Impossibility Results for Leakage-Resilient Zero Knowledge and Multi-Party Computation

Rafail Ostrovsky
UCLA
USA
rafail@cs.ucla.edu

Giuseppe Persiano
University of Salerno
ITALY
giuper@gmail.com

Ivan Visconti
University of Salerno
ITALY
visconti@unisa.it

Abstract

In [AGP14] Ananth et al. showed that continual leakage-resilient non-transferable interactive proofs exist when a leak-free input-encoding phase is allowed and a common reference string is available. They left open the problem of removing the need of a common reference string.

In [BGJK12] Boyle et al. showed that for some interesting functionalities continual leakage-resilient secure computation is possible when leak-free interactive preprocessing and input-encoding phases are allowed. They left open the problem of removing the interactive pre-processing.

In this work we study the above questions. Our main contribution shows that leakage-resilient black-box zero-knowledge is impossible when relying on a leak-free input-encoding phase only (i.e., without CRS/preprocessing). Additionally, we also show that leakage-resilient multi-party computation for all functionalities is impossible (regardless of the number of players assuming just one corrupted player) when relying only on a leak-free input-encoding phase (i.e., without CRS/preprocessing).

Our results are achieved by means of a new technique to prove lower bounds for leakage-resilient security. We use leakage queries to run an execution of a communication-efficient insecure (i.e., non-simulatable) protocol in the head of the adversary. Moreover our work shows an interesting connection between leakage resilience and security against reset attacks.

Keywords: leakage resilience, zero knowledge, MPC, resettability.

1 Introduction

The intriguing notion of a zero-knowledge proof introduced by Goldwasser, Micali and Rackoff [GMR85] has been for almost three decades a source of fascinating open questions in Cryptography and Complexity Theory. Indeed, motivated by new real-world attacks, the notion has been studied in different flavors (e.g., non-interactive zero knowledge [BSMP91], non-malleable zero knowledge [DDN91], concurrent zero knowledge [DNS98], resettable zero knowledge [CGGM00]) and each of them required extensive research to figure out the proper definitions and their (in)feasibility. Moreover all such real-world attacks have been considered also for the natural generalization of the concept of zero knowledge: secure computation [GMW87].

Leakage attacks. Leakage resilience deals with modeling real-word attacks where the adversary manages through some physical observations to obtain side-channel information on the state (e.g., private input, memory content, randomness) of the honest player. Starting with the works of [ISW03, MR04, DP08] leakage resilience has been a main-stream research topic in Cryptography, and the gap between theory and practice has been significantly reduced [DDF14].

Recently the notions of leakage-resilient zero knowledge (LRZK) and secure multi-party computation (LRMPC) have been considered. Despite the above intensive research on leakage resilience, LRZK and LRMPC are still rich of interesting open problems.

1.1 Previous Work and Open Problems

Leakage resilience vs tolerance. The first definition for leakage-resilient zero knowledge (LRZK, in short) was given by Garg et al. in [GJS11]. In their definition, the simulator is allowed to make leakage queries in the ideal world. This formalization of security has been extensively studied by Bitansky et al. in [BCH12] for the case of universally composable secure computation [Can01]. The main motivation behind such definitions is the fact that a private input can not be protected in full from leakage attacks. Therefore the best one can hope for is to tolerate the leakage of private information due to leakage queries making sure that the protocol does not leak even more. Similar definitions have been used in [BGK11, BGJ$^+$13, BGK14].

In [GJS11], constructions for LRZK in the standard model and for non-interactive LRZK in the common reference string (CRS) model were given. Bitansky et al. [BCH12] showed a composition theorem and gave constructions of various UC-secure computations (including zero knowledge) in the CRS model. Very recently Pandey showed in [Pan14] a constant-round construction for LRZK under the definition of [GJS11], and Bitansky et al. [BDL14] obtained UC-secure continual leakage tolerance using an input-independent leak-free preprocessing phase.

Nowadays, the commonly accepted term for the security used in [GJS11, BCH12, Pan14] is referred to as *leakage tolerance* as it does not prevent a leakage attack but only guarantees that a protocol does not leak more than what can be obtained through leakage queries.

Open problems: leakage resilience with leak-free encoding. The above argument that a private input can not be protected in full from a leakage query is quite extreme and does not necessarily fit all real-world scenarios. Indeed, it is commonly expected that an adversary attacks the honest player during the execution of the protocol, while they are connected through some communication channel. Nothing prevents a honest player to receive an input in a preliminary phase, before having ever had any interaction with the adversary. Once this input is received, the honest player can encode it in order to make it somewhat intelligible from leakage queries but still valid for the execution of a protocol. This encoding phase can be considered leak-free since, as stressed before, the honest player has never been in touch with the adversary[1]. Later on, when the interaction with the adversary starts, leakage queries will be possible but they will affect the current state of the honest player that contains an encoding of the input. The need of a leak-free phase to protect a secret from leakage queries was considered also in [FRR$^+$10, GR10, GR12].

The above realistic scenario circumvents the argument that leakage tolerance is the best one can hope for, and opens the following challenging open questions:

Open Question 1: *"Assuming players can encode their inputs during a leak-free phase, is it possible to construct LRZK argument/proof systems?"*

Open Question 2: *"Assuming players can encode their inputs during a leak-free phase, is it possible to construct protocols for leakage resilient Multi-Party Computation (LRMPC)?"*

Leakage resilience assuming the existence of a CRS. Very recently, Ananth et al. [AGP14], showed that in the CRS model it is possible to have an interactive argument system that remains non-transferable even in presence of continual leakage attacks. More precisely, in their model a prover in a leak-free environment can encode a witness and, later on, the prover can run a protocol with a verifier using the encoded witness. During the execution of the protocol, the adversarial verifier is allowed to launch leakage queries. At the end of the execution of the protocol the prover can refresh (in a leak-free environment) its encoded witness and then it can play again with the verifier (under leakage attacks). The problem solved in [AGP14] in the positive is to show that there exists an encoding/refreshing mechanism and a protocol for non-transferable arguments against such continual leakage attacks. They left explicitly open the following open problem (see page 167 of [AGP14]): is it possible to obtain non-transferable arguments/proofs that remain secure against continual leakage attacks without relying on a CRS? This problem has similarities with Open Problem 1. Indeed, zero knowledge (without a CRS) implies non-transferability and therefore solving Open Problem 1 in the positive would solve the problem opened by [AGP14] too.

