Unconditionally Secure Computation Against Low-Complexity Leakage[*]

Andrej Bogdanov
Chinese University of Hong Kong

Yuval Ishai
Technion

Akshayaram Srinivasan[†]
Tata Institute of Fundamental Research

August 27, 2021

Abstract

We consider the problem of constructing leakage-resilient circuit compilers that are secure against global leakage functions with bounded output length. By global, we mean that the leakage can depend on all circuit wires and output a low-complexity function (represented as a multi-output Boolean circuit) applied on these wires. In this work, we design compilers both in the stateless (a.k.a. single-shot leakage) setting and the stateful (a.k.a. continuous leakage) setting that are *unconditionally* secure against AC^0 leakage and similar low-complexity classes.

In the stateless case, we show that the original private circuits construction of Ishai, Sahai, and Wagner (Crypto 2003) is actually secure against AC^0 leakage. In the stateful case, we modify the construction of Rothblum (Crypto 2012), obtaining a simple construction with unconditional security. Prior works that designed leakage-resilient circuit compilers against AC^0 leakage had to rely either on secure hardware components (Faust et al., Eurocrypt 2010, Miles-Viola, STOC 2013) or on (unproven) complexity-theoretic assumptions (Rothblum, Crypto 2012).

1 Introduction

There is a rich body of work on protecting computations that involve sensitive data against partial information leakage. This line of work is motivated by practical side-channel attacks that use physical measurements such as running time [Koc96] or power consumption [KJJ99] to compromise secret keys embedded in cryptographic hardware or software. The recent high-profile Meltdown, Spectre, and Foreshadow attacks [KGG+18, LSG+18, BMW+18] demonstrated the vulnerability of most modern computer systems to these kinds of attacks. While the latter attacks are quite involved, exploiting hardware optimizations such as speculative execution and caching, they further motivate the basic study of provable resilience to side-channel attacks.

A clean theoretical model that captures the goal of protecting general computations against leakage is that of a *leakage resilient circuit compiler* (LRCC). Here the computation is modeled as a logical circuit, and the leakage as a function applied to the internal wires of the circuit. The goal of a LRCC is to randomize the computation of a given circuit in a way that resists broad classes

[*]Preliminary version of this article appeared in Crypto 2019 [BIS19].
[†]Work done while at UC Berkeley.

of leakage while at the same time respecting the input-output relation of the original circuit. The problem of LRCC has many flavors, depending on the computational model and the type of leakage.

A crude form of LRCC was already given in the 1980s by the seminal works on secure multiparty computation [Yao86, GMW87, BGW88, CCD88]. Such protocols distribute computations across multiple parties in a way that resists leakage from a bounded number of parties. The work of Ishai, Sahai, and Wagner (ISW) [ISW03] initiated a more explicit and refined study of LRCC at the circuit level, but still focused on the case of localized "probing attack" leakage that applies to a bounded number of circuit wires. In spite of its restricted nature, this leakage model turned out to be quite relevant to practical defenses against side-channel attacks. This is due in part to the simplicity of the constructions and the ability of the same leakage model to accommodate more realistic *noisy* leakage [FRR$^+$10, DDF14] that obtains an independent noisy measurement of every wire in the circuit. LRCCs in this model have been the subject of a large body of theoretical and applied work (see, e.g., [RP10, Ajt11, CPRR13, Cor14, DFS15, BCPZ16, BBP$^+$16, FPS17, AIS18] and references therein).

Originating from the works of Micali and Reyzin [MR04] and Faust et al. [FRR$^+$10, FRR$^+$14], another line of work focused on accommodating more general types of leakage classes that apply restricted types of functions to *all* wires in the circuit. In particular, Faust et al. [FRR$^+$10] presented a variant of the ISW compiler that employs small leak-free hardware components to protect against any class of "computationally simple" leakage functions for which strong average-case lower bounds are known. The most prominent example is that of AC^0 *leakage*, computed by constant-depth polynomial-size circuits with unbounded fan-in AND/OR/NOT gates and a bounded number of outputs. Subsequent works along this line studied LRCCs for different classes of global leakage under a variety of trusted hardware or setups and computational intractability assumptions [GR10, JV10, GJS11, BGJK12, BCH12, DF12, Rot12, GR12, BGJ$^+$13, MV13, BDL14, Mil14, DLZ15, GIM$^+$16, GIW17, BDIR18].

