutation the fraction of points y which the malicious sender $\widehat{S}$ is able to invert is a negligible $\epsilon = \text{neg}(n)$ fraction of $\{0,1\}^n$. Therefore if the ILS is ρ-binding for non-negligible ρ (and in particular $\epsilon < \rho$) with a noticeable probability, by the end of the IHS with outputs $\{y_0, y_1\}$, there is at most one y where $\widehat{S}$ can invert (i.e., present x such that $y = f(x)$) and claim that y was his private message. So if at the last step of the commitment phase $\widehat{S}$ sends a bit $d = b \oplus c$ where b is his bit-message and y is such that $y = y_c$, then she gets committed to only one possible value for b for which she can find a consistent c.[5]

But here we are interested in unconditionally secure protocols and we can not rely on the existence of one-way permutations or any other computational assumption. The idea is to use the locking oracle σ to hold a random (non-zero) point function $\sigma(x) = y$ where the sender knows the image y and the preimage x and thus can "invert" y to x. By using the IHS over the image y, the sender gets committed to a unique y (with probability at least $1/2$) assuming that there are

[5]The argument needs to be formalized through an *efficient* reduction and we refer the reader to the rather complicated full proof of [NOVY98].

fewer than $\rho \cdot 2^n$ many "invertible" points in the set $\sigma(\{0,1\}^n)$. But the receiver can guarantee that the latter is the case by picking $100/\rho$ random samples $x' \leftarrow \{0,1\}^n$ (conditioned on $x \neq x'$) and verifying that $\sigma(x') = 0$.

Proof. (of Lemma 4.3) We first describe the bit-ILS $\Lambda = (S, \sigma, R)$.

Construction 4.4 (Bit-ILS from IHS)**.** Let $b \in \{0,1\}$ be the private message hold by (S, σ). We will use the IHS Ψ with the properties described in Lemma 4.3 over the message space $\{0,1\}^n$ with binding ρ.

The commitment phase of Λ:

1. The common randomness r_R of (R, σ) will be $(x, y) \leftarrow \{0,1\}^n \times \{0,1\}^n$ where $y \neq 0$. The locking oracle σ will contain a point function: $\sigma(x) = y$ and $\sigma(x') = 0$ for $x' \neq x$.
2. S and R engage in the IHS Ψ where the sender S uses y as the private message of S_Ψ.
3. Let b be the private bit given to S, and let $(y_0 < y_1)$ be the output of the protocol Ψ and let c be such that $y = y_c$. As the last message of the commitment phase the sender S sends the bit $d = b \oplus c$ to the receiver R. Note that this does not require an extra round of interaction since d can be concatenated to the last message of the sender.

The decommitment phase of Λ:

1. S sends b and (x, y) to R.
2. R verifies that $\sigma(x) = y$ and $y = y_{d \oplus b}$ (and rejects otherwise).
3. For $i \in [n/\rho]$ the receiver R samples $x'_i \leftarrow \{0,1\}^n$ conditioned on $x'_i \neq x$ at random and verifies that $\sigma(x'_i) = 0$ holds for all $i \in [n/\rho]$ (and rejects otherwise).

Now we prove the properties of the ILS Λ of Construction 4.4.

Completeness is immediate.

Binding. Let $T = \{\sigma(x) \mid x \in \{0,1\}^n\} \setminus \{0\}$ and $T^{-1} = \{x \mid \sigma(x) \neq 0\}$ be the set of non-zero points in the oracle σ. There are two cases: either T^{-1} has size $|T^{-1}| \geq \rho \cdot 2^n$ or $|T^{-1}| < \rho \cdot 2^n$. If the first case $|T^{-1}| \geq \rho \cdot 2^n$ holds, then with probability at least $1 - (1-\rho)^{n/\rho} > 1 - 2^n$ one of the receiver's samples $\{x'_i\}$ will be sampled from T^{-1} and then the receiver R rejects. (The way we described the protocol the random samples x'_i's are chosen in the decommitment phase, but they are part of the randomness r_R of the receiver and are chosen in the beginning of the commitment phase.)

On the other hand, if the second case $|T^{-1}| < \rho \cdot 2^n$ holds, it implies that the size of $|T| \leq |T^{-1}| < \rho \cdot 2^n$ is small as well. Now by the binding property of the IHS Ψ, with probability at least $1/2$ one of $\{y_0, y_1\}$ lies out of T in which case there is only one way for the sender S to decommit to a bit b consistent with y_0, y_1 and d. Therefore the $1/2$-binding holds in either of the cases.

Hiding. If a malicious receiver $\widehat{R}$ asks up to $2^{n/2}$ queries from the oracle σ, they will be all answered zero with probability at least $1 - 2^{-n/2}$ by the uniformity of $x \leftarrow \{0,1\}^n$. But in that case the oracle queries of $\widehat{R}$ do not reveal any information about y and the hiding of the ILS Γ follows from the hiding of the IHS Ψ.

Equivocability. Let τ be the transcript of any malicious receiver $\widehat{R}$ by the end of the commitment phase. Given (b, τ) the sampler algorithm Sam does the following. Let $\sigma(x'_1) = y'_1, \ldots, \sigma(x'_t) = y'_t$ be the oracle queries asked by $\widehat{R}$ from σ appeared in the transcript τ and let (y_0, y_1) be the output of the executed IHS. If there exist $i \in [t]$ such that $y'_t \neq 0$, then it means that $\widehat{R}$ has found the committed bit $b = d \oplus c$ where c is such that $y'_i = y_c$. In this case the sampler Sam simply outputs (x'_i, y'_i). On the other hand if $y'_i = 0$ for all $i \in [t]$, then the sampler chooses $y = y_c$ where $c = b \oplus d$ and chooses $x \leftarrow \{0,1\}^n \setminus \{x'_1, \ldots, x'_t\}$ and outputs (x, y) (pretending that $\sigma(x) = y$). ■

By using the IHS of [DHRS04] (which has the fewest number of rounds among known IHS's) we get a 2-round ILS. The following proposition shows that the Construction 4.4 of ILS's from IHS's is not able to achieve efficient ILS's with one round (in the commitment phase).

Proposition 4.5 (Nontrivial IHS's need two rounds)**.** *Any IHS Ψ with at most 3 messages exchanged between the sender and the receiver has binding at most $\rho = 100/\sqrt{|W|}$. Therefore for $|W| = 2^n$, any IHS Ψ with non-negligible binding ρ needs at least two rounds of interaction.*

Proof. For starters we assume that Ψ has only one round (i.e. two messages) where the receiver sends its message first. Later we will extend the result to 3-message protocols. Suppose the first message sent from the receiver is q and is followed by an answer a from the sender. In the following discussion we fix a query q. Now any answer a corresponds to two elements $\{w_1^a, w_2^a\} \subset W$ which determines an edge in a graph G_q with the vertex set W. Each edge a has a probability of being used as the sender's message. By duplicating the edges we can change G_q into a multi-graph such that the distribution of the message of the sender corresponds to selecting a random edge from G_q. The hiding property implies that any two connected vertices x and y have the same degree $d(x) = d(y)$. Therefore each connected component of G_q is a regular graph. Now we show that such graphs have a relatively large matching.

Lemma 4.6 (Large matchings in regular graphs)**.** *Let G be any graph with v vertices such that the vertices have degree at least one and each connected component of G is regular. Then G has a matching of size at least $v/4$.*

Proof. By the Vising theorem each connected component of G can be properly colored with $\Delta + 1$ colors where Δ is the degree of the vertices of that component. Since each color forms a matching, therefore each component of G with u vertices has a matching of size at least $\frac{u\Delta}{2(\Delta+1)} \geq u/4$. Therefore G has a matching of size at least $v/2$. ■

Let M be the matching of size $|W|/4$ in G guaranteed by Lemma 4.6. The sender samples a random set $T \subset W$ by choosing each $w \in W$ with probability $10/\sqrt{|W|}$. Each edge of the matching M gets covered by T with probability at least $100/|W|$. Therefore with probability at least $1 - (1 - 100/|W|)^{|W|/4} > 1 - 2^{-25} > 99/10$ at least on of the edges in M gets covered.

Now we let q, the message of the receiver, also be chosen according to its distribution. For each such q, when we construct the graph G_q, the random set T covers one of the edges of G_q with probability at least $99/100$. Therefore, with probability at least $9/10$ over the choice of T, the set

T has the property that: with probability at least $9/10$ over the choice of q, at least one of the edges of G_q is covered by T. We call such T a good set.

On the other hand by Markov inequality with probability at least $9/10$, the size of T is bounded by $100\sqrt{|W|}$. Therefore by a union bound with probability at least $8/10$, T is both a good set and also of size at most $100\sqrt{|W|}$. This set T can be used by the sender to break the binding of Ψ, because with probability at least $9/10$ over the first message q of the receiver, the sender is able to send a message a (corresponding to the edge covered by T) which determines two elements in the set T as the output of the protocol.

Now we study 3-message protocols. If we add one more message q_1 from the receiver to to the sender (in addition to the the 2-messages in the protocols studied above), the last message of the receiver does not restrict the sender's cheating power, because the set of messages that the sender can claim to be his message does not depend on the last message of the sender.

Now suppose the protocol has three messages where the sender starts the interaction. In this case, the first message of the sender a_0 might decrease the space of possible values for w which are consistent with a_0. Now the sender *fixes* a value a_0 for his first message. Let W' be the set of values for w which are consistent with a_0. By using the same attack above (for the protocols which start with a_0), the sender can still find a set T of size at most $100\sqrt{|W'|} < 100\sqrt{|W|}$ to break the binding of Ψ.