Leakage resilience assuming leak-free preprocessing. In [BGJK12], Boyle et al. proposed a model for leakage-resilient secure computation based on the following three phases:

1. a leak-free interactive preprocessing to be run only once, obliviously w.r.t. inputs and functions;
2. a leak-free stand-alone input-encoding phase to be run when a new input arrives (and of course after the interactive preprocessing), obliviously w.r.t. functions to be computed later;
3. an on-line phase where parties, on input the states generated during the last executions of the input-encoding phases, and on input a function f, run a protocol that aims at securely computing the output of f.

In the model of [BGJK12] leakage attacks are not possible during the first two phases but are possible in any other moment, including the 3rd phase and in between phases.

[BGJK12] showed a) the impossibility of leakage-resilient 2-party computation and, more in general, of n-party LRMPC when $n-1$ players are corrupted; b) the feasibility of leakage-resilient MPC when the number of players is polynomial and a constant fraction of them is honest.

The positive result works for an even stronger notion of leakage resilience referred to as "continual leakage" [DHLW10, BKKV10, DLWW11, BSW11, DF12, BSW13, ADVW13] that means

[1]Moreover such a phase can be run on a different device disconnected from the network, running an operating system installed on some read-only storage.

that the same input can be re-used through unbounded multiple executions of the protocol each allowing for a bounded leakage, as long as the state can be refreshed after each execution. Leakage queries are allowed also during the refreshing.

Boyle et al. explicitly leave open (see paragraph "LR-MPC with Non-Interactive Preprocessing" on page 1240 of [BGJK12]) the problem of achieving their results without the preprocessing (i.e., Open Question 2) and implicitly left open the case of zero-knowledge arguments/proofs. (i.e., Open Question 1) since when considering the ZK functionality only, the function is known in advance and therefore their impossibility for the two-party case does not directly hold.

We notice that the result of [AGP14] does not yield a continual leakage-resilient non-transferable proof system for the model of [BGJK12]. Indeed, while the preprocessing of [BGJK12] can be used to establish the CRS needed by [AGP14], the refresh of the state of [AGP14] requires a leak-free phase that is not available in the model of [BGJK12]. We finally stress that the construction of [AGP14] is not proved to be LRZK.

However the interesting open question in the model of [BGJK12] consists in achieving continual LRZK *without* an interactive preprocessing. Indeed, if an interactive preprocessing is allowed, continual LRZK can be trivially achieved as follows. The preprocessing can be used to run a secure 2-party computation for generating a shared random string. The input-encoding phase can replace the witness with a non-interactive zero-knowledge proof of knowledge (NIZKPK). The on-line phase can be implemented by simply sending the previously computed NIZKPK. This trivial solution would allow the leakage of the entire state, therefore guaranteeing continual leakage (i.e., no refresh is needed), and would even guarantee non-transferability since the CRSs established among pairs of players would be independent from each other.

1.2 Our Results

In this paper we study the above open questions and show the following results.

1.2.1 Black-Box LRZK without CRS/Preprocessing

As a main result, we show that black-box LRZK is impossible for non-trivial languages if we only rely on a leak-free input-encoding phase (i.e., without CRS/preprocessing). More in details, with respect to the works of [BGJK12, AGP14], our results shows that, by removing the CRS/preprocessing, not only non-transferable continual black-box LRZK is impossible, but even ignoring non-transferability and continual leakage, the simple notion of 1-time black-box LRZK is impossible. We design an adversarial verifier $\mathsf{V}^\star$ that uses leakage queries to obtain a very small amount of data compared to the state of the prover and whose view cannot be simulated in a black-box manner. The impossibility holds even knowing already at the input-encoding phase which protocol will be played later.

Overview of our techniques. We prove the above impossibility result by means of a new technique: the adversary will attack the honest player without running the actual protocol at all! Indeed, the adversary will only run an execution of another (insecure) protocol in its head, using leakage queries to get messages from the other player for the "virtual" execution of the (insecure) protocol.

More in details, assuming by contradiction the existence of a protocol (P, V) for a language $L \notin \mathsf{BPP}$, we show an adversary $\mathsf{V}^\star$ that first runs a leakage query to obtain a collision-resistant (CR) hash $\tilde{w}$ of the state $\hat{w}$ of the prover. Then it takes a communication-efficient (insecure) protocol $\Pi = (\Pi.\mathsf{P}, \Pi.\mathsf{V})$ and, through leakage queries, $\mathsf{V}^\star$ runs in its head an execution of Π

playing as a honest verifier $\Pi.\mathsf{V}$, while the prover P will have to play as $\Pi.\mathsf{P}$ proving that the hash is a good one: namely, it corresponds to a state that would convince a honest verifier V on the membership of the instance in L.

Notice that in the real-world execution P would convince $\mathsf{V}^\star$ during the "virtual" execution of Π since P runs as input an encoded witness that by the completeness of (P, V) convinces V.

Therefore a black-box simulator will have to do the same without having the encoding of a witness but just relying on rewinding capabilities. To show our impossibility we make the capabilities of the simulator useless. This is done by connecting leakage resilience with resettability. Indeed we choose Π not only to be communication efficient on $\Pi.\mathsf{P}$'s side (this helps so that the sizes of the outputs of leakage queries will correspond to a small portion of the state of P), but also to be a resettable argument of knowledge (and therefore resettably sound). Such arguments of knowledge admit an extractor $\Pi.\mathsf{Ext}$ that works even against a resetting prover $\Pi.\mathsf{P}^\star$ (i.e., such an adversary in our impossibility will be the simulator Sim of (P, V)).

The existence of a family of CR hash functions gives not only the CR hash function required by the first leakage query but also the communication-efficient resettable argument of knowledge for $\mathbb{NP}$. Indeed we can use Barak's public-coin universal argument [Bar04] that enjoys a *weak* argument of knowledge property when used for languages in NEXP. Instead when used for $\mathbb{NP}$ languages, Barak's construction is a *regular* argument of knowledge with a black-box extractor. We can finally make it extractable also in presence of a resetting prover by using the transformation of Barak et al. [BGGL01] that only requires the existence of one-way functions.

Summing up, we will show that the existence of a black-box simulator for (P, V) implies either that the language is in BPP, or that (P, V) is not sound or that the family of hash functions is not collision resistant.