Constant-depth leakage. The focus of this work is mainly on the class of AC^0 leakage and related constant-depth complexity classes, such as AC^0 augmented with additional mod-p gates. This type of leakage strictly generalizes the ISW leakage model, which as discussed above is relevant to many realistic scenarios. Moreover, while the class AC^0 does not capture some natural leakage functions, such as ones that take weighted sums of many wire values, it does apply to a wide variety of natural attacks. For instance, suppose that a system crashes if a secret value represented by a wire bundle is in a certain forbidden range, and there are many such wire bundles that may lead to the system crashing. Then, whether the system crashes at a given moment is a single bit of depth-3 AC^0 leakage that can be observed by the outside world. One can similarly cast in this class other types of natural leakage functions that take the conjunction, disjunction, maximum, or minimum of values that can themselves be computed by low-depth circuits.

Stateless vs. stateful LRCC. Before describing our contributions, it is instructive to present the current state of the art in a more precise way. The ISW paper introduced two variants of the LRCC problem: a simpler *stateless* variant and a more complex *stateful* variant. The stateless variant captures standard computations that map a secret input to a secret or public output, where the computation is subject to a single round of *one-shot leakage*. For instance, this scenario can apply to zero-knowledge authentication by a hardware device, or computations performed by payment terminals and access control readers (see [GIW17] for further discussion). In a more theoretical context, stateless LRCCs have also been applied towards constructing different zero-knowledge flavors of probabilistically checkable proofs [IWY16, Wei19]. The *stateful* variant of LRCCs captures a system (such as a personal computer or an IoT device) with persistent memory that may store secrets. Users interacting with this system can feed it with a sequence of inputs and observe the resulting outputs. For instance, think of an encryption device that stores a secret encryption key, takes a plaintext as input and produces a ciphertext as output. Stateful LRCCs may be subject to *continuous leakage* that applies a different leakage function in each round. To help defend against this kind of leakage, they are allowed to refresh their internal state.

More formally, in the *stateless* variant of LRCC, the goal is to compile a (deterministic, stateless) circuit C into a randomized circuit $\widehat{C}$, such that together with leak-free randomized input encoder Enc and output decoder Dec we get the following correctness and security guarantees: (1) For any input x, we have $\mathsf{Dec}(\widehat{C}(\mathsf{Enc}(x))) = C(x)$; (2) For any admissible leakage function $\ell \in \mathcal{L}$, applying ℓ to the internal wires of the computation $\widehat{C}(\mathsf{Enc}(x))$ reveals essentially nothing about x. To rule out a trivial solution in which the entire computation is carried out by the leak-free components Enc and Dec, these components are required to be *universal* in the sense that they depend only on the input and output size of C and not on C itself. The ISW construction protects computations against leakage that involves a bounded number of wire-probes. That is, the leakage ℓ can output the values of t wires in $\widehat{C}$. Here we are interested in a bigger class $\mathcal{L}$ that includes constant-depth circuits with t bits of output.

The *stateful* variant of LRCC considers the more challenging goal of protecting computations against *continual leakage*. Here the ideal functionality is specified by a deterministic, *stateful* circuit C, mapping the current input and state to the current output and the next state. The input and output are considered to be public whereas the state is secret. The goal, as before, is to transform C into a leakage-resilient randomized circuit $\widehat{C}$. The circuit $\widehat{C}$ is initialized with some randomized encoding $\hat{s}_0$ of the initial secret state s_0 of C. The computation can then proceed in a virtually unlimited number of rounds, where in each round $\widehat{C}$ receives an input, produces an output, and replaces the old encoding of the secret state by a fresh encoding of a new state. The correctness goal is to ensure that $\widehat{C}[\widehat{s}_0]$ has the same input-output functionality as $C[s_0]$. The security goal is defined again with respect to a class $\mathcal{L}$ of leakage functions, where the adversary may adaptively choose a different function $\ell \in \mathcal{L}$ in each round. The security goal is to ensure that whatever the adversary learns by interacting with $\widehat{C}[\widehat{s}_0]$ and by additionally observing the leakage, it can simulate by interacting with $C[s_0]$ without obtaining any leakage.