■

4.2 A 1-round ILS

Theorem 4.7 (A 1-round ILS)**.** *There exists an efficient bit-ILS Λ_{1R} which is $2^{n/10}$-secure and has only one round of interaction in the commitment phase.*

Before proving Theorem 4.7 we need to develop some basis tools.

Lemma 4.8. *For any $m \in \mathbb{N}$, there is a set $\mathcal{M}$ of $m \times m$ Boolean matrices such that:* **(1)** *$0_{m\times m} \in M$, and all other $M \in \mathcal{M}$ are full rank,* **(2)** *$|\mathcal{M}| = 2^m$, and* **(3)** *for any nonzero (fixed) $x \in \{0,1\}^m$, xM is uniformly distributed over $\{0,1\}^m$ if we choose $M \in \mathcal{M}$ at random.*

Proof. Let $a \in GF(2^n)$. If $a \neq 0$, then the function $f_a(x) = ax$ is a permutation over $GF(2^m)$. The multiplication is a linear operation in the field, therefore we can think of M_a as the matrix representation of the mapping f_a which will have full rank if $a \neq 0$. Moreover if $x \neq 0$, then ax is uniformly distributed for a random a. So we can take $\mathcal{M} = \{M_a \mid a \in GF(2^m)\}$. ■

Thus we also get the following lemma, whose proof is immediate.

Lemma 4.9. *For $n > m$ let $\mathcal{A}$ be the family of $n \times m$ Boolean matrices as follows. To get a uniform member of A, choose the first $n-m$ rows all at random, and take the last m rows to be an independently chosen random member of $\mathcal{M}$ from Lemma 4.8. Then for any $0 \neq x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, it holds that $\Pr_{A\leftarrow\mathcal{A}}[xA = y] = 2^{-m}$.*

Proof. (of Theorem 4.7) We use the following construction for the ILS Λ_{1R}.

Construction 4.10 (A 1-round ILS)**.** Suppose $b \in \{0,1\}$ is the private message given to sender and the oracle (S, σ), and suppose R is the receiver. Let $m = 3n/4$. Below we associate $\{0,1\}^n$ with the integers $[0, 2^n)$ and all additions and subtractions below are modulo 2^n.

The commitment phase of Λ_{1R}:

1. Sender S chooses $a \leftarrow \{0,1\}^n$ at random. Let f_b be the function: $f_b(x) = 1$ iff $a \leq x < a+2^m$, and let f_{1-b} be the zero function over $\{0,1\}^n$. The locking oracle will be the combination of the two functions $\sigma = (f_0|f_1)$ (indexed by the first bit of the query to σ).

2. Receiver R samples $A \leftarrow \mathcal{A}$ from the family of matrices of Lemma 4.9 conditioned on the last m rows of A being independent[6] and sends A to S.

3. Sender S checks that the last m rows of A are independent, and if so he sends $h = aA$ to the receiver R.

The decommitment phase of Λ_{1R}:

1. Sender S sends (b, a) to the receiver R.

2. Receiver R does the following checks and rejects if any of them does not hold.

   (a) Check that $aA = h$.

   (b) Check that $f_{1-b}(a) = 0$, and $f_b(a) = 1$.

   (c) For each $i \in [0, m]$, sample $10n$ random points from $[a, a + 2^i)$ and check that $f_b(x) = 1$ for all of them, and also sample $10n$ random points from $(a - 2^i, a - 1]$ and check that $f_b(x) = 0$ for all of them

Now we study the properties of the ILS Λ_{1R}.

Completeness is immediate.

Binding. As a mental experiment we pretend that the randomness used during the decommitment phase by R is chosen in the decommitment phase (rather than in the beginning of the commitment phase).

For a fixed locking oracle σ, Let X_0 (resp. X_1) be the set of possible values of a that sender S can send to the receiver R as the decommitment of $b = 0$ (resp. $b = 1$) and get accepted in the decommitment phase with probability at least 2^{-2n}. We prove that by the end of the commitment phase, with probability at least $1 - 2^{-n/8}$, it holds that $|X_0| = 0$ or $|X_1| = 0$ which means that the sender has only one way to decommit the value b and get accepted with probability more than 2^{-2n}. But now if we choose the receiver's randomness in the commitment phase, since there are at most 2^{n+1} possible values for (b, a), it follows by a simple average argument that with probability at least $1 - 2^{2n-n-1}$ over the commitment phase, the prover gets committed to only one possible value for (b, a) which he can use to pass the decommitment phase successfully.

Claim 4.11. $X_0 \cap X_1 = \varnothing$.

Proof. If $a \in X_0 \cap X_1$. Then when a is used as the decommitment of 0, in Step 2b of the decommitment phase the receiver R checks that $f_0(a) = 1, f_1(a) = 0$. On the other hand in the case of decommitting to 1, receiver R checks that $f_b(a) = 0, f_{1-b}(a) = 1$, but they can't both hold at the same time. ■

[6]Note that the last rows of A are independent with probability $1 - 2^{-m} = 1 - 2^{-n}$.

Claim 4.12. *It holds that* $|X_0| \leq 2^{n-m}$ *and* $|X_1| \leq 2^{n-m}$.

Proof. We show that if $\{a, a'\} \subset X_0$ then $|a - a'| \geq 2^m$ (and this would show that $X_0 \leq 2^n/2^m$). Assume on the contrary that $a' < a$ and $a - a' < 2^m$. Let $i \in [1, m]$ be such that $2^{i-1} \leq a - a' < 2^i$. Then by the pigeonhole principle ether at least half of $\sigma([a', a])$ are zero or at least half of the values $\sigma([a', a])$ are one. Without loss of generality let assume that at least half of $\sigma([a', a])$ is zero. In this case at least $1/4$ of the values $\sigma([[a', a' + 2^i)])$ are zero. But then by Step 2c of the decommitment phase $(0, a')$ will be accepted with probability at most $(3/4)^{10n} < 2^{-2n}$, and therefore $a' \notin X_0$ which is a contradiction. ■

Claim 4.13. *With probability at least* $1 - 2^{\Omega(n)}$ *over the choice of* A*, it holds that* $|X_0| = 0$ *or* $|X_1| = 0$.

Proof. Fix any pair $a_0 \in X_0$ and $a_1 \in X_1$, we know that $a_0 \neq a_1$. Therefore, $\Pr_A[a_0 A = a_1 A] = \Pr_A[(a_0 - a_1)A = 0] = 2^{-m}$. Claim 4.12 yields that there are at most $2^{n-m}2^{n-m}$ such pairs, so by using a union bound, with probability at least $1 - 2^{-m}2^{2n-2m} = 1 - 2^{2n-3m}$ over the choice of A, it holds that $X_0 A \cap X_1 A = \varnothing$ which implies that if the sender sends any hash value h, the consistency check of Step 2a of the decommitment phase either makes $|X_0| = 0$ or $|X_1| = 0$. ■

As we said before Claim 4.13 implies that with probability $1 - \text{poly}(n) \cdot 2^{2n-3m} = 1 - \text{poly}(n) \cdot 2^{-n/4} \geq 1 - 2^{-n/8}$ over the interaction in the commitment phase the sender gets bound to a fixed $b \in \{0, 1\}$ to which he can decommit successfully.

Hiding. Suppose receiver R can ask at most $u \leq 2^{n/8}$ queries from the locking oracle σ. We claim that before sending the matrix A, all of receiver R's queries to σ are answered zero with probability at least $1 - 2^{-n/4}$. To see why, think of Z_{2^n} as being divided into $2^{n-m} = 2^{n/4}$ equal intervals such that a is the beginning of one of them. Since receiver R asks up to $2^{n/8}$ queries, before sending the matrix Z, he will ask a query from the interval beginning with a with probability at most $2^{n/8}/2^{n/4} = 2^{-n/8}$. Therefore (up to $2^{-n/8}$ statistical distance in the experiment) we can assume that the matrix A is chosen by receiver R independently of a.

After receiving h, the information that the receiver R knows about a is that it satisfies the equation $aA = h$. If we choose and fix the first $n - m$ bits of (a potential) a, then the remaining bits are determined uniquely because the last m rows of A are full rank. It means that for every $y \in [0, 2^{n-m})$ there is a unique solution for a in the interval $[y2^m, y2^m + 2^m)$, and they are all equally probable to be the true answer from the receiver's point of view.

Now again we claim that (although there are 2^m nonzero points in f_b) all the queries that the receiver R asks from f_b are answered 0 with probability at least $1 - 2^{-n/8}$. Let $Z = \{z \mid zA = h\}$ be the set of possible values for a. For $z \in Z$, let $I(z) = [z, z + 2^m)$. We claim that no $x \in \{0, 1\}^n$ can be in $I(z)$ for three different z's from Z. To see why, let $z_1 < z_2 < z_3$ and that $x \in I(z_1) \cap I(z_2) \cap I(z_3)$. But now the interval $[y2^m, y2^m + 2^m)$, containing z_2 separates z_1 and z_3, and so $z_3 - z_1 > 2^m$. Therefore $I(z_1) \cap I(z_3) = \varnothing$ which is a contradiction. So, if the receiver R asks u queries from f_b, he can ask queries from $I(z)$'s for at most $2u$ different z's (out of 2^{n-m} many of them). As a mental experiment assume that a is chosen from Z after the receiver R asked his queries, it holds that $I(a)$ will be an interval that the receiver R never asked any query from with probability at least $1 - u/2^{n-m} \geq 1 - \cdot 2^{-n/8}$. Therefore with probability at least $1 - 2^{-n/9}$ all of receiver R's queries during the commitment phase will be answered zero. But putting the oracle queries aside,

the hash value h does not carry any information about the bit-message b and therefore the scheme is $(1 - 2^{n/8})$-hiding.