1.2.2 Impossibility of Leakage-Resilient MPC for Any Functionality

Additionally, we address Open Question 2 by showing that for many functionalities LRMPC with a leak-free input-encoding phase (and without an interactive preprocessing phase) is impossible. This impossibility holds regardless of the number of players involved in the computation and only assumes that one player is corrupted. It applies to functionalities that when executed multiple times keeping unchanged the input x_i of a honest player P_i, produce outputs delivered to the dishonest players that reveal more information on x_i than what a single output would reveal. We also require outputs to be short.

Our impossibility is actually even stronger since it holds also in case the functionality and the corresponding protocol to be run later are already known during the input-encoding phase.

For simplicity, we will discuss a direct example of such a class of functionalities: a variation of Yao's Millionaires' Problem, where n players send their inputs to the functionality that will then send as output a bit b specifying whether player P_1 is the richest one.

High-level overview. The adversary will focus on attacking player P_1 that has an input to protect. The adversary can play in its head by means of a single leakage query the entire protocol selecting inputs and randomnesses for all other players, and obtaining as output of the leakage query the output of the function (i.e., the bit b). This "virtual" execution can be repeated multiple times, therefore extracting more information on the input of the player. Indeed playing multiple times and changing the inputs of the other players while the input of P_1 remains the same, it is possible to restrict the possible input of P_1 to a much smaller range of values than what can be inferred by a single execution.

The above attack will be clearly impossible to simulate since it would require the execution of multiple queries in the ideal world, but the simulator by definition can make only one query.

When running the protocol through leakage queries, we are of course assuming that authenticated channels do not need to be simulated by the adversary[2] since their management is transparent to the state of the players running the leakage-resilient protocol. This is already assumed in previous work like [BGJK12] since otherwise leakage-resilient authenticated channels would have been required, while instead [BGJK12] only requires an authenticated broadcast channel (see Section 3 of [BGJK12]).

Since this second result is technically less interesting than our main result, in the paper we will concentrate on the main result and only sketch this additional simpler result.

2 Definitions

We will denote by "$\alpha \circ \beta$" the string resulting from appending β to α, and by $[k]$ the set $\{1, \ldots, k\}$. A polynomial-time relation R is a relation for which it is possible to verify in time polynomial in $|x|$ whether $R(x, w) = 1$. We will consider $\mathbb{NP}$-languages L and denote by R_L the corresponding polynomial-time relation such that $x \in L$ if and only if there exists w such that $R_L(x, w) = 1$. We will call such a w a *valid witness for* $x \in L$ and denote by $W_L(x)$ the set of valid witnesses for $x \in L$. We will slightly abuse notation and, whenever L is clear from the context, we will simply write $W(x)$ instead of $W_L(x)$. A *negligible* function $\nu(k)$ is a function such that for any constant $c < 0$ and for all sufficiently large k, $\nu(k) < k^c$.

We will now give all definitions required for the main result of our work, the impossibility of black-box LRZK. Since we will only sketch the additional result on LRMPC, we defer the reader to [BGJK12] for the additional definitions.

2.1 Interactive Proof Systems

An *interactive proof system* [GMR85] for a language L is a pair of interactive Turing machines (P, V), satisfying the requirements of *completeness* and *soundness*. Informally, completeness requires that for any $x \in L$, at the end of the interaction between P and V, where P has on input a valid witness for $x \in L$, V rejects with negligible probability. Soundness requires that for any $x \notin L$, for any computationally unbounded $\mathsf{P}^\star$, at the end of the interaction between $\mathsf{P}^\star$ and V, V accepts with negligible probability. When $\mathsf{P}^\star$ is only probabilistic polynomial-time, then we have an argument system. We denote by $\langle \mathsf{P}, \mathsf{V} \rangle(x)$ the output of the verifier V when interacting on common input x with prover P. Also, sometimes we will use the notation $\langle \mathsf{P}(w), \mathsf{V} \rangle(x)$ to stress that prover P receives as additional input witness w for $x \in L$. We will write $\langle \mathsf{P}(w; r_P), \mathsf{V}(r_V) \rangle(x)$ to make explicit the randomness used by P and V. We will also write $\mathsf{V}^\star(z)$ to denote an adversarial verifier $\mathsf{V}^\star$ that runs on input an auxiliary string z.

Definition 2.1 *[GMR85] A pair of interactive Turing machines* (P, V) *is an* interactive proof system *for the language L, if* V *is probabilistic polynomial-time and*

1. *Completeness: There exists a negligible function $\nu(\cdot)$ such that for every $x \in L$ and for every $w \in W(x)$* $\mathrm{Prob}[\ \langle \mathsf{P}(w), \mathsf{V} \rangle(x) = 1\ ] \geq 1 - \nu(|x|)$.

2. *Soundness: For every $x \notin L$ and for every interactive Turing machines* $\mathsf{P}^\star$ *there exists a negligible function $\nu(\cdot)$ such that* $\mathrm{Prob}[\ \langle \mathsf{P}^\star, \mathsf{V} \rangle(x) = 1\ ] \leq \nu(|x|)$.

[2]More in details, we are assuming that the encoded state of the player does not include any information useful to check if a message supposed to be from a player P_j is genuine.

If the soundness condition holds only with respect to probabilistic polynomial-time interactive Turing machines $\mathsf{P}^\star$ *then* (P,V) *is called an* argument*.*

We now define the notions of reset attack and of resetting prover.

Definition 2.2 *[BGGL01] A reset attack of a prover* $\mathsf{P}^\star$ *on* V *is defined by the following two-step random process, indexed by a security parameter* k*.*

1. *Uniformly select and fix* $t = \mathsf{poly}(k)$ *random tapes, denoted by* $r_1,\ldots,r_t$*, for* V*, resulting in deterministic strategies* $\mathsf{V}^{(i)}(x) = \mathsf{V}_{x,r_i}$ *defined by* $\mathsf{V}_{x,r_i}(\alpha) = \mathsf{V}(x,r_i,\alpha)$*, where* $x \in \{0,1\}^k$ *and* $i \in 1,\ldots,t$*. Each* $\mathsf{V}^{(i)}(x)$ *is called an* `incarnation` *of* V*.*
2. *On input* 1^k*, machine* $\mathsf{P}^\star$ *is allowed to initiate* $\mathsf{poly}(k)$*-many interactions with* V*. The activity of* $\mathsf{P}^\star$ *proceeds in rounds. In each round* P *chooses* $x \in \{0,1\}^k$ *and* $i \in 1,\ldots,t$*, thus defining* $\mathsf{V}^{(i)}(x)$*, and conducts a complete session (a session is complete if is either terminated or aborted) with it.*

We call resetting prover *a prover that launches a reset attack.*

We now define proofs/arguments of knowledge, in particular considering the case of a prover launching a reset attack.