State of the art. Existing results of LRCCs for AC^0 and similar constant-depth leakage classes leave a number of basic questions open. In the stateful case, the works of Faust et al. [FRR+10] and Miles and Viola [MV13] yield constructions that require small but leak-free trusted hardware components, whose number is linear in the size of C and whose size grows with a statistical security parameter. Alternatively, Rothblum [Rot12] showed how to eliminate the trusted hardware components, but at the cost of further complicating the construction and relying on an unproven complexity theoretic conjecture (the so-called "IPPP conjecture") that remains open to date. In the stateless case, the trusted hardware components in the constructions of [FRR+10, MV13] can be replaced by correlated random input bits that are fed directly into the stateless circuit in addition to the input x [MV13, BIVW16, GIW17]. However, this requires the user of the leakage-resilient circuit $\widehat{C}$ to work at least as hard as computing C rather than simply feed $\widehat{C}$ with its input.

We note that unlike the case of security against *noisy* leakage, which is implied by security against probing attacks [DDF14], this is *not* the case for security against AC^0 leakage. Indeed, there are pairs of distributions over $\{0,1\}^N$ that cannot be distinguished by probing any $N^{0.99}$ of their bits, and yet they *can* be distinguished by AC^0 circuits with one bit of output [BIVW16, BKT19]. In the stateful case, an additional difficulty stems from the need to prove simulation-based security rather than mere indistinguishability by AC^0 circuits. The efficient simulation requirement poses a major challenge in some related contexts [IWY16].

1.1 Our Contribution

In this work, we improve the above state of the art in both the stateless and stateful case by proving two main unconditional results.

In the *stateless* case (with one-shot leakage), we show that the original ISW construction [ISW03], which is quite simple and concretely efficient, is actually unconditionally secure against a much wider class of low-complexity leakage functions that includes AC^0. We also show similar results for leakage computed by AC^0 circuits with mod-p gates, for a prime modulus $p > 2$, though in this case our security only follows from standard complexity-theoretic conjectures. In contrast to previous constructions from [MV13, BIVW16, GIW17], here the circuit $\widehat{C}$ directly computes on the input x and does not require additional correlated random inputs or trusted leak-free hardware. This construction is also simpler and more efficient than the (conditional) construction from [Rot12].

In the *stateful* case (with continuous leakage), we modify the previous construction of Rothblum [Rot12], obtaining the first construction that *unconditionally* resists AC^0 leakage without relying on trusted leak-free hardware.

At a higher level of generality, both of our constructions satisfy a composition theorem of the following form (Theorems 4.1 and 5.1): For any given class of leakage functions $\mathcal{L}$, if parity has low correlation with $\mathcal{L}$ composed with NC^0 (namely, functions where each output depends on a constant number of inputs), then our constructions are secure against leakage from $\mathcal{L}$. For $\mathcal{L} = \mathsf{NC}^0$ we recover the ISW result, for $\mathcal{L} = \mathsf{AC}^0$ we obtain our main result, and for $\mathcal{L} = \mathsf{AC}^0[\mathrm{mod}\, p]$ we get the extension to constant-depth circuits with mod p gates, assuming this class has low correlation with parities.

Here is a formal statement of the results in these cases of interest. For the definitions of LRCC for stateful and stateless circuits, see Definition 3.10 and Definition 3.12 respectively. The corresponding constructions are described in Sections 4 and 5.

Corollary 1.1 *The ISW compiler when applied to circuits of size S and input length k is a $k\varepsilon$-leakage resilient stateless circuit compiler against the following classes, where n is the security parameter:*

1. *Functions that depend on the values of at most $(n-1)/2$ wires, with $\varepsilon = 0$,*
2. *Unbounded fan-in AND/OR/NOT circuits of size $s - O(n^2 S)$, depth d, and $c_d n/(\log s)^d$ outputs, with $\varepsilon = 2^{-c_d n/(\log s)^d}$,*
3. *Unbounded fan-in AND/OR/NOT/MOD$_p$ circuits of size $s - O(n^2 S)$, depth d, and m outputs, assuming n-bit random parity-0 and parity-1 strings are $2 \cdot 3^{-m}\varepsilon$-indistinguishable by such circuits of size s and depth $d+1$ (and one output).*

Here c_d is a constant that depends on d only. Part 1 recovers the stateless security result of Ishai, Sahai, and Wagner. Parts 2 and 3 are new.