Equivocability. Let τ be the view of an arbitrary receiver $\widehat{R}$ after the commitment phase. Let (A, h) be the matrix sent by $\widehat{R}$ and the hash value received from S, and let $(x_1, y_1), \ldots, (x_t, y_t)$ be the oracle query/answer pairs asked from f_b and described in τ (where b is the input given to the sampler algorithm Sam). For each x_i let $Z_i = \{z \mid y_i \in I(z) \wedge zA = h\}$ be the set of possible values for a which y_i falls into $I(z)$, and recall that $Z = \{z \mid zA = h\}$. (As we said before, since the last m rows of A are independent, $|Z_i| \leq 2$ for all i.) Note that $y_i = 0$ if and only if $a \notin Z_i$. Therefore the set of possible values for a is $W = Z \cap_{y_i \neq 0} Z_i \setminus \cup_{y_i = 0} Z_i$. Thus the sampler Sam simply outputs a random element of W as the value for a. This can be done efficiently since any candidate for a (and in particular the members of W) can be identified by their first $n - m$ bits, and the simulator is given the list $\cup_{y_i=0} Z_i$ of impossible values for a which is of length $O(t)$, and therefore the verifier can finish the sampling in time $\text{poly}(|\tau|) \geq \text{poly}(t)$ which is allowed. ■

4.3 Noninteractive Locking Schemes Cannot Be Efficient

By a *noninteractive locking scheme* (NLS), we mean an ILS where the commitment phase is noninteractive and sender S only participates in the decommitment phase. Note that an efficient locking scheme by definition uses $\text{poly}(n)$-sized circuits to implement the locking oracle σ, and therefore σ can have at most $\text{poly}(n)$ entropy. In this section we show that there exist no efficient NLS with super-polynomial security.

Since we are going to prove that NLS's cannot be efficient, we need to deal with unbounded senders. Thus we can no longer assume that the decommitment phase is only a message (b, r_S) sent to the receiver, because the randomness r_S used by the sender can be exponentially long. Therefore to prove the strongest possible *negative* result, we allow the decommitment phase of a NLS to be interactive.

The following theorem clearly implies Part 2 of Theorem 4.1.

Theorem 4.14. *Let $\Lambda = (S, \sigma, R)$ be any NLS for message space $\{0,1\}$ in which the function σ of the locking oracle has Shannon entropy at most $H(\sigma) \leq \frac{uq}{1000}$ when the committed bit b is chosen at random $b \leftarrow \{0,1\}$. Let u be an upper bound on the number of oracle queries to σ asked by the receiver R in the decommitment phase. Then either of the following holds:*

- **Violation of binding:** *There is a fixed locking oracle $\widehat{\sigma}$, and a sender strategy $\widehat{S}$ such that when $\widehat{\sigma}$ is used as the locking orale, for both $b = 0$ and $b = 1$, $\widehat{S}$ can decommit successfully with probability at least $4/5$.*
- **Violation of hiding:** *There exists an unbounded receiver $\widehat{R}$ who can guess the random bit $b \leftarrow \{0,1\}$ used by (S, σ) with probability at least $4/5$ by asking at most u queries to the locking oracle σ.*

Intuition behind Theorem 4.14. Our main tool in proving Theorem 4.14 is the notion of "canonical entropy learner" (EL) introduced in Definition 4.15. Roughly speaking, EL is an efficient-query (computationally unbounded) algorithm which learns a randomized function f (with an oracle access to f) under the uniform distribution assuming that f has a bounded amount of entropy. EL proceeds by choosing to ask one of the "unbiased" queries of f at any step and stop if such

queries do not exist. An unbiased query x is one whose answer $f(x)$ is not highly predictable with the current knowledge gathered about f by EL. Whenever EL chooses to ask a query it learns non-negligible entropy of f, and thus the process will stop after $\mathrm{poly}(n)$ steps. On the other hand, when EL stops, all the remaining queries are biased and thus will have a predictable answer *over the randomness of* f. We prove that either the receiver is able to find out the secret message of the sender (in an NLS) by running the EL algorithm, or otherwise if by the end of the learning phase still part of the entropy left in the locking oracle is hiding the secret message, then a malicious prover can plant at least two different messages in the locking oracle in such a way that it can decommit to successfully.

Notation. In the following, for clarity, we use bold letters to denote random variables and use the non-bold version of the bold variables to denote the samples of that random variable: $x \leftarrow \mathbf{x}$.

Definition 4.15 (Canonical entropy learner)**.** Suppose $\mathbf{f}$ is a random variable with the support set $\{f \mid f\colon \{0,1\}^n \to \{0,1\}\}$ of all Boolean functions defined over n bits of input. The *canonical entropy learner* $\mathsf{EL}_\epsilon^{\mathbf{f}}$ is an oracle algorithm with access to $\mathbf{f}$ which asks its queries as follows. At any time there is a set $\mathbf{Q}$ of query/answer pairs that $\mathsf{EL}_\epsilon^{\mathbf{f}}$ knows about $\mathbf{f}$. Then if there is any new query x (outside of $\mathbf{Q}$) for which $\epsilon \leq \Pr[\mathbf{f}(x) = 0 \mid \mathbf{Q}] \leq 1-\epsilon$ (in which case we call x an ϵ-unbiased query of $\mathbf{f}$ with respect to $\mathbf{Q}$) then $\mathsf{EL}_\epsilon^{\mathbf{f}}$ asks the lexicographically first such x. $\mathsf{EL}_\epsilon^{\mathbf{f}}$ continues asking as long as there is any ϵ-unbiased queries left. We call a query x $(1-\epsilon)$-biased (with respect to $\mathbf{Q}$), if it is not ϵ-unbiased.

Lemma 4.16. *Suppose $\epsilon < 1/2$ and let the random variable $\mathbf{Q}$ be the set of query/answer pairs that $\mathsf{EL}_\epsilon^{\mathbf{f}}$ learns when interacting with $\mathbf{f}$. Then it holds that $\mathbf{E}[|\mathbf{Q}|] \leq H(\mathbf{f})/\epsilon$ where $H(\cdot)$ is the Shannon entropy.*

Proof. When x is an ϵ-unbiased query of $\mathbf{f}$ with respect to $\mathbf{Q}$, then conditioned on $\mathbf{Q}$, $\mathbf{f}(x)$ has Shannon entropy at least $H(\mathbf{f}(x) \mid \mathbf{Q}) \geq \epsilon \log(1/\epsilon) + (1-\epsilon)\log(1/(1-\epsilon)) > \epsilon \log(1/\epsilon) > \epsilon$. Suppose $x_1, \ldots, x_{|\mathbf{Q}|}$ are the queries asked by $\mathsf{EL}_\epsilon^{\mathbf{f}}$ and $x_{|\mathbf{Q}|+1}, \ldots, x_{2^n}$ are the remaining queries not asked by $\mathsf{EL}_\epsilon^{\mathbf{f}}$ in the lexicographic order, and suppose $\mathbf{Q}_i$ is the set containing the pairs of the queries $x_1, \ldots, x_i$ joint with their answers. Since the information about $x_1, \ldots, x_i$ and $\mathbf{f}(x_1), \ldots, \mathbf{f}(x_i)$ is encoded in $\mathbf{Q}_{i-1}$, therefore it holds that $H(\mathbf{f}) = \sum_{i \in [2^n]} H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1})$. Note that

$$H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1}) = \Pr[|\mathbf{Q}_i| \geq i] H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| \geq i) + \Pr[|\mathbf{Q}_i| < i] H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| < i) \ .$$

We claim that $H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| \geq i) \ \geq \epsilon$. The reason is that we are conditioning on the cases of $\mathbf{Q}_{i-1}$ which determine x_i as an ϵ-unbiased point (and we do not condition on any more information). Therefore it holds that

$$\begin{aligned} H(\mathbf{f}) &= \sum_{i \in [2^n]} H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1}) \\ &\geq \sum_{i \in [2^n]} \Pr[|\mathbf{Q}_i| \geq i] H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| \geq i) \\ &\geq \sum_{i \in [2^n]} \Pr[|\mathbf{Q}_i| \geq i] \epsilon = \epsilon \mathbf{E}[|\mathbf{Q}|], \end{aligned}$$

and so $\mathbf{E}[|\mathbf{Q}|] \leq H(\mathbf{f})/\epsilon$. ∎

Now we turn to proving Theorem 4.14.

Proof. (of Theorem 4.14)

For simplicity, and without loss of generality, we assume that the oracle σ is Boolean and is only defined over inputs of length n.

The attack of $\widehat{R}$ is as follows. The receiver runs the canonical entropy learner $\mathsf{EL}^{\mathbf{f}}_{\epsilon}$ with $\epsilon = 1/100u$ over the (random) function of the oracle σ (where the randomness of σ comes from the randomness r_S shared with the sender which includes the random bit $b \leftarrow \{0,1\}$). Recall that the receiver cannot ask more than u queries. We define $\mathbf{Q}_u$ to be the set of query/answer pairs that the receiver (who asks at most u oracle queries) learns about σ. It always holds that $\mathbf{Q}_u \subseteq \mathbf{Q}$ where $\mathbf{Q}$ is the set of query/answer pairs that $\mathsf{EL}^{\mathbf{f}}_{\epsilon}$ would learn without stopping after u queries. We call a set T of query/answer pairs for σ *predicting* if there exists $c \in \{0,1\}$ such that $\Pr[b = c \mid T] \geq 9/10$. Let Q_u be the value sampled by the random variable $\mathbf{Q}_u$ and Q be the value sampled by the random variable $\mathbf{Q}$. There are two possibilities.