Definition 2.3 *[BG93] Let* R *be a binary relation and* $\epsilon : \{0,1\}^\star \rightarrow [0,1]$*. We say that a probabilistic, polynomial-time interactive machine* V *is a knowledge verifier for the relation* R *with knowledge error* ϵ *if the following two conditions hold:*

Non-triviality: *There exists a probabilistic polynomial-time interactive machine* P *such that for every* $(x,w) \in R$*, with overwhelming probability an interaction of* V *with* P *on common input* x*, where* P *has auxiliary input* w*, is accepting.*

Validity (or knowledge soundness) with negligible error ϵ**:** *for every probabilistic polynomial-time machine* $\mathsf{P}^\star$*, there exists an expected polynomial-time machine* Ext*, such that and for every* $x, aux, r \in \{0,1\}^\star$*,* Ext *satisfies the following condition: Denote by* $p(x,aux,r)$ *the probability (over the random tape of* V*) that* V *accepts upon input* x*, when interacting with the prover* $\mathsf{P}^\star$ *who has input* x*, auxiliary-input* aux *and random-tape* r*. Then, machine* Ext*, upon input* (x,aux,r)*, outputs a solution* $w \in R(x)$ *with probability at least* $p(x,aux,r) - \epsilon(|x|)$*.*

A pair (P,V) *such that* V *is a knowledge verifier with* negligible *knowledge error for a relation* R *and* P *is a machine satisfying the non-triviality condition (with respect to* V *and* R*) is called an argument of knowledge for the relation* R*. If the validity condition holds with respect to any (not necessarily polynomial- time) machine* $\mathsf{P}^\star$*, then* (P,V) *is called a proof of knowledge for* R*. If the validity condition holds with respect to a polynomial-time machine* $\mathsf{P}^\star$ *launching a reset attack, then* (P,V) *is called a resettable argument of knowledge for* R*.*

In the above definition if there exists a universal extractor (i.e., the same extractor works with all possible provers) Ext then the interactive argument/proof system is a *black-box* (resettable) argument/proof of knowledge.

The input-encoding phase. Following previous work we will assume that the prover receives the input and encodes it in a leak-free environment. This is unavoidable since otherwise a leakage query can cask for some bits of the witness and therefore zero knowledge would be trivially impossible to achieve. After this leak-free phase that we call input-encoding phase, the prover has a state consisting only of the encoded witness and is ready to start the actual leakage-resilient protocol.

Leakage-resilient protocol [Pan14]. As in previous work, we assume that random coins are available only in the specific step in which they are needed. More in details, the prover P at each round of the protocol obtains fresh randomness r for the computations related to that round. However, unlike in previous work, we do not require the prover to updates its state by appending r to it. We allow the prover to erase randomness and change its state during the protocol execution. This makes our impossibility results even stronger.

The adversarial verifier performs a leakage query by specifying a polynomial-sized circuit C that takes as input the current state of the prover. The verifier gets immediately the output of C and can adaptively decide how to continue. An attack of the verifier that includes leakage queries is called a leakage attack.

Definition 2.4 *We say that an interactive argument/proof system* (P, V) *for a language* $L \in \mathbb{NP}$ *with a witness relation* R*, is leakage-resilient zero-knowledge if for every probabilistic polynomial-time machine* V^* *launching a leakage attack on* P*, there exists a probabilistic polynomial-time machine* Sim *such that for every* $x \in L$*, every* w *such that* $R(x, w) = 1$*, and every* $z \in \{0,1\}^*$ *distributions* $\langle \mathsf{P}(w), \mathsf{V}^\star(z)\rangle(x)$ *and* $\mathsf{Sim}(x, z)$ *are computationally indistinguishable.*

The definition of standard zero-knowledge is obtained by enforcing that no leakage query is allowed to any machine and removing the input-encoding phase.

In the above definition if there exists a universal simulator (i.e., the same simulator works with all possible verifiers) Sim then the interactive argument/proof system is leakage-resilient *black-box* zero knowledge.

3 Impossibility of Leakage-Resilient ZK

In this section, we prove that LRZK argument systems exist only for BPP languages.

Tools. In our proof we assume the existence of a communication-efficient argument system $\Pi = (\Pi.\mathsf{P}, \Pi.\mathsf{V})$ for a specific auxiliary $\mathbb{NP}$ language (to be defined later). Moreover we require such an argument system to be a resettable argument of knowledge. Specifically, we require that on common input x, $\Pi.\mathsf{P}$ sends $O(|x|^\epsilon)$ bits to $\Pi.\mathsf{V}$ for an arbitrarily chosen constant $\epsilon > 0$. We denote, with a slight abuse of notation, by $\Pi.\mathsf{P}$ the prover's next message function; that is, $\Pi.\mathsf{P}$ on input x, randomness $r_1, \ldots, r_{i-1}$ used in the previous $i-1$ rounds, fresh randomness r_i and verifier messages $v_1, \ldots, v_i$ received so far, outputs $\mathtt{msg}_i$, the prover's i-th message. Similarly, we denote the verifier's next message function by $\Pi.\mathsf{V}$. Finally, we denote by $\Pi.\mathsf{Ext}$ the extractor that in expected polynomial time outputs a witness for $x \in L$ whenever a polynomial-time prover can make $\Pi.\mathsf{V}$ accept $x \in L$ with non-negligible probability.

Such a resettable argument of knowledge Π exists based on the existence of a family of collision-resistant hash functions. It can be obtained by starting with the the public-coin universal argument of [Bar04] that for $\mathbb{NP}$ languages is also an argument of knowledge. Then by applying the transformation of [BGGL01] that requires one-way functions, we have that the resulting protocol is still communication efficient, and moreover is a resettable argument of knowledge.

Theorem 3.1 *Assume the existence of a family of collision-resistant hash functions. If an* $\mathbb{NP}$*-language* L *admits a leakage-resilient black-box zero-knowledge argument system* $\Pi_{LRZK} = (\mathsf{P}, \mathsf{V})$ *then* $L \in \mathsf{BPP}$.