Corollary 1.2 *There exists a construction of LRCC for a class of stateful circuits of size S that is $O(\varepsilon T(S+n))$-leakage resilient stateful circuit compiler against the following leakage classes, where T, S, and n are the number of rounds of the leakage experiment, the circuit size, and the security parameter, respectively:*

1. *Unbounded fan-in AND/OR/NOT circuits of size $2^{n^{O(1/d)}} - O(n^3 S)$, depth d, and $n^{O(1/d)}$ outputs, with $\varepsilon = 2^{-n^{O(1/d)}}$.*

2. *Unbounded fan-in AND/OR/NOT/MOD$_p$ circuits of size $s - O(n^3 S)$, depth d, and m outputs, assuming n-bit random parity-0 and parity-1 strings are $2 \cdot 3^{-m}\varepsilon$-indistinguishable by such circuits of size $O(2^m s)$ and depth $d+1$ (and one output).*

1.2 Open Problems

Our work takes the first step in constructing unconditionally secure leakage-resilient circuit compilers against low-complexity leakage and it opens up several intriguing directions.

1. Can we show that the stateful version of the ISW compiler is secure against AC^0 leakage? In our work, we show that the ISW construction in the stateless setting is secure against AC^0 leakages. Our construction in the stateful case takes a different approach and currently, we do not know if there is any attack against the stateful version of the ISW construction under AC^0 leakage.

2. Can we improve the asymptotic efficiency of our constructions in the stateless setting? The ISW construction incurs a $O(n^2)$ blow-up in the circuit size but there are constructions proven secure against wire-probe leakage (see [DIK10]) that have a blow-up of $O(\mathsf{poly}\log(n))$. Can we use these techniques to get a similar efficiency improvement against AC^0 leakages?

3. Can we use our techniques to construct compilers that have provable guarantees against leakages from richer complexity classes such as NC^1 (assuming standard complexity theoretic conjectures) without the use of trusted hardware? This would improve the results of Miles and Viola [MV13, Mil14].

2 Our Techniques

In this section, we give a high-level overview of our techniques for constructing a leakage resilient compiler that is unconditionally secure against AC^0 leakage. We start with a brief overview of the prior approaches and highlight the limitations of these approaches in obtaining an unconditional result. Next, in Section 2.1, we give an overview of the proof that the original private circuit construction of Ishai, Sahai and Wagner [ISW03] is secure against AC^0 leakage in the stateless a.k.a. single-shot leakage setting. Finally, in Section 2.2, we discuss our construction of a leakage resilient circuit compiler in the stateful a.k.a. continuous leakage setting.

Prior Approaches. All the prior works [ISW03, FRR$^+$10, Rot12, MV13, Mil14] (including ours) follow the same high-level blue print in constructing a leakage resilient circuit compiler. Each wire in the original circuit C is transformed into a "bundle" of n-wires in the compiled circuit $\widehat{C}$ such that the bundle encodes the bit carried by the wire (using a suitable encoding procedure). Few examples of the encoding procedures used in the prior work are the (i) parity encoding [ISW03, FRR+10, Rot12] i.e., the parity of the wire bundle is equal to the value carried by the wire and (ii) group encoding [MV13, Mil14] i.e., each element in the bundle is represented as an element of an alternating group and the product of the group elements encodes the bit carried by the wire. For concreteness, let us assume that the wires are encoded using the parity encoding. The next step in these constructions is to implement the addition and the multiplication gates over the wire bundles. That is, every gate $g \in \{+, *\}$ in the original circuit C, is transformed into a gadget $\widehat{g}$ that takes in 2 wire bundles, say $\mathbf{a}, \mathbf{b} \in \{0,1\}^n$ and outputs a wire bundle $\mathbf{c}$ such that parity of $\mathbf{c}$ is equal to $g(\oplus\mathbf{a}, \oplus\mathbf{b})$. Thus, evaluating these gate gadgets in $\widehat{C}$ will eventually lead us to the output wire bundles which are finally decoded by computing their parity. This construction ensures correctness i.e., the compiled circuit computes the same function as that of the original circuit. However, to prove security, these works required an additional refreshing gadget (denoted as $\mathsf{Refresh}$). The refreshing gadget takes in a wire bundle $\mathbf{x}$ and outputs a random bundle $\mathbf{y}$ conditioned on $\oplus\mathbf{y} = \oplus\mathbf{x}$. In other words, this gadget refreshes the randomness used in the encoding. To get a secure construction, the implementation of each gate gadget $\widehat{g}$ were augmented in such a way that the output wire bundle, say $\mathbf{c}$ is sent through the $\mathsf{Refresh}$ gadget and the resultant wire bundle is the new output. At an intuitive level, this leads to a secure construction as the $\mathsf{Refresh}$ gadget ensures that the randomness used in encoding the output of each gate is refreshed and hence, the leakage that has been accumulated as a result of the $\widehat{g}$ computation does not propagate to the higher layers. This allowed the prior works to argue security against specific leakage classes such as AC^0 circuits. However, the task of implementing this refreshing gadget is highly challenging and this is the primary reason that the prior works had to rely on secure hardware components [FRR+10, MV13, Mil14] or computational assumptions [Rot12]. Specifically, Faust et al. used a secure hardware component to generate a random vector $\mathbf{z}$ whose parity is 0 and implemented the $\mathsf{Refresh}$ gadget as $\mathbf{y} = \mathbf{x} + \mathbf{z}$. This ensures that $\mathbf{y}$ has the same parity as that of $\mathbf{x}$ and additionally, it is distributed randomly conditioned on its parity being fixed. Rothblum removed the need of secure hardware components by generating random encodings of 0 using a more involved procedure (that will be explained later) but had to rely on a computational assumption in the proof of security. In the next two subsections, we discuss our approach of dealing with the problem of generating a random encoding of 0, first in the stateless setting and then in the more complicated stateful setting.