(1) There is a possible Q_u (as a value of $\mathbf{Q}_u$) where Q_u is not predicting and $Q = Q_u$ (i.e. there is no ϵ-unbiased query conditioned on Q_u).

(2) All possible values of Q_u (as the result of the receiver's learning algorithm) are either predicting or $Q \neq Q_u$.

In the following we prove that Case **(1)** implies the violation of binding and Case **(2)** implies the violation of hiding according to Theorem 4.14.

Case (1). First suppose that Case **(1)** holds. In particular there is a possible value Q as the canonical entropy learner's knowledge about σ such that Q is not predicting, namely:

$$1/10 < \Pr[b = 0 \mid Q] < 9/10 \quad \text{and} \quad 1/10 < \Pr[b = 1 \mid Q] < 9/10 \ .$$

Also note that conditioned on Q all the points x out of Q are $(1-\epsilon)$-biased and thus have a unique "likely answer". Namely for each x there is l_x such that $\Pr[\sigma(x) \neq l_x \mid Q] < 1/100u$. Since Q is not predicting, for all x it holds that:

$$\Pr[\sigma(x) \neq l_x \mid Q \wedge b = 0] = \frac{\Pr[\sigma(x) \neq l_x \wedge b = 0 \mid Q]}{\Pr[b = 0 \mid Q]} < \frac{\Pr[\sigma(x) \neq l_x \mid Q]}{1/10} < 1/10u \ .$$

(The same also holds if we condition on $b = 1$.) Therefore in addition to Q, if we condition on $b = 0$ or $b = 1$, still all the points in σ are $(1 - 1/10u)$-biased.

Now we show that the binding property could be violated by a malicious sender and oracle $(\widehat{S}, \widehat{\sigma})$. Consider the following cheating strategy:

1. Distribution of $\widehat{\sigma}$: Sample two oracles along with their (possibly exponentially long) randomness $(\sigma_0, r_0), (\sigma_1, r_1)$ according to the distributions $(\sigma_0, r_0) \leftarrow (\sigma, r_S \mid Q \wedge b = 0)$ and $(\sigma_1, r_1) \leftarrow (\sigma, r_S \mid Q \wedge b = 1)$ and define the oracle $\widehat{\sigma}$ as follows.
   - If $\sigma_0(x) = \sigma_1(x) = l_x$, then $\widehat{\sigma}(x) = \sigma_0(x) = \sigma_1(x)$.
   - If $\sigma_0(x) \neq l_x$ but $\sigma_1(x) = l_x$, then $\widehat{\sigma}(x) = \sigma_0(x)$.
   - If $\sigma_1(x) \neq l_x$ but $\sigma_0(x) = l_x$, then $\widehat{\sigma}(x) = \sigma_1(x)$.

- If $\sigma_0(x) = \sigma_1(x) \neq l_x$, then $\widehat{\sigma}(x) = \sigma_0(x) = \sigma_1(x)$.

2. The algorithm of $\widehat{S}$: To decommit to $b = 0$, the sender $\widehat{S}$ announces $b = 0$ and uses r_0 as the randomness of S to interact in the decommitment phase. To decommit to $b = 1$, $\widehat{S}$ acts similarly by announcing $b = 1$ and using the randomness r_1 in its interaction of the decommitment phase.

Claim 4.17. *If the sender generates the locking oracle $\widehat{\sigma}$ (from σ_0, σ_1) as above and decommits to zero by announcing 0 and using the randomness r_0 in the decommitment phase, then the receiver will accept the interaction with probability at least 9/10.*

Proof. After running the decommitment to zero, we call the experiment a defeat if the receiver asks any query x such that $\sigma_1(x) \neq l_x$, and will call it a win otherwise. Note that if the experiment is a win, then the receiver will accept the decommitment because all of the answers he gets for his queries from $\widehat{\sigma}$ will agree with σ_0.

Now consider another experiment in which $\widehat{\sigma}$ is again generated exactly the same as above, but now σ_0 is used during the decommitment phase. It is clear that now the receiver will accept, but again, we will call the experiment a defeat if the receiver asks any query x (from σ_0) such that $\sigma_1(x) \neq l_x$.

We claim that the probability that the first experiment is a defeat is exactly the same as that of the second experiment. The reason is that, although the receiver in general behaves differently in the two experiments, the difference starts by asking a query x such that $\sigma_1(x) \neq l_x$, and otherwise all the answers given to to the receiver will be the same between $\widehat{\sigma}$ and σ_0.

Now we claim that the probability of defeat in the second experiment is less than 1/10. It would finish the proof since in that case the probability of defeat in the first experiment also will be less than 1/10, and therefore the receiver will accept in the first experiment with probability more than 9/10.

In the second experiment we can pretend that σ_1 is sampled *after* the verification is done using the oracle σ_0. The reason is that the behavior of the receiver is defined only based on σ_0, while σ_1 is only used to determine whether or not the experiment ends in defeat. So suppose $X = \{x_1, \ldots, x_u\}$ be the queries of the receiver before sampling σ_1. Then for each one of $x_i \in X$ it holds that $\Pr[\sigma_1(x) \neq l_x] < 1/10u$ and by the union bound, the probability that for at least one of the $x \in X$, we get $\sigma_1(x) \neq l_x$ is less than 1/10. ∎

By symmetry, the same argument shows that if $\widehat{\sigma}$ is generated as above and the sender pretends that σ_1 is in the oracle and decommits to $b = 1$ (by using r_1) the process leads to the accept of the receiver R with probability more than 9/10.

We call a *fixed* $(\widehat{\sigma}, r_0, r_1)$ generated as above a good sample for $b = 0$ if the decommitment of $\widehat{S}$ to $b = 0$ as above leads to accept with probability at least 4/5. A simple average argument shows that the sampled $(\widehat{\sigma}, r_0, r_1)$ is good for $b = 0$ with probability more than 1/2. A similar argument shows that the sampled $(\widehat{\sigma}, r_0, r_1)$ is also good for $b = 1$ with probability more than 1/2. By the pigeonhole principle there is a fixed $(\widehat{\sigma}, r_0, r_1)$ which is both good for $b = 0$ and $b = 1$. Using this $(\widehat{\sigma}, r_0, r_1)$ the sender can decommit to both $b = 0$ and $b = 1$ and succeed with probability at least 4/5.

Case (2). If Case **(2)** holds, then we claim that the receiver R can guess the bit b with probability at least $4/5$. The reason is that by Lemma 4.16 it holds that $\mathbf{E}[|\mathbf{Q}|] \leq H(\sigma)/\epsilon \leq u/10$. Therefore by Markov inequality, with probability at least $9/10$ it holds that $|\mathbf{Q}| \leq u$ in which case $Q_u = Q$. Moreover since Case **(2)** holds, whenever $Q = Q_u$, then Q is predicting and the receiver R can guess b correctly with probability at least $9/10$. Therefore the receiver R would guess b correctly with probability at least $(9/10)(9/10) > 4/5$ which is a violation of hiding according to Theorem 4.14. ■

5 The Impossibility of Oblivious Transfer

In this section we prove that in the stateless token model, there is no statistically secure protocol for Oblivious Transfer (OT), when the only limitation on malicious parties is being bounded to make polynomially many queries to the tokens. We first define the (bit) oblivious transfer functionality. For simplicity we use a definition with completeness one, but our negative result easily extends to protocols with imperfect completeness.

The stateless token model. In the stateless (tamper-proof hardware) token model, two (computationally unbounded) interactive algorithms A and B will interact with the following extra feature to the standard model. Each party at any time during the protocol can construct a circuit T and put it inside a "token" and send the token T to the other party. The party receiving the token T will have *oracle access* to T and is limited to ask $\text{poly}(n)$ number of queries to the token. The parties can exchange $\text{poly}(n)$ number of tokens during the interaction. The stateless token model clearly extends the IPCP model in which there is only one token sent from the prover to the verifier in the beginning of the game. Therefore proving any *impossibility* result in the stateless token model clearly implies the same result for the the IPCP model. It is easy to see that without loss of generality the parties can avoid sending "explicit messages" to each other and can only use tokens (with messages planted inside the tokens) to simulate all the classical communication with the tokens.

Definition 5.1 (Oblivious Transfer)**.** A protocol (S, R) for oblivious transfer (in the standard or the token model), consists of a sender S and a receiver R. The parties both receive 1^n where n is the security parameter. The sender S holds the secret input $(x_0, x_1), x_i \in \{0,1\}$, and the receiver R holds the input $i \in \{0,1\}$. At the end of the protocol, the receiver outputs y where (with probability one) it holds that $y = x_i$. We define the following security measures.