Proof. For sake of contradiction, we assume that language L admits a leakage-resilient zero-knowledge argument system (P, V) with black-box simulator Sim. We now describe an adversarial verifier $\mathsf{V}^\star = \mathsf{V}^\star_{x,s,h,t}$, parameterized by input x, strings s and t, and function h from a family of collision-resistant hash functions. In the description of $\mathsf{V}^\star$, we let $\{F_s\}$ be a pseudorandom family of functions. Moreover we assume wlog that the number of rounds of Π is 2ℓ (i.e., ℓ messages played by the verifier and ℓ messages played by the prover) where $\ell > 1$ and that the verifier speaks first.

Our proof makes use of the auxiliary language Λ consisting of the tuples $\tau = (h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}})$ for which there exists $\hat{w}$ such that $h(\hat{w}) = \tilde{w}$ and $\langle \mathsf{P}(\hat{w}; \mathtt{rand}^{\mathsf{P}}), \mathsf{V}(\mathtt{rand}^{\mathsf{V}})\rangle(x) = 1$. Clearly, $\Lambda \in \mathbb{NP}$.

1. At the start of the interaction between P and $\mathsf{V}^\star$ on an n-bit input x with $n = \mathsf{poly}(k)$, the state of P consists solely of the encoding $\hat{w}$ of the witness w for $x \in L$, where $|\hat{w}| = \mathsf{poly}(n)$.

2. $\mathsf{V}^\star$ issues leakage query Q_0 by specifying function h; as a reply, $\mathsf{V}^\star$ receives $\tilde{w} = h(\hat{w})$, a hash of the encoding of the witness used by P.

3. $\mathsf{V}^\star$ then selects randomness

$$\mathtt{rand} = (\mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}, \mathtt{rand}_1^{\Pi.\mathsf{P}}, \ldots, \mathtt{rand}_\ell^{\Pi.\mathsf{P}}, \mathtt{rand}_1^{\Pi.\mathsf{V}}, \ldots, \mathtt{rand}_\ell^{\Pi.\mathsf{V}}, \mathtt{rand}_{\ell+1}^{\Pi.\mathsf{V}})$$

   by setting $\mathtt{rand} = F_s(\tilde{w} \circ x)$.

4. $\mathsf{V}^\star$ performs, by means of leakage queries, an execution of the protocol Π on common input $(h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}})$.

   Specifically, for round $i = 1, \ldots, \ell$, $\mathsf{V}^\star$ computes

$$v_i = \Pi.\mathsf{V}\left((h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}), \{\mathtt{msg}_j\}_{0<j<i}, \{\mathtt{rand}_j^{\Pi.\mathsf{V}}\}_{0<j\le i}\right)$$

   and issues leakage query Q_i for the prover's next message function

$$\Pi.\mathsf{P}\left((h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}),\ \cdot\ , \{\mathtt{rand}_j^{\Pi.\mathsf{P}}\}_{0<j\le i}, \{v_j\}_{0<j\le i}\right)$$

   that is to be applied to the state $\hat{w}$ of prover P. In other words, the query computes the prover's i-th message $\mathtt{msg}_i$ of an interaction of protocol Π in which prover $\Pi.\mathsf{P}$ (running on randomness $\mathtt{rand}_1^{\Pi.\mathsf{P}}, \ldots, \mathtt{rand}_\ell^{\Pi.\mathsf{P}}$) tries to convince verifier $\Pi.\mathsf{V}$ (running on randomness $\mathtt{rand}_1^{\Pi.\mathsf{V}}, \ldots, \mathtt{rand}_\ell^{\Pi.\mathsf{V}}, \mathtt{rand}_{\ell+1}^{\Pi.\mathsf{V}}$) that $(h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}) \in \Lambda$.

   After receiving prover $\Pi.\mathsf{P}$'s last message, $\mathsf{V}^\star$ computes $\Pi.\mathsf{V}$'s output in this interaction:

$$b = \Pi.\mathsf{V}((h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}), \mathtt{msg}_1, \ldots, \mathtt{msg}_\ell, \mathtt{rand}_1^{\Pi.\mathsf{V}}, \ldots, \mathtt{rand}_{\ell+1}^{\Pi.\mathsf{V}}).$$

5. If $b = 1$ then $\mathsf{V}^\star$ outputs t; otherwise, $\mathsf{V}^\star$ outputs $\perp$.

This concludes the description of $\mathsf{V}^\star$.

Counting the number of bits leaked. The total number of bits leaked is equal to the output of the first leakage query (i.e., the length in bits of a range element of the collision-resistant hash function) $|\tilde{w}| = k$ and the number of bits sent by the prover in Π which, for inputs of length n, is $O(n^\epsilon)$ for an arbitrarily constant $\epsilon > 0$. Being $n = \mathsf{poly}(k)$, we have that the amount of leakage is extremely small compared to the size of the state of the prover.

Sim **can get t only by succeeding in Π, therefore properly answering to leakage queries.** We continue by observing that the output of the real game (i.e., when P and $\mathsf{V}^\star_{x,s,h,t}$ interact) is t. Therefore, Sim must output t when interacting with $\mathsf{V}^\star_{x,s,h,t}$ with overwhelming probability. Since Sim is a black-box simulator, and since all messages of $\mathsf{V}^\star_{x,s,h,t}$ except for the last one, are independent of t, the only way Sim can obtain t from $\mathsf{V}^\star_{x,s,h,t}$ is by replying with a value $\tilde{w}$ to the first leakage query and by replying to queries $Q_1, \ldots, Q_\ell$ so to define a transcript $\mathsf{Conv} = (v_1, \mathtt{msg}_1, \ldots, v_\ell, \mathtt{msg}_\ell)$ that for common input $(h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}})$ produces $1 = \Pi.\mathsf{V}((h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}), \mathtt{msg}_1, \ldots, \mathtt{msg}_\ell, \mathtt{rand}_1^{\Pi.\mathsf{V}}, \ldots, \mathtt{rand}_{\ell+1}^{\Pi.\mathsf{V}})$.

By the security of the pseudorandom function, we can consider the same experiment except having that $\mathtt{rand} = \mathcal{R}(\tilde{w} \circ x)$ (computed by $\mathsf{V}^\star$ in step 3 of its description) where $\mathcal{R}$ is a truly random function (i.e., each time $\tilde{w} \circ x$ is new, $\mathtt{rand}$ is computed by sampling fresh randomness).

We denote by $\mathsf{Sim}_R^{\mathsf{V}^\star}$ the simulation in such a modified game. We now show that when considering $\mathsf{Sim}_R^{\mathsf{V}^\star}$, still t is given in output with overwhelming probability.