2.1 Unconditional Result in the Stateless Setting

The key insight behind our unconditional result in the stateless setting is that refreshing the output of every gate gadget is actually an overkill and a far weaker property called as "local sampleability" is sufficient. Before we go into the details, let us first give the definition of a local sampler. A circuit $\mathsf{Samp}(\mathbf{x}; r)$ ($\mathbf{x} \in \{0,1\}^n$ is the regular input and r is the randomness) is said to be a 2-local sampler if each output bit of the circuit depends on at most two bits of the regular input $\mathbf{x}$. It can be easily seen that for every r, $\mathsf{Samp}(\mathsf{PAR}(n,0); r)$ is indistinguishable to $\mathsf{Samp}(\mathsf{PAR}(n,1); r)$ by AC^0 circuits where $\mathsf{PAR}(n,b)$ is an uniform distribution over n-bit strings whose parity is b.

The main technical lemma which allows us to prove security in the stateless setting is the following. Fix the encodings of all input bits except one, say $\mathbf{x}$ and let $\widehat{C}$ be the compiled circuit in the construction of Ishai, Sahai and Wagner [ISW03]. Then, the distribution of the wires in $\widehat{C}$ is identical to the output of a 2-local sampler $\mathsf{Samp}(\mathbf{x}; r)$ for an uniformly chosen r. This allows us to prove an unconditional result as we can go over a sequence of hybrids such that in each hybrid, we fix the encodings of all bits except one (say, $\mathbf{x}$), use $\mathsf{Samp}(\mathbf{x}; r)$ to generate the distribution of all the wires in $\widehat{C}$ and then conclude that the wire distribution is indistinguishable to AC^0 circuits when $\mathbf{x}$ encodes the bit 0 or 1. We stress that unlike the prior unconditional results in the stateless setting [FRR+10, MV13], our construction does not require a source of correlated randomness generated in a leak-free manner. We also remark that in the prior results, the number of bits of this correlated randomness string is very large and in the worst case, could be as large as the circuit itself.

Before we delve into the details of the proof of the main lemma, let us first recall the construction of Ishai, Sahai and Wagner [ISW03]. As mentioned before, in this construction, each wire in the original circuit is transformed into a bundle of n wires such that the parity of this wire bundle is equal to the value carried by the wire. Given this encoding, implementing the addition gadget is simple. It takes in two wire bundles, $\mathbf{a}, \mathbf{b} \in \{0,1\}^n$ and outputs $\mathbf{c} = \mathbf{a} + \mathbf{b}$. We give the details of the multiplication gadget below.

Construction 2.1 *On input two wire bundles* $\mathbf{a}$ *and* $\mathbf{b}$*, the multiplication gadget does the following:*

1. *Define the matrix* $\mathbf{M} \in \{0,1\}^{n\times n}$ *such that* $M_{i,j} = a_i b_j$.
2. *For every* $1 \le i, j \le n$ *and* $i < j$*, choose a random bit* $z_{i,j}$.
3. *For every* $1 \le i, j \le n$ *and* $i < j$*, set* $z_{j,i} = z_{i,j} \oplus (M_{j,i} \oplus M_{i,j})$.
4. *For every* $1 \le i \le n$*, set* $c_i = (\oplus_{j \ne i} z_{i,j}) \oplus M_{i,i}$.
5. *Output* $\mathbf{c} = (c_1, \ldots, c_n)$.