- **Receiver's security:** This property guarantees that the sender is not able to find out which of x_0 and x_1 is read by the receiver. Formally, for any malicious sender $\widehat{S}$ who asks $\text{poly}(n)$ token queries and outputs j, it holds that $\Pr_{i \leftarrow \{0,1\}}[i = j] \leq 1/2 + \text{neg}(n)$.
- **Sender's security:** This property guarantees that no receiver can read both of x_0 and x_1. Formally, for any malicious receiver $\widehat{R}$ who asks $\text{poly}(n)$ token queries, with probability $1 - \text{neg}(n)$ the view $v_{\widehat{R}}$ of $\widehat{R}$ satisfies either of the following:

$$\Pr_{(x_0,x_1)\leftarrow\{0,1\}^2}[x_0 = 0 \mid v_{\widehat{R}}] = \frac{1}{2} \pm \text{neg}(n) \quad \text{or} \quad \Pr_{(x_0,x_1)\leftarrow\{0,1\}^2}[x_1 = 0 \mid v_{\widehat{R}}] = \frac{1}{2} \pm \text{neg}(n).$$

Oblivious transfer by semi-honest parties. If one of the parties is semi-honest (i.e. runs the protocol honestly, and only remember's its view for further off-line investigation), then in fact unconditionally secure OT *is* possible in the stateless token model. If the receiver is honest, then the protocol is simply a token T sent from the sender which encodes $T(0) = x_0, T(1) = x_1$. The receiver will read $T(i)$ to learn x_i. Moreover it is well known that secure OT in one direction implies the existence of secure OT in the other direction, so if the sender is semi-honest unconditionally secure OT is possible in the stateless token model.

In this section we prove that statistically secure OT is impossible in the stateless token model, if both parties are *slightly* more malicious than just being semi-honest. Roughly speaking, we define the notion of "curious" parties who run the original protocol (honestly), but will ask more queries from the tokens along the way.[7] We will prove that for any protocol (A, B) aiming to implement OT, there are curious extensions of the original parties $(A_{\mathsf{cur}}, B_{\mathsf{cur}})$ who break the security of the protocol. More formally we define curious parties as follows.

Definition 5.2. Let (A, B) be a two party protocol in the token model. We call the protocol A_{cur} a *curious* extension of A, if A_{cur} runs the same protocol as A, but at any point during the protocol it might ask arbitrary queries from tokens sent from B. We call A_{cur} efficient, if it asks only $\mathrm{poly}(n)$ queries totally. By $Q_{A_{\mathsf{cur}}}$ we denote the set of triples representing total query/answer information that A_{cur} gathers about the tokens it receives, and we use Q_A to denote only the query/answers for the underlying emulated algorithm A (and so $Q_A \subseteq Q_{A_{\mathsf{cur}}}$).

We will prove the following theorem.

Theorem 5.3 (No unconditional OT from stateless tokens)**.** *Let (S, R) be any protocol for the oblivious transfer in the stateless token model. Then there are curious extensions $(S_{\mathsf{cur}}, R_{\mathsf{cur}})$ to the original algorithms where $(S_{\mathsf{cur}}, R_{\mathsf{cur}})$ (and thus (S, R)) is not a secure protocol for oblivious transfer even when the inputs are random. More formally either of the following holds:*

- **Violation of sender's security:** *When the sender S chooses x_0 and x_1 at random from $\{0,1\}$ and interacts with R_{cur}, then R_{cur} can find out both of x_0 and x_1 with probability at least $51/100$.*
- **Violation of receiver's security:** *When the receiver R chooses $i \leftarrow \{0,1\}$ at random and interacts with S_{cur}, then S_{cur} can guess i correctly with probability at least $51/100$.*

In the following section we first show that any protocol (S, R) with the following sampling condition can not implement OT securely (Lemma 5.5). The sampling condition is that one can sample the randomness of the receiver R conditioned on the view of the sender S. This sampling condition extends the "accessible entropy" notion introduced by Haitner et al [HHRS07] to the token model (in an information theoretic way) and is equivalent to saying that the entropy of the receiver is accessible. Then we show (Theorem 5.8) that for any protocol (S, R), there is a curious extension $S_{\mathsf{cur}}, R_{\mathsf{cur}}$ where almost all the entropy is accessible, and this will prove Theorem 5.3.

[7] The term "honest but curious" is sometimes used equivalent to "semi-honest". Our notion is different from both of them because a curious party deviates from the protocol slightly by learning more but emulates the original protocol honestly.

Notation. In the following, for clarity, we use bold letters to denote random variables and use the non-bold version of the bold variables to denote the samples of that random variable: $x \leftarrow \mathbf{x}$. For example $\mathbf{r}_A$ will denote the randomness of Alice as a random variable and r_A denotes the actual randomness used by Alice.

5.0.1 Breaking oblivious transfer by accessing the entropy

In the standard model of interaction (without tokens) all the information exchanged between two parties is represented in the transcript τ of the interaction. This means that information theoretically, an unbounded Bob can sample from $r \leftarrow (\mathbf{r}_B \mid \tau)$: his own space of randomness conditioned on what Alice knows about it (which is the transcript of the protocol τ) and vice versa. So information theoretically Bob is not committed to anything more than what Alice knows about him (but of course Bob might not be able to sample from this distribution *efficiently*). Roughly speaking if the sampling $r_B \leftarrow (\mathsf{sup}(\mathbf{r}_B) \mid \tau)$ can be done efficiently at any time during the protocol, [HHRS07] calls such protocol one with "accessible entropy".

The fact that in the standard model unbounded parties can access all the entropy makes unconditional oblivious transfer to be impossible in this model. The reason is that at the end of the protocol, either the sender S can find out which bit is read by the receiver R by sampling from $r \leftarrow (\mathbf{r}_R \mid \tau)$, or otherwise a sample from $r \leftarrow (\mathbf{r}_R \mid \tau)$ will reveal either of the inputs x_0, x_1 with probability $\approx 1/2$ and thus Bob can find out both of them by taking enough number of samples. We will formalize this argument in the case of token model in Lemma 5.5 below.

In the token model, however, the information exchanged between the parties is not "shared" by them. Namely, the set Q_A (see Definition 5.2) captures the information that the algorithm A gathers from the tokens he has received from the other party B, and this set is different from Q_B. Moreover, it might be impossible for A to find out Q_B exactly. Therefore if we consider $\tau = (Q_A, Q_B)$ to be the transcript of the protocol in the token model, Alice and Bob each know part of τ, and this makes the above argument (for the impossibility of OT) fail. In fact, as we said, if either of the parties is semi-honest, then unconditional OT *is* possible in the stateless token model.

The following definition generalizes the notion of accessible entropy to the token model (from an information theoretic perspective).

Definition 5.4 (Accessible entropy in the token model)**.** Let (A, B) be a protocol in the token model between Alice and Bob. We say the entropy of Bob can be $(1-\delta)$-accessed if there is a sampling algorithm Sam_B as follows. Sam_B receives a possible view for Bob (r, Q) as input and outputs another possible view for Bob (r', Q') with the following property: With probability at least $1-\delta$ over the choice of $(r_A, r_B) \leftarrow (\mathbf{r}_A, \mathbf{r}_B)$, if Q_A, Q_B denote to the information that Alice and Bob has gathered from the tokens *at any time during* the execution of the protocol, it holds that

$$\mathsf{SD}((\mathbf{r}_B, \mathbf{Q}_B \mid r_A, Q_A), \mathsf{Sam}_B(r_B, Q_B)) \leq \delta.$$

We say that B has δ inaccessible entropy, if the entropy of B is not $(1-\delta)$-accessible.

Comments about Definition 5.4 .

One can define a computational version of Definition 5.4 in the standard model where Sam_B is efficient and given τ it samples from a distribution which is δ-close to $(\mathbf{r}_B \mid \tau)$. It can be shown

that an interactive algorithm B has non-negligible inaccessible entropy in this definition if and only if B has non-negligible inaccessible entropy according to the definition of [HHRS07].

[HHRS07] shows that in the standard computational setting non-negligible inaccessible entropy implies statistically hiding commitments. We shall point out that in the stateless token model, as we will see, if both parties are "curious enough", then almost all the entropy can be accessed, but as we saw in Theorem 4.1 unconditionally secure commitment schemes *does* exist. The difference between the standard computational model of interaction and the token model is that in the standard computational model the sampled $r \leftarrow (\mathbf{r}_B \mid \tau)$ can be used (by a potentially malicious Bob) as a witness that Bob is running the protocol appropriately and even be used to continue the protocol without being caught by Alice, but in the token model a sampled $r = \mathsf{Sam}_B(r_B, Q_B)$ might contradict the future queries of Alice to the tokens she that is holding from Bob. The fact that Alice can caught Bob's claim by asking more token queries is highly used in our both Constructions 4.4 and 4.10 of ILS's. In fact, if one defines a *simple commitment* scheme as one where the receiver is not allowed to ask token queries during the decommitment phase, then a similar argument to Lemma 5.5 below shows that simple commitment schemes imply (non-negligible) inaccessible entropy, and therefore unconditional simple commitments are impossible to achieve in the stateless token model.