Lemma 3.2 *The output of* $\mathsf{Sim}_R^{\mathsf{V}^\star}$ *is computationally indistinguishable from the output of* $\mathsf{Sim}^{\mathsf{V}^\star}$.

Proof. Suppose that the claim does not hold. We can break the security of the pseudorandom functions (PRF) by means of the following adversary $\mathcal{A}_{PRF}$. $\mathcal{A}_{PRF}$ runs precisely as $\mathsf{Sim}_R^{\mathsf{V}^\star}$ and $\mathsf{Sim}^{\mathsf{V}^\star}$ except when $\mathtt{rand}$ is computed, that is the only step where $\mathsf{Sim}_R^{\mathsf{V}^\star}$ and $\mathsf{Sim}^{\mathsf{V}^\star}$ differ. $\mathcal{A}_{PRF}$ computes $\mathtt{rand}$ by querying the PRF challenger with $\tilde{w} \circ x$. We have that if the challenger uses a truly random function, then the experiment corresponds to $\mathsf{Sim}_R^{\mathsf{V}^\star}$, while if the challenger uses a PRF, then the experiment corresponds to $\mathsf{Sim}^{\mathsf{V}^\star}$. We can therefore use the contradicting assumption to conclude that $\mathcal{A}_{PRF}$ breaks the security of the PRF. □

Lemma 3.3 *If $L \notin \mathsf{BPP}$ then there exists some $x \notin L$ such that $\mathsf{Sim}_R^{\mathsf{V}^\star}(x,z)$ outputs t with probability greater than $2/3$.*

Proof. Suppose by contradiction that for all $x \notin L$, $\mathsf{Sim}_R^{\mathsf{V}^\star}$ outputs t with probability at most $2/3$. Then we contradict the fact that $L \notin \mathsf{BPP}$. Let $\mathsf{time}_{\mathsf{Sim}_R}$ be the expected running time of Sim_R and $\mathsf{time}_{\mathsf{V}^\star}$ be the running time of $\mathsf{V}^\star$. Consider the following machine M: on input x, M runs Sim_R and $\mathsf{V}^\star_{x,s,h,t}$ up to $8\mathsf{time}_{\mathsf{Sim}} \cdot \mathsf{time}_{\mathsf{V}^\star}$ steps, for randomly chosen strings s and t, and a randomly chosen hash function h from a family $\mathcal{H}$ of collision-resistant hash functions. This is repeated k times. If in more than $3/4$ of the k iterations Sim_R outputs a value different from t then M rejects x; otherwise, M accepts x. Notice that, for $x \in L$, by Lemma 3.2 and by the zero-knowledge property Sim_R would output t in all k repetitions with overwhelming probability. Notice however that for each iteration M is guaranteed to run Sim_R up to $8\mathsf{time}_{\mathsf{Sim}_R}$ steps. We know by the expected running time of Sim_R that the probability that a run of Sim_R takes more than $8\mathsf{time}_{\mathsf{Sim}_R}$ steps is at most $1/8$. Given that there are k repetitions, by applying Chernoff bound the probability that M rejects x is therefore negligible in k.

On the other hand, when $x \notin L$, by the above contradiction we know that for all $x \notin L$, $\mathsf{Sim}_R^{\mathsf{V}^\star}$ outputs t with probability at most $2/3$. Therefore again applying Chernoff bound, M's output is correct (i.e., M rejects x) with overwhelming probability.

We therefore conclude that the claim holds. □

As usual we set $\mathsf{V}^\star = \mathsf{V}^\star_{x,s,h,t}$. Let $x \notin L$ be a special statement such that $\mathsf{Sim}_R^{\mathsf{V}^\star}(x,z)$ outputs t with non-negligible probability (such an x exists since we are assuming that $L \notin \mathsf{BPP}$). This means that Sim_R feeds $\mathsf{V}^\star$ with a transcript of messages that with non-negligible probability produces t as output.

Let $\mathsf{time}_{\mathsf{Sim}_R}$ be the expected running time of Sim_R. Consider the strict polynomial time machines Sim_{pR} that consists of running the first $3\mathsf{time}_{\mathsf{Sim}_R}$ steps of Sim_R.

Lemma 3.4 *If $L \notin \mathsf{BPP}$ then there exists some $x \notin L$ such that $\mathsf{Sim}_{pR}^{\mathsf{V}^\star}(x,z)$ outputs t with probability greater than $1/3$.*

Proof. Suppose by contradiction that the claim does not hold. Then a run of $\mathsf{Sim}_R^{\mathsf{V}^\star}$ that takes at most $3\mathsf{time}_{\mathsf{Sim}_R}$ steps has probability $\leq 1/3$ of giving in output t. The remaining runs of $\mathsf{Sim}_R^{\mathsf{V}^\star}$ happen with probability less than $1/3$, otherwise the expected running time of $\mathsf{Sim}_R^{\mathsf{V}^\star}$ would be greater than $\mathsf{time}_{\mathsf{Sim}_R}$. If all such remaining runs give in output t we have that $\mathsf{Sim}_R^{\mathsf{V}^\star}$ gives in output t with probability $\leq 1/3 + 1/3 = 2/3$, which contradicts Lemma 3.3. Therefore the claim holds. □

For notation purposes, we say that a query of Sim_{pR} to $\mathsf{V}^\star$ belongs to the i-th session if it is a tuple $(h, \tilde{w}, \ldots)$ where $\tilde{w}$ is the i-th different value played by Sim_{pR} as first message of $\Pi.\mathsf{P}$ in a query to $\mathsf{V}^\star$. Let $\mathsf{time}_{\mathsf{Sim}_{pR}}$ be the strict polynomial corresponding to the running time of Sim_{pR}.

Lemma 3.5 *There exist $x \notin L$ and $i \in [\mathsf{time}_{\mathsf{Sim}_{pR}}]$ such that $\mathsf{Sim}_{pR}^{\mathsf{V}^\star}$ obtains t as answer to a query of the i-th session with non-negligible probability.*

Proof. Assume by contradiction that this is not the case. Consider the special x that exists by Lemma 3.4. Since Sim_{pR} runs in time $\mathsf{time}_{\mathsf{Sim}_{pR}}$, there are at most $\mathsf{time}_{\mathsf{Sim}_{pR}}$ sessions played by Sim_{pR} during the simulation and by the contradicting hypothesis each of them has negligible probability of ending with t as output. This implies that the probability of $\mathsf{Sim}_{pR}^{\mathsf{V}^\star}$ of giving in output t is not non-negligible and this contradicts the fact that by Lemma 3.4 $\mathsf{Sim}_{pR}^{\mathsf{V}^\star}$ must give in output t with non-negligible probability. Therefore the claim holds. □

Consider the augmented simulator $\mathsf{Sim}_{pR}^{i^{\mathsf{V}^\star}}$ that works as $\mathsf{Sim}_{pR}^{\mathsf{V}^\star}$ except that $\mathsf{V}^\star$ in the i-th session will only send h, while all other messages of $\mathsf{V}^\star$ will be asked to an external oracle that plays as honest verifier of Π. Let $\mathsf{time}_{\Pi.\mathsf{Ext}}$ be the expected running time of $\Pi.\mathsf{Ext}$.