Correctness of both the gadgets is straightforward to verify. Let us fix the encodings of all the input bits except one, say $\mathbf{x}$. To prove the main lemma, we need to show that the wire distribution in the compiled circuit conditioned on this fixing is identical to the output of a 2-local sampler.

Proof Overview. We prove this lemma via an inductive argument. We first prove that the distribution of the internal wires in an addition and a multiplication gate is identical to a locally sampleable distribution. We then use induction to prove that the wire assignment in the entire circuit is locally sampleable.

Local sampleability of addition gadget is trivial and the main challenge is to show local sampleability of multiplication gadget. For simplicity, let us consider a multiplication gate at the first layer of the circuit where one input is $\mathbf{x}$ (which is the non-fixed encoding) and the other input is $\mathbf{b}$ (for some fixed $\mathbf{b}$). The other cases are dealt in section 4 of our paper. We need to show that for any $\mathbf{b}$, there exists a 2-local sampler $\mathsf{Samp}_{\mathsf{mult}}(\mathbf{x}; z')$ such that the output of the sampler (for an uniform z') is identical to the distribution of the internal wire assignments of a multiplication gate on input $\mathbf{x}, \mathbf{b}$.

At first inspection, it appears that the internal wire assignments of the multiplication gadget are "non-local." Specifically, consider the wires in the computation of c_n; it depends on every bit of $\mathbf{x}$. So the main question is how do we prove that the wires are 2-locally sampleable? The key insight is that while the internal wires of the multiplication gadget could be non-local, they are distributed identically to a 2-locally sampleable distribution. So, we need to demonstrate a 2-locally sampleable distribution (which is the output of a $\mathsf{Samp}_{\mathsf{mult}}$) and argue that this distribution is identical to the distribution of the internal wires of the multiplication gadget. We now give details of such a sampler $\mathsf{Samp}_{\mathsf{mult}}$. On input $\mathbf{x}$ and uniform randomness z', $\mathsf{Samp}_{\mathsf{mult}}$ (that depends on $\mathbf{b}$) does the following:

1. Define the matrix $\mathbf{M} \in \{0,1\}^{n \times n}$ where the (i,j)-th element $M_{i,j} = x_i \cdot b_j$.
2. For every $1 \leq i \leq n, 1 \leq j \leq n$ and $i < j$, choose a random bit $z'_{i,j}$ and define $z_{i,j} = \underline{z'_{i,j} \oplus M_{i,j}}$.
3. For every $1 \leq i \leq n, 1 \leq j \leq n$ and $i < j$, set $z_{j,i} = z_{i,j} \oplus (M_{j,i} \oplus M_{i,j})$.
4. For every $1 \leq i \leq n$, set $c'_i = (\oplus_{j \neq i} z_{i,j}) \oplus M_{i,i}$.
5. Output $\mathbf{M}$, $\{z_{i,j}\}_{i<j}$, all the wires in the computation of $\{z_{i,j}\}_{i>j}$ and the computation of $\{c'_i\}_{i \in [n]}$ along with the vector $\mathbf{c}' = (c'_1, \ldots, c'_n)$ (which are the output wires).

The only difference between the wire assignments output by $\mathsf{Samp}_{\mathsf{mult}}$ and the actual wire assignments in multiplication gate is how $\{z_{i,j}\}_{i<j}$ is set. Note that if z' is chosen uniformly at random then the distribution of $\{z_{i,j}\}_{i<j}$ is identical to the uniform distribution. Thus, the wire assignment output by $\mathsf{Samp}_{\mathsf{mult}}$ is identical to the actual wire assignment in the implementation of the multiplication gate for a randomly chosen z. To see the 2-local sampleability of $\mathsf{Samp}_{\mathsf{mult}}$, observe that for any $i < j$, $z_{i,j}$ depends only on x_i. Furthermore, for any $i > j$, it can be observed that $z_{i,j} = z'_{j,i} \oplus M_{i,j}$ depends on only x_i and wires used in computing $z_{i,j}$ and is therefore a 2-local function in $\mathbf{x}$. These two observations imply that for every $i \in [n]$, computing c'_i depends only on x_i and hence the wires in this computation are locally sampleable. This shows that the output of $\mathsf{Samp}_{\mathsf{mult}}$ is a 2-local distribution. Combining this with the inductive argument allows us to obtain an unconditional result in the stateless setting.