Lemma 5.5 (Oblivious transfer needs inaccessible entropy)**.** *Let (S, R) be a protocol for* random *oblivious transfer where the sender's inputs x_0, x_1 are part of r_S and receiver's input i is part of r_B. If (S, R) has perfect completeness and the entropy of the receiver R can be* 0.99*-accessed, then either of the following holds:*

- **Violation of sender's security:** *When the sender S interacts with R, then R can find out both of x_0 and x_1 correctly with probability at least* 51/100*.*
- **Violation of receiver's security:** *When the receiver R chooses $i \leftarrow \{0, 1\}$ at random and interacts with S, then S can guess i correctly with probability at least* 51/100*.*

Proof. We show that if Sam_R 1-accesses the entropy, then either of the security failures above holds with probability at least 53/10. Using the actual Sam_R (which 0.99-accesses the entropy) will change the success probability of the attacks by at most 2/100 which still remain at least 51/100. So in the following we assume that Bob also can sample from the distribution $(\mathbf{r}_R \mid r_S, Q_S)$ where by Q_S we denote the final set of query/answers that the sender learns about the receiver's tokens. We call (r_S, Q_S) a predicting pair, if either $\Pr[i = 0 \mid r_S, Q_S] \geq 9/10$, or $\Pr[i = 1 \mid r_S, Q_S] \geq 9/10$. Now there are two possible cases:

- $\Pr[(r_S, Q_S)$ is predicting$] \geq 1/10$: Then the sender can predict i by probability at least $1/2 + 1/10 > 53/10$ as follows: Whenever (r_S, Q_S) is predicting, use the predicted value, and whenever it is not use a random guess. This violates the security of the receiver.
- $\Pr[(r_S, Q_S)$ is predicting$] \leq 1/10$: Therefore, with probability at least 9/10 (r_S, Q_S) is not predicting. For a non-predicting (r_S, Q_S), by sampling $(r'_R, Q'_R) \leftarrow (\mathbf{r}_R, \mathbf{Q}_R \mid r_S, Q_S)$ we get either of the cases $i = 0$ or $i = 1$ with probability at last 1/10. By the assumed perfect completeness of the protocol the output of the sampled (r'_R, Q'_B) is equal to x_i. Note that sender can sample from $(\mathbf{r}_R, \mathbf{Q}_R \mid r_S, Q_S)$ by using the sampler Sam_R. When (r_S, Q_S) is not predicting, by sampling n samples $(r'_R, Q'_R) \leftarrow (\mathbf{r}_R, \mathbf{Q}_R \mid r_S, Q_S)$, the receiver can find both of x_0 and x_1 with probability $1 - \mathrm{neg}(n)$. Therefore, since (r_S, Q_S) is not predicting with probability 9/10, the receiver can find both inputs of the sender with probability at least $(9/10) \cdot (9/10 - \mathrm{neg}(n)) > 53/100$.

■

5.0.2 Curious extensions who access the entropy

In this section we define a specific curious extension to the protocols in the stateless token model which we call "canonical" curious extensions and prove that almost all the entropy of such protocols can be accessed. Roughly speaking, a canonical curious extension of an interactive algorithm is a curious extension of the original protocol where the parties ask their "extra" token queries according to the canonical entropy learner of Definition 4.15. Each time that an unbiased query q is asked, it reveals a noticeable amount of information about the other party's private randomness. By the same argument as Lemma 4.16, since the entropy $H(\mathbf{r}_A) + H(\mathbf{r}_B) \leq \text{poly}(n)$ is bounded, by asking only efficient number of queries one can learn enough information to predict the answer to any other (fixed) query to the tokens. In particular, the answer to the query that is going to be asked in the emulation of the original protocol will be predictable, so we can ignore asking such queries and use the predicted answers in the emulation of the original protocol. We call the new protocol which uses the predicted values an "ideal" protocol $(A_{\mathsf{id}}, B_{\mathsf{id}})$ which will simulate the original protocol (A, B) up to $1/\text{poly}(n)$ statistical distance.

Finally, the main point is that in the ideal protocol the parties know *exactly* what the other party knows about them! The proof uses induction. Assuming that Alice and Bob both know $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$, they can find out which queries are ϵ-unbiased from Alice's and Bob's point of view. When Alice is asking such a query Bob knows the query *and* its answer (since it is from a token generated by Bob) and vice versa. Therefore, $(r_{A_{\mathsf{id}}}, Q_{A_{\mathsf{id}}})$ will determine $Q_{B_{\mathsf{id}}}$ and $(r_{B_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$ will determine $Q_{A_{\mathsf{id}}}$ and in a sense we are back to the standard model where the transcript is known to both parties, and thus all the entropy can be accessed!

It will be easier for us to first define the ideal experiment Ideal and then define how the canonical curious parties behave.

Definition 5.6 (Ideal experiment.)**.** Let (A, B) be a protocol in the stateless token model. The protocol $(A_{\mathsf{id}}, B_{\mathsf{id}})$ will depend on (A, B) and will be executed in the experiment Ideal as follows.

- **Randomness:** $(A_{\mathsf{id}}, B_{\mathsf{id}})$ will use the same randomness (r_A, r_B) and will (try to) emulate (A, B).
- **Set of token query/answers:** $(A_{\mathsf{id}}, B_{\mathsf{id}})$ will inductively update the sets $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ which encode the information they have gathered from the other party's tokens so far.
- **Mutual Knowledge:** A crucial point is that $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ will be known to *both* Alice and Bob inductively. Namely, A_{id} can find out the updates to the set $Q_{B_{\mathsf{id}}}$ without B_{id} explicitly revealing that information to A_{id} and vice versa. How this condition holds and how the emulation of (A, B) is done is clarified next.
- **Emulation:** Suppose during the emulation we are at a time where A needs to ask the query q_0 from a token T. Instead of doing so, A_{id} runs the canonical entropy learner of Definition 4.15 over the tokens $(T_1^B, \ldots, T_i^B)$ that he holds from Bob. Namely, A_{id} considers the uniform distribution of $(\mathbf{r}_A, \mathbf{r}_B)$ consistent with the "transcript" of the interaction: $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$. Note that, by induction's assumption A_{id} knows $Q_{B_{\mathsf{id}}}$ as well. Now if there is any query q to any of the tokens $(T_1^B, \ldots, T_i^B)$ sent from Alice such that q is ϵ-unbiased, then A_{id} asks the lexicographically first such query q_1 and updates $Q_{A_{\mathsf{id}}}$. We stress that since B_{id} knows also

$Q_{A_{\mathsf{id}}}$, he is also able to find the query q_1, and moreover since this query is asked to one of the tokens that Bob has generated, B_{id} knows the answer as well. So Bob also knows the update to the set $Q_{A_{\mathsf{id}}}$ the same as Alice. When A_{id} is done with the learning phase before asking q_0, it *not* ask the query q_0, but rather will use the likely answer l_{q_0} which is uniquely determined by the updated $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$ (but of course l_{q_0} might be different from the actual solution provided by the token). B_{id} will emulate B similarly.

Now we define the experiment CanCur in which a canonical curious extension of the protocol (A, B) is defined.

Definition 5.7 (Canonical curious extensions)**.** Let (A, B) be a two party protocol in the stateless token model. A *Canonical Curious* extension of A denoted by A_{cc} receives two parameters ϵ: the curiosity parameter and u: the upper bound on the number of extra queries asked by A. A_{cc} acts in the experiment CanCur as follows:

- A_{cur} runs the algorithm A with randomness r_A and emulates the interaction of A with B. The set Q_A denotes the set of token query/answers corresponding to this emulation.
- A_{cur} also keeps updating a set of token query/answers $Q_{A_{\mathsf{id}}}$ by *pretending* that it is being executed in the experiment Ideal. A_{cur} stops asking more extra queries (i.e. the queries out of Q_A) whenever $|Q_{A_{\mathsf{id}}}| = u$. Note that in CanCur (as opposed to CanCur), A_{cc} *does* ask the emulated queries of A from the tokens and uses the answer for further computations.
- The total token queries/answer knowledge of A_{cur} is encoded in $Q_{A_{\mathsf{cur}}} = Q_A \cup Q_{A_{\mathsf{id}}}$.

Now we prove the following theorem showing that by taking ϵ small enough, one can access almost all the entropy of $(A_{\mathsf{id}}, B_{\mathsf{id}})$ efficiently.

Theorem 5.8 (Canonical curious parties can access the entropy)**.** *Let (A, B) be a protocol in the stateless token model between Alice and Bob where they totally ask $k = \mathrm{poly}(n)$ number of queries to the tokens exchanged. For a given $\delta = 1/\mathrm{poly}(n)$ let ϵ be such that $\delta = 2(\epsilon k + \sqrt{\epsilon k})$ and let $u = \frac{|r_A|}{k\epsilon^2}$. If ϵ is used as the curiosity parameter and u is used as the upper-bound parameter, then the entropy of B_{cc} in $(A_{\mathsf{cc}}, B_{\mathsf{cc}})$ is $(1-\delta)$-accessible.*

Note that Theorem 5.8 and Lemma 5.5 together imply Theorem 5.3 as a corollary.

Intuition. Roughly speaking Theorem 5.8 follows from the following facts:

1. **Entropy is accessible in Ideal:** The entropy of A_{id} is 1-accessible, because $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ are known to both parties at any time during the protocol.
2. **Parties are efficient in Ideal:** By Lemma 4.16 A_{id} on average asks at most $H(\mathbf{r}_B)/\epsilon = |\mathbf{r}_B|/\epsilon \leq \mathrm{poly}(n)$ number queries.
3. **Ideal and CanCur are close:** As long as the likely answers used by A_{id} and B_{id} for the emulation of (A, B) are correct, the two experiments Ideal and CanCur are *exactly* the same. Let WG (Wrong Guess) be the event that one of these likely answers used is incorrect. By definition, he probability that WG happens at any particular query in Ideal is at most ϵ, and therefore the probability of WG is bounded by $\epsilon \cdot k$. By taking ϵ small enough we can make $\epsilon \cdot k$ arbitrary small.

The formal proof follows.

Proof. (of Theorem 5.8) The sampling algorithm Sam_B in CanCur will perform the sampling by simply pretending that it is being used in the experiment Ideal (by ignoring Q_B and only using $Q_{B_{\mathsf{id}}}$ and r_B). We will show that with probability $1-\delta$ over $(\mathbf{r}_A, \mathbf{r}_B)$ the sampler Sam_B will sample δ-close to the right distribution all along the execution of CanCur.