Lemma 3.6 *There exist $x \notin L$ and $i \in [\mathsf{time}_{\mathsf{Sim}_{pR}}]$ such that the extractor $\Pi.\mathsf{Ext}$ of Π outputs a witness $\hat{w}$ for $\tau = (h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}) \in \Lambda$ with non-negligible probability and running in expected polynomial time. Moreover* $\text{Prob}[\ \langle \mathsf{P}(\hat{w}), \mathsf{V}\rangle(x) = 1\ ]$ *is non-negligible.*

Proof. The existence of such an $x \notin L$ and i that allow $\Pi.\mathsf{Ext}$ to obtain a witness for $\tau \in \Lambda$ follows directly from Lemma 3.5 and from the facts that $\mathsf{Sim}_{pR}^{i^{\mathsf{V}^\star}}$ only makes oracle calls to the verifier of Π and that the argument of knowledge property of Π holds even against resetting provers.

To prove that $\hat{w}$ is a witness that can convince a honest verifier with non-negligible probability let us assume by contradiction that this is not the case. Therefore we have the following three facts.

1. $\hat{w}$ is such that $\tilde{w} = h(\hat{w})$ (this follows from the argument of knowledge property of Π);
2. $\text{Prob}[\ \langle \mathsf{P}(\hat{w}), \mathsf{V}\rangle(x) = 1\ ]$ is negligible (this follows from the above assumed contradiction);
3. Sim_{pR} can feed to $\mathsf{V}^\star$ for the i-th session an accepting transcript for $\tau = (h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}^{\mathsf{V}}) \in \Lambda$ with non-negligible probability (this follows from Lemma 3.6).

By the above fact 3), we have that with non-negligible probability if we pick a new random string $\mathtt{rand}_{new}^{\mathsf{V}}$ and we play it in the i-th session, we have that Sim_{pR} would still feed with non-negligible probability $\mathsf{V}^\star$ for the i-th session an accepting transcript for $\tau = (h, \tilde{w}, \mathtt{rand}^{\mathsf{P}}, \mathtt{rand}_{new}^{\mathsf{V}}) \in \Lambda$. However we know by fact 2) that the probability that the same witness $\tilde{w}$ works with a verifier running with fresh randomness is negligible. As a consequence, by running $\Pi.\mathsf{Ext}$ with respect to Sim^i_{pR}, under the sole variation that $\mathtt{rand}^{\mathsf{V}}_{new}$ is used instead of $\mathtt{rand}^{\mathsf{V}}$ in the i-th session, we have that $\Pi.\mathsf{Ext}$ would output with non-negligible probability a different witness $\hat{w}'$ such that $\tilde{w} = h(\hat{w}') = h(\hat{w})$. Finally observe that the above still holds if instead of running the expected polynomial-time $\Pi.\mathsf{Ext}$, we run the strict polynomial time machine $\Pi.\mathsf{Ext}_p$ that behaves precisely as $\Pi.\mathsf{Ext}$, but stops after $q(k) \cdot \mathsf{time}_{\Pi.\mathsf{Ext}}$ steps for some polynomial q that depends on the success probability of Sim^i_{pR}.

Summing up, we have obtained in polynomial time $\hat{w} \neq \hat{w}'$ such that $\tilde{w} = h(\hat{w}') = h(\hat{w})$. This of course contradicts the collision resistance property of h, therefore we have reached a contradiction and the claim holds. □

We now show an adversarial prover $\mathsf{P}^\star$ that violates the soundness of Π_{LRZK}. Let $\Pi.\mathsf{Ext}_p$ be the strict polynomial-time extractor that behaves precisely as $\Pi.\mathsf{Ext}$ (up to a given number of steps) as specified in the last part of the proof of Lemma 3.6.

$\mathsf{P}^\star$ works as follows:

1. $\mathsf{P}^\star$ picks at random $i \in [\mathsf{time}_{\mathsf{Sim}_{pR}}]$ and then runs $\Pi.\mathsf{Ext}_p$ with respect to $\mathsf{Sim}^{i\mathsf{V}^\star}_{pR}$. If $\Pi.\mathsf{Ext}_p$ does not give in output a state $\hat{w}$ as part of a witness proving that $\tau \in \Lambda$, then $\mathsf{P}^\star$ aborts.

2. $\mathsf{P}^\star$ then runs the honest prover P of Π_{LRZK} on input $\hat{w}$ for proving to a honest verifier V that $x \in L$ where x is the above special statement (i.e., $x \notin L$).

First of all, the running time of $\mathsf{P}^\star$ is clearly polynomial since both the above steps take polynomial time. Then, we notice that by Lemma 3.6, both Step 1 and 2 correspond to runs without aborting with non-negligible probability. This is due to the fact that the extractor $\Pi.\mathsf{Ext}_p$ fails only with negligible probability and that the extracted state $\hat{w}$ gives to a honest prover of (P, V) non-negligible probability to convince the verifier. Therefore $\mathsf{P}^\star$ succeeds in proving a false statement to honest V with non-negligible probability.

We have proved that if $L \notin \mathsf{BPP}$ then Π_{LRZK} can not be both LRZK and sound. □

4 Impossibility of LRMPC

We now use again our technique of running a protocol in the head of the adversary through leakage queries to show that LRMPC is impossible, therefore solving a problem opened in [BGJK12]. For this simpler result we give only a sketch of the proof and we defer for the additional definitions to [BGJK12]. We stress that the only variation here is that the interactive preprocessing does not take place (as required in the formulation of the open problem in [BGJK12]).

We can show that for many functionalities LRMPC with a leak-free input-encoding phase is impossible. The involved functionalities are the ones such that when they are run multiple times keeping unchanged the input x_i of a honest player P_i, the (short) outputs delivered to the dishonest players reveal more information on x_i than what a single output would reveal. Our impossibility requires just one dishonest player.