2.2 Unconditional Result in the Stateful Setting

In this subsection, we give a high level overview of our construction of a leakage-resilient circuit compiler against AC^0 circuits in the stateful setting that has unconditional security. As mentioned before, the prior results in this setting either relied on secure hardware components or on computational assumptions.

Main Challenges. In the stateful setting, there are two key challenges that we need to overcome. The first challenge is dealing with absence of a trusted decoder. In the stateless setting, a trusted decoder was available and this allowed the simulator to "cheat" by hardwiring the correct output in the trusted decoder such that even when the circuit is run on some junk inputs, the output obtained is consistent with the actual output. However, in the stateful case, no such trusted decoder is available and this makes the task of simulation much harder. In this case, the simulator must somehow incorporate the correct output (without knowing the actual input) in the wire distribution such that a leakage function cannot distinguish this from the real word distribution. When considering leakage classes such as AC^0 functions, this task is even more challenging as these functions can check local consistency of the gates. The second challenge in the stateful setting is the necessity to refresh the randomness. Unlike the stateless setting where we observed that local sampleability is sufficient, in the stateful case, we need to additionally refresh the randomness used in the encoding procedure. To see why this is the case, consider a stateful circuit that has a PRF key k as its state and computes $\mathsf{PRF}(k,x)$ on a regular input x. If the randomness of the key k is not refreshed across multiple queries, then in $O(n|k|)$ leakage queries, the entire key can be successfully retrieved by leakage functions that output a single bit. Thus, we need to refresh the randomness of the state bundles across queries and for technical reasons, we also need to refresh the randomness of the output of every gate.

Rothblum's Construction. The starting point of our construction is the work of Rothblum [Rot12] who showed that under a complexity theoretic assumption referred to as "Inner Products with Pre-Processing" (IPPP),[1] there exists a construction of a leakage resilient circuit compiler against AC^0 in the stateful setting. Unfortunately, this assumption is unproven and even the state of highly restricted versions of the assumption such as allowing only linear functions in the pre-processing phase [ABG+14] is far from being resolved. In the rest of this subsection, we first give a high level overview of the construction of Rothblum, indicate why the IPPP assumption is needed, and then discuss our approach of removing the need for the assumption.

Recall that in the stateful setting, the output of every gate is refreshed and thus, the first step is to implement the $\mathsf{Refresh}$ gadget. This $\mathsf{Refresh}$ gadget in fact helps in overcoming both the challenges that we discussed earlier. Firstly, it helps in refreshing the randomness and thus, helps in overcoming the second challenge. To overcome the first challenge, we additionally send the wire bundles coming out of the output gate through the $\mathsf{Refresh}$ gadget and compute the parity of the resultant output. In the ideal world distribution, the simulator will change the internal workings of the $\mathsf{Refresh}$ gadget such that instead of only refreshing the randomness, this gadget could also switch the parity when needed. This helps the simulator to hardcode the correct output of the circuit even when it is run with some junk input.

Now, to implement the $\mathsf{Refresh}$ gadget, it is sufficient to generate a random encoding of the bit 0. The main technical contribution in Rothblum's work is a method to securely generate a random encoding of 0 without the use of hardware components. This is done as follows. A generator matrix $\mathbf{G} \in \{0,1\}^{n \times 2n}$ is chosen uniformly at random subject to the parity of each column of $\mathbf{G}$ being 0. This generator matrix is part of the state of the compiled circuit $\widehat{C}$. Whenever a random encoding of 0 is required, choose $\mathbf{r}$ uniformly at random from $\{0,1\}^{2n}$ and compute $\mathbf{G} \cdot \mathbf{r}$. It is straightforward to see that the resultant vector is statistically close to a random vector whose parity is 0. This vector is then used in the $\mathsf{Refresh}$ gadget. In Rothblum's work, the circuit C_{MV} for computing the matrix-vector product $\mathbf{G} \cdot \mathbf{r}$ is the canonical $O(n^2)$ sized circuit.