As a mental experiment, we sample $(r_A, r_B) \leftarrow (\mathbf{r}_A, \mathbf{r}_B)$ and run *both* of the experiments CanCur and Ideal in parallel with the same randomness (r_A, r_B). We use the notation $\Pr_{\mathsf{cc}}[\cdot]$ to denote the probability of an event in CanCur and use $\Pr_{\mathsf{id}}[\cdot]$ to denote the probability of an event in Ideal.

Let WG_i to be the event in the experiments CanCur and Ideal that holds iff the i'th query q_i asked by Alice or Bob in is *not* the likely answer, and define the event $\mathsf{WG} = \bigvee_i \mathsf{WG}_i$.

We modify CanCur slightly and let the size of $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ to grow beyond u. Our proof will take care of this issue indirectly. By this change, it is easy to see that as long as WG does not happen the experiments CanCur and Ideal are exactly the same. In particular it holds that $\Pr_{\mathsf{cc}}[\mathsf{WG}] = \Pr_{\mathsf{id}}[\mathsf{WG}]$. Since (r_A, r_B) determines the whole execution of the experiments CanCur and Ideal we can talk about the event $\mathsf{WG}(\mathbf{r}_A, \mathbf{r}_B)$, and $\Pr_{(\mathbf{r}_A,\mathbf{r}_B)}[\mathsf{WG}]$ is the same in CanCur and Ideal.

The following claim shows that by taking ϵ small enough CanCur and Ideal will become statistically close.

Claim 5.9. *It holds that* $\Pr_{(\mathbf{r}_A,\mathbf{r}_B)}[\mathsf{WG}] \leq k\epsilon$.

Proof. If show that $\Pr_{\mathsf{id}}[\mathsf{WG}_i] \leq \epsilon$ the claim follows by union bound over $i \in [k]$. But $\Pr_{\mathsf{id}}[\mathsf{WG}_i] \leq \epsilon$ holds by the definition of the experiment Ideal and ϵ-unbiased queries. ■

We call (r_A, r_B) a *good sample* if the following conditions hold when (r_A, r_B) is used to run the experiments:

1. WG does not happen.
2. $|Q_{B_{\mathsf{id}}}|$ and $|Q_{B_{\mathsf{id}}}|$ does not pass the upper limit u.
3. At any time during the execution if $Q_{A_{\mathsf{id}}}$, $Q_{B_{\mathsf{id}}}$, $Q_{A_{\mathsf{cc}}}$, and $Q_{B_{\mathsf{cc}}}$ denote the set of query/answers in the both experiments in that moment it holds that

$$\Pr_{\mathsf{id}}[\mathsf{WG} \text{ happened before } \mid Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}}] \leq \sqrt{k\epsilon} \quad \text{and}$$
$$\Pr_{\mathsf{cc}}[\mathsf{WG} \text{ happened before } \mid Q_{A_{\mathsf{cc}}}, Q_{B_{\mathsf{cc}}}] \leq \sqrt{k\epsilon}.$$

Claim 5.10. $\Pr[(\mathbf{r}_A, \mathbf{r}_B) \textit{ is good }] \geq 1-\delta$.

Proof. By Claim 5.9 the first property holds with probability at least $1-k\epsilon$.

By Lemma 4.16 and using the Markov inequality the second property holds in Ideal with probability at least $1-k\epsilon$. So both properties 1 and 2 hold with probability at least $1-2k\epsilon$.

We use Lemma 2.3 with the following parameters $X = (\mathbf{r}_A, \mathbf{r}_B), F = \mathsf{WG}$, $Z_i = (\mathbf{Q}_{A_{\mathsf{id}}}, \mathbf{Q}_{B_{\mathsf{id}}})$ where $(\mathbf{Q}_{A_{\mathsf{id}}}, \mathbf{Q}_{B_{\mathsf{id}}})$ their value at the i'th moment and $p = \sqrt{k\epsilon}$. Now Lemma 2.3 implies that the third property holds for Ideal (and similarly for CanCur) with probability at least $1-\sqrt{k\epsilon}$.

Therefore all the properties hold for a random (r_A, r_B) with probability at least $1-2k\epsilon-2\sqrt{k\epsilon} = 1-\delta$. ■

Suppose the sampled (r_A, r_B) is a good one. Then, by the fist property, all along the execution CanCur and Ideal will be the same. The second property will guarantee the efficaciously with respect to the upper bound u.

Now suppose Sam_B is run over the input $(r_B, Q_{B_{\mathsf{cc}}})$ at an arbitrary moment during the execution of the system. By the first property, the sampler Sam_B will conclude the correct $Q_{A_{\mathsf{cc}}}$. Therefore Sam_B will sample exactly according to $(\mathbf{r}_{B_{\mathsf{id}}}, \mathbf{Q}_{A_{\mathsf{id}}} \mid r_{A_{\mathsf{id}}}, Q_{A_{\mathsf{id}}})$ in Ideal. By the third property the probability that WG has happened before is at most $\sqrt{k\epsilon}$ in either of CanCur or Ideal. Finally we note that if one samples $(r_B, Q_{B_{\mathsf{cc}}})$ conditioned on $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$ *and* conditioned on $\neg\mathsf{WG}$, the samples have the same distribution in Ideal and CanCur. This means that if (r_A, r_B) is good, the sampler Sam_B will sample $2\sqrt{k\epsilon} < \delta$-close to the right distribution. This finishes the proof that Sam_B $(1-\delta)$-accesses the entropy of A_{cc} in $(A_{\mathsf{cc}}, B_{\mathsf{cc}})$. ■

6 Unconditional UC Secure Computation: Encapsulating Tokens inside Tokens

6.1 The Model

First we define the functionality $\mathcal{F}^{\mathsf{enc}}$ that models the fact that the parties can put any number of received tokens inside a new one. The idea of $\mathcal{F}^{\mathsf{enc}}$, described in Figure 1, is to model the following real-world functionality: party P_i sends a stateless token M to party P_j. This stateless token may have encapsulated inside it a number of other stateless tokens which P_i received from other parties (such encapsulated tokens subsequently become unavailable to P_i after sending out M). Since the token is stateless, P_j can run M multiple times on inputs of its choice. While running, M can access its internal tokens in a black box way by querying any of them any number of times. Thus, $\mathcal{F}^{\mathsf{enc}}$ saves the description of the oracle machines it gets from a party in create messages, and lets the other party run them multiple times. Each machine is uniquely identified by a machine identifier mid.

6.2 Construction for Unconditional UC Secure Computation

We first show how to construct an unconditional UC secure commitment scheme COM in the token encapsulation model described above. We then present a construction for unconditional UC secure OT building on the commitment scheme COM. This in turn implies the existence of unconditional general UC secure computation by invoking the results from [IPS08, Kil88].

Unconditional UC Secure Commitment The committer Alice wishes to commit to a bit b to the receiver Bob. We build upon the interactive locking scheme in Section 4. We can get a commitment scheme in the stateless hardware token model using an interactive locking scheme where the sender is the committer and the locking oracle is implemented using a stateless hardware token. We call this commitment scheme *com*. The commit phase of our commitment scheme COM proceeds as follows:

- Bob first generates a random string $r \leftarrow \{0,1\}^k$ and uses the commitment scheme *com* to commit r to Alice.

Functionality $\mathcal{F}^{\mathsf{enc}}$

$\mathcal{F}^{\mathsf{enc}}$ is parameterized by a polynomial $p(\cdot)$ and an implicit security parameter κ.

Create Upon receiving $(\mathsf{create}, \mathsf{sid}, P_i, P_j, \mathsf{mid}, M, List)$ from P_i, where M is an Oracle machine, do:

1. If any of the tokens specified in $List$ is marked unavailable or does not have P_i as the recipient, do nothing. Else, mark all tokens in $List$ as unavailable. (This step ensures that a received token cannot be used in constructing multiple other tokens.)
2. Send $(\mathsf{create}, \mathsf{sid}, P_i, P_j, \mathsf{mid})$ to P_j.
3. Store $(P_i, P_j, \mathsf{mid}, M, List)$.

Execute Upon receiving $(\mathsf{run}, \mathsf{sid}, P_i, \mathsf{mid}, \mathsf{msg})$ from P_j, find the unique stored tuple $(P_i, P_j, \mathsf{mid}, M)$. If no such tuple exists or if this token is marked unavailable, do nothing. (This step ensures that a received token cannot be queried after it has been "used up" in constructing another token.)
Run $M(\mathsf{msg})$. While running if M makes a query specifying mid, check if mid is in $List$ associated with M. If not, output $\perp$.
Else, evaluate the machine mid on the given query recursively and supply the response to M. Let out be the output of M ($\mathsf{out} = \perp$ if the entire computation does not halt in $p(k)$ steps). Send $(\mathsf{sid}, P_i, \mathsf{mid}, \mathsf{out})$ to P_j.

Figure 1: Ideal functionality for stateless tokens with encapsulation

- Alice breaks the bit b into k pairs of 2-out-of-2 random shares. That is, for all $i \in [k]$, Alice picks b_i^0 and b_i^1 at random such that $b_i^0 \oplus b_i^1 = b$. Now $\forall i, j$, Alice commits b_i^j to Bob using the commitment scheme *com*.

- Alice further construct a token T as follows. Alice puts her entire state in T including all tokens received from Bob, the input and randomness and the protocol transcript so far. The functionality of T is described in the next step.

- The token T expects as input from Bob a string r' that Bob committed to in the first step along with the decommitment to r' (i.e., the randomness he used to commit to r' in the first step). T verifies that the decommitment is valid by possibly querying the encapsulated tokens Alice received from Bob in the first step. If the check fail, T outputs $\perp$. Otherwise, $\forall i \in [k]$, T outputs $b_i^{r'[i]}$ and the decommitment to $b_i^{r'[i]}$ (where $r'[i]$ represents the i-th bit of the string r').