For simplicity we will now consider one such functionality: a variation of Yao's Millionaires' Problem, where n players $P_1, \ldots, P_n$ send their inputs to the functionality $\mathcal{F}$ and then $\mathcal{F}$ outputs to all players a bit b specifying whether P_1 is the richest one.

Theorem 4.1 *Consider the n-party functionality $\mathcal{F}$ that on input n k-bit strings $x_1, \ldots, x_n$ outputs to all players the bit $b = 1$ when $x_1 \geq x_j$ for $1 < j \in [n]$ and 0 otherwise. If at least one among $P_2, \ldots, P_n$ is corrupted then there exists no LRMPC for $\mathcal{F}$.*

PROOF. We will sketch the proof since the main ideas were already used in the proof of the impossibility of LRZK.

Assume by contradiction that there exists a secure multi-party protocol Π. Assume wlog that all players are honest except P_n. The adversary Adv controls P_n and works as follows.

1. It sends a leakage query that includes different encodings of the same value $x_2 = \cdots = x_n = 2^{k-1}$ for players $P_2, \dots, P_n$; the leakage query asks for a "virtual" execution of the protocol where P_1 uses its state $\hat{w}_1$, and requires to give in output the output for P_n.

2. It repeats Step 1 changing the value to be used for the $n-1$ encodings of $P_2, \dots, P_n$ (still a unique value for all of them) according to binary search.

3. Adv ends the protocol by giving in output the first two bits of the original (i.e., pre-encoding) input of P_1.

The communication complexity (from honest player to adversary) of this execution through leakage queries is constant. Notice that Adv in the real world obtains the first two bits of w_1, the original input of P_1. Sim in the ideal world does not have such an information since it can perform only one query to $\mathcal{F}$, therefore getting at most one bit. □

5 Acknowledgments

Work supported in part by NSF grants 09165174, 1065276, 1118126 and 1136174, US-Israel BSF grant 2008411, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014 -11 -1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

[ADVW13] Shweta Agrawal, Yevgeniy Dodis, Vinod Vaikuntanathan, and Daniel Wichs. On continual leakage of discrete log representations. In *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 401–420. Springer, 2013.

[AGP14] Prabhanjan Ananth, Vipul Goyal, and Omkant Pandey. Interactive proofs under continual memory leakage. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 164–182. Springer, 2014.

[Bar04] Boaz Barak. Non-black-box techniques in cryptography. PhD Thesis, 2004. `http://www.boazbarak.org/Papers/thesis.pdf`.

[BCH12] Nir Bitansky, Ran Canetti, and Shai Halevi. Leakage-tolerant interactive protocols. In *9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 266–284. Springer, 2012.

[BDL14] Nir Bitansky, Dana Dachman-Soled, and Huijia Lin. Leakage-tolerant computation with input-independent preprocessing. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 146–163. Springer, 2014.

[BG93] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer, 1993.

[BGGL01] Boaz Barak, Oded Goldreich, Shafi Goldwasser, and Yehuda Lindell. Resettably-sound zero-knowledge and its applications. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 116–125. IEEE Computer Society, 2001.

[BGJ$^+$13] Elette Boyle, Sanjam Garg, Abhishek Jain, Yael Tauman Kalai, and Amit Sahai. Secure computation against adaptive auxiliary information. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 316–334. Springer, 2013.

[BGJK12] Elette Boyle, Shafi Goldwasser, Abhishek Jain, and Yael Tauman Kalai. Multiparty computation secure against continual memory leakage. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 1235–1254. ACM, 2012. `http://people.csail.mit.edu/abhishek/papers/LR-MPC-STOC12.pdf`.

[BGK11] Elette Boyle, Shafi Goldwasser, and Yael Tauman Kalai. Leakage-resilient coin tossing. In *Distributed Computing - 25th International Symposium, DISC 2011, Rome, Italy, September 20-22, 2011. Proceedings*, pages 181–196, 2011.

[BGK14] Elette Boyle, Shafi Goldwasser, and Yael Tauman Kalai. Leakage-resilient coin tossing. *Distributed Computing*, 27(3):147–164, 2014.

[BKKV10] Zvika Brakerski, Yael Tauman Kalai, Jonathan Katz, and Vinod Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 501–510. IEEE Computer Society, 2010.

[BSMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, 1991.

[BSW11] Elette Boyle, Gil Segev, and Daniel Wichs. Fully leakage-resilient signatures. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 89–108. Springer, 2011.

[BSW13] Elette Boyle, Gil Segev, and Daniel Wichs. Fully leakage-resilient signatures. *J. Cryptology*, 26(3):513–558, 2013.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 136–145. IEEE Computer Society, 2001.

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 235–244. ACM, 2000.

[DDF14] Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying leakage models: From probing attacks to noisy leakage. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 423–440. Springer, 2014.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 542–552. ACM, 1991.

[DF12] Stefan Dziembowski and Sebastian Faust. Leakage-resilient circuits without computational assumptions. In *9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 230–247. Springer, 2012.

[DHLW10] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Cryptography against continuous memory attacks. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 511–520. IEEE Computer Society, 2010.

[DLWW11] Yevgeniy Dodis, Allison B. Lewko, Brent Waters, and Daniel Wichs. Storing secrets on continually leaky devices. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 688–697. IEEE, 2011.

[DNS98] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 409–418. ACM, 1998.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 293–302. IEEE Computer Society, 2008.

[FRR$^+$10] Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting circuits from leakage: the computationally-bounded and noisy cases. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 135–156. Springer, 2010.

[GJS11] Sanjam Garg, Abhishek Jain, and Amit Sahai. Leakage-resilient zero knowledge. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 297–315. Springer, 2011.

[GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304. ACM, 1985.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229. ACM, 1987.

[GR10] Shafi Goldwasser and Guy N. Rothblum. Securing computation against continuous leakage. In *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 59–79. Springer, 2010.

[GR12] Shafi Goldwasser and Guy N. Rothblum. How to compute in the presence of leakage. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 31–40. IEEE Computer Society, 2012.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481. Springer, 2003.

[MR04] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In *First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, volume 2951 of *Lecture Notes in Computer Science*, pages 278–296. Springer, 2004.

[Pan14] Omkant Pandey. Achieving constant round leakage-resilient zero-knowledge. In *11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, volume 8349 of *Lecture Notes in Computer Science*, pages 146–166. Springer, 2014.