Although the construction is simple, the proof that it is secure in the presence of AC^0 leakage is involved and relies on the (unproven) IPPP assumption. In Rothblum's security proof it is argued that the actual distribution over the wires of C_{MV} is AC^0-indistinguishable from the alternative distribution on the wires when its input $\mathbf{G}$ is chosen uniformly at random. The distribution over the inputs $(\mathbf{G}, \mathbf{r})$ in the two distribution is straightforward to prove. To argue indistinguishability of the other circuit wires, Rothblum shows that their values can be sampled in AC^0 assuming the availability of preprocessed representations of $\mathbf{G}$ and $\mathbf{r}$. The IPPP assumption is used to argue that the preprocessing does not affect AC^0-indistinguishability.

Our Approach. In this work, we remove the need for the IPPP assumption by designing a new gadget called "$\mathsf{RandZero}$" that generates a random encoding of 0. Crucially, unlike the circuit C_{MV}, it has a special property that its wire assignments are locally sampleable. This allows us to get rid of the pre-processing phase in Rothblum's paper and obtain an unconditionally secure construction. We now give more details of our approach.

Like in Rothblum's construction, we choose a generator matrix $\mathbf{G} \leftarrow \{0,1\}^{n \times n}$ uniformly at

[1]Let D'_0, D'_1 be uniform distributions over $2n$-bit strings such that for every $(\mathbf{x}, \mathbf{y}) \in D'_b$, $\langle \mathbf{x}, \mathbf{y} \rangle = b$. IPPP states that it is hard for AC^0 circuits to distinguish between D'_0 and D'_1 even when given $f(\mathbf{x})$ and $g(\mathbf{y})$ for arbitrary functions f and g with polynomial output size (alternatively, polynomial-time computable f and g). See [FIKK20] for the current state of the art on the IPPP problem and related work in complexity theory.

random subject to its column parity being 0 and make it part of the state. When we have to generate a random encoding of 0, we choose $\mathbf{r}$ uniformly at random and compute $\mathsf{RandZero}(\mathbf{G}, \mathbf{r})$. Below, we give the description of this gadget.

Construction 2.2 *Given a matrix $\mathbf{G} \in \{0,1\}^{n\times n}$ and a vector $\mathbf{r} \in \{0,1\}^n$,* RandZero *does the following:*

1. *Define the matrix $\mathbf{M} \in \{0,1\}^{n\times n}$ where the (i, j)-th element $M_{i,j} = G_{i,j}r_j$.*
2. *For every $1 \le i \le n, 1 \le j \le n$ and $i < j$, choose a random bit $z_{i,j}$.*
3. *For every $1 \le i \le n, 1 \le j \le n$ and $i < j$, set $z_{j,i} = z_{i,j} \oplus (M_{j,i} \oplus M_{i,j})$.*
4. *For every $1 \le i \le n$, compute $c_i = (\oplus_{j\neq i} z_{i,j}) \oplus M_{i,i}$.*
5. *Output $\mathbf{c} = (c_1, \ldots, c_n)$.*

We first make a couple of simple observations. The first observation is that the parity of the output $\mathbf{c}$ is same as that of the vector $\mathbf{G} \cdot \mathbf{r}$. The second observation is that the distribution of $\mathbf{c}$ is uniformly random subject to its parity being equal to parity of the vector $\mathbf{G} \cdot \mathbf{r}$. Thus, when the column parity of $\mathbf{G}$ is 0, we can use the output of this gadget to refresh the randomness.

Notice that the above gadget has a lot of similarities with the multiplication gadget in the work of Ishai, Sahai and Wagner [ISW03] (described in Construction 2.1). In fact, the only difference is how the matrix $\mathbf{M}$ is defined. We thus, extend the local sampleability property that we proved for Construction 2.1 to this construction. Specifically, to show that for a fixed $\mathbf{G}$, the wires of the gadget are locally sampleable in the input $\mathbf{r}$, we define $z_{i,j}$ for $i < j$ to be a random bit plus $M_{j,i}$. This distribution is identical to the wires of the actual gadget. By definition, for every $i < j$, $z_{i,j}$ depends only on r_i. It can also be shown that for every $i > j$, $z_{i,j} = z_{j,i} \oplus M_{i,j} \oplus M_{j,i}$ also depends only on r_i. The wires of the gadget are therefore locally sampleable. We also prove local sampleability in the case when $\mathbf{r}$ is fixed and the wires of the gadget are a local function of the input $\mathbf{G}$. These two kinds of local sampleability are crucially used in the proof of security in moving from the real distribution to the simulated distribution.

In the actual proof of security, we go over a sequence of hybrids (