- Bob queries T with the string r and its decommitment and gets in return, $\forall i \in [k]$, $b_i^{r[i]}$ and the decommitment to $b_i^{r[i]}$. Bob verifies all the decommitments (by possibly querying the tokens he received from Alice in the second step. Bob aborts and rejects the commitment in case any of the decommitments is found invalid.

The decommitment phase for COM simply consists of Alice revealing b_i^j and its decommitment $\forall i \in [k], j \in \{0,1\}$ along with the bit b. Bob verifies that all the decommitments are valid and that $\forall i$, $b_i^0 \oplus b_i^1 = b$. If so, Bob accepts the decommitment to bit b and rejects otherwise.

Proof Sketch. The above protocol can be shown to be UC secure using standard techniques. Below we only provide the main idea behind the proof and defer the details to the full version.

Corrupted Committer: In this case, the simulator Sim works by extracting the input of the committer and sending it to the ideal trusted functionality. This can be done using the following idea. The simulator runs the commit phase honestly with the committer. At the end of the commit phase, Sim utilizes the equivocability property of the interactive locking scheme to make additional queries to the token T and get a valid response (thus enabling him to extract b). In other words, Sim chooses a random string r_1 and sends a decommitment to r_1. Sim uses the simulator associated with the equivocability property of ILS to generate such a valid decommitment to r_1 (and answer the queries made by T to the encapsulated token). If T gives a valid answer, Sim computes the committed bit b (and sends it to the ideal functionality). Else, Sim tries again with another random string. Sim will terminate in expected polynomial time since T gave a valid response on the first (honest) query made by Sim.

Corrupted Receiver: In this case, Sim executes the commitment phase honestly and commits to a random bit. At the start of the decommit phase, Sim gets the bit b to be decommitted to from the ideal functionality. Then Sim, for each of the k pair of shares, chooses the unopened share to be such that the XOR of the two shares equals b. Next, Sim uses the equivocability property of the interactive locking scheme to decommit all his commitments (in step 2 of the protocol) to the appropriate shares as chosen.

Unconditional UC Secure Oblivious Transfer The sender Alice has the input strings (s_0, s_1) while the receiver Bob has the selection bit c. The OT protocol proceeds as follows:

- Bob commits to its selection bit c using the commitment scheme COM.
- Alice constructs a token G as follows. Alice puts her entire state in G including all tokens received from Bob, the input and randomness and the protocol transcript so far. The functionality of G is described in the next step.
- The token G expects as input from Bob a string c' that Bob committed to in the first step along with the decommitment to c' (i.e., the randomness he used to commit to c' in the first step). G verifies that the decommitment is valid by possibly querying the encapsulated tokens Alice received from Bob in the first step. If the check fail, G outputs $\perp$. Otherwise, G outputs the string $s_{c'}$.
- Bob queries G with the string c and its decommitment and gets in return s_c as his output of the protocol. If G aborts without giving a response, Bob takes 0 to be received output.

Proof Sketch. We consider the following two cases separately.

Corrupted Sender: In this case, the simulator Sim works by extracting the input strings of the sender and sending them to the ideal trusted functionality. This can be done as follows. Sim commits to a random choice bit in the first step and receives the token G. Sim then simply uses the equivocation property of the commitment scheme COM to query the token G on both 0 and 1 as the choice bits and get the response (if G aborts in any query, Sim takes 0 to be response). Sim then sends the two extracted strings to the ideal functionality.

Corrupted Receiver: In this case, Sim simply extracts the choice bit c of the receiver in the first step. This can be done by using the simulator associated with the commitment scheme COM. Sim then sends the extracted choice bit c to the trusted functionality and gets the output s_c. Sim then picks $s_{\bar{c}}$ at random and honestly prepares the token G with $\{s_c, s_{\bar{c}}\}$ as the input strings.

References

[AH91] William Aiello and Johan Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *J. Comput. Syst. Sci.*, 42(3):327–345, 1991.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of np. *J. ACM*, 45(1):70–122, 1998.

[BFL90] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. In *FOCS*, pages 16–25, 1990.

[BGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *STOC*, pages 113–131, 1988.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[BM88] László Babai and Shlomo Moran. Arthur-Merlin games: A randomized proof system and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36:254–276, 1988.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, pages 11–19, 1988.

[CCM98] Christian Cachin, Claude Crépeau, and Julien Marcil. Oblivious transfer with a memory-bounded receiver. pages 493–502, 1998.

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *STOC*, pages 235–244, 2000.

[CGS08] Nishanth Chandran, Vipul Goyal, and Amit Sahai. New constructions for UC secure computation using tamper-proof hardware. In *EUROCRYPT*, pages 545–562, 2008.

[CK88] Claude Crépeau and Joe Kilian. Achieving oblivious transfer using weakened security assumptions (extended abstract). In *FOCS*, pages 42–52, 1988.

[DFK+92] Cynthia Dwork, Uriel Feige, Joe Kilian, Moni Naor, and Shmuel Safra. Low communication 2-prover zero-knowledge proofs for np. In *CRYPTO*, pages 215–227, 1992.

[DHRS04] Yan Zong Ding, Danny Harnik, Alon Rosen, and Ronen Shaltiel. Constant-round oblivious transfer in the bounded storage model. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004*, pages 446–472, 2004.

[For89] Lance Fortnow. The complexity of perfect zero-knowledge. *Advances in Computing Research: Randomness and Computation*, 5:327–343, 1989.

[FRS88] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. In *Theoretical Computer Science*, pages 156–161, 1988.

[GIS+10] Vipul Goyal, Yuval Ishai, Amit Sahai, Ramarathnam Venkatesan, and Akshay Wadia. Founding cryptography on tamper-proof hardware tokens. In *TCC*, 2010.

[GKR08a] Shafi Goldwasser, Yael Tauman Kalai, and Guy Rothblum. One-time programs. In *CRYPTO*, Lecture Notes in Computer Science, pages 39–56. Springer, 2008.

[GKR08b] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: interactive proofs for muggles. In *STOC*, pages 113–122, 2008.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. Preliminary version in *STOC'85*.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(1):691–729, 1991. Preliminary version in *FOCS'86*.

[GO96] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious rams. *J. ACM*, 43(3):431–473, 1996.

[HHK+05] Iftach Haitner, Omer Horvitz, Jonathan Katz, Chiu-Yuen Koo, Ruggero Morselli, and Ronen Shaltiel. Reducing complexity assumptions for statistically-hiding commitment. In *Advances in Cryptology – EUROCRYPT 2005*, pages 58–77, 2005. See also preliminary draft of full version, `www.wisdom.weizmann.ac.il/~iftachh/papers/SCfromRegularOWF.pdf`.

[HHRS07] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols – A tight lower bound on the round complexity of statistically-hiding commitments. In *Proceedings of the 47th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 2007.

[HL08] Carmit Hazay and Yehuda Lindell. Constructions of truly practical secure protocols using standardsmartcards. In *ACM Conference on Computer and Communications Security*, pages 491–500, 2008.

[HR07] Iftach Haitner and Omer Reingold. A new interactive hashing theorem. In *IEEE Conference on Computational Complexity*, pages 319–332, 2007. See also preliminary draft of full version at the first author's home page.

[HRVW09] Iftach Haitner, Omer Reingold, Salil P. Vadhan, and Hoeteck Wee. Inaccessible entropy. In *STOC*, pages 611–620, 2009.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591. Springer, 2008.

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 44–61. ACM Press, 1989.

[Kat07] Jonathan Katz. Universally composable multi-party computation using tamper-proof hardware. In *EUROCRYPT*, Lecture Notes in Computer Science, pages 115–128. Springer, 2007.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 20–31, 1988.

[KLR06] Kushilevitz, Lindell, and Rabin. Information-theoretically secure protocols and security under composition. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2006.

[Kol10] Vladimir Kolesnikov. Truly efficient string oblivious transfer using resettable tamper-proof tokens. In *TCC*, pages 327–342, 2010.

[KPT97] Joe Kilian, Erez Petrank, and Gábor Tardos. Probabilistically checkable proofs with zero knowledge. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 1997.

[KR08] Yael Tauman Kalai and Ran Raz. Interactive PCP. In *ICALP (2)*, pages 536–547, 2008.

[LS95] Dror Lapidot and Adi Shamir. A one-round, two-prover, zero-knowledge protocol for np. *Combinatorica*, 15(2):204–214, 1995.

[Mau92] Ueli M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *J. Cryptology*, 5(1):53–66, 1992.

[MS08] Tal Moran and Gil Segev. David and Goliath commitments: UC computation for asymmetric parties using tamper-proof hardware. In *EUROCRYPT*, pages 527–544, 2008.

[NOVY98] Moni Naor, Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *Journal of Cryptology*, 11(2):87–108, 1998. Preliminary version in *CRYPTO'92*.

[OVY93a] Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Fair games against an all-powerful adversary. *AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 155–169, 1993. Preliminary version in *SEQUENCES'91*.

[OVY93b] Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Interactive hashing simplifies zero-knowledge protocol design. In *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 267–273. Springer, 1993.

[OW93] Rafail Ostrovsky and Avi Wigderson. One-way fuctions are essential for non-trivial zero-knowledge. In *ISTCS*, pages 3–17, 1993.

[Rab81] M. O. Rabin. How to exchange secrets by oblivious transfer. TR-81, Harvard, 1981.

[RB89] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *STOC*, pages 73–85, 1989.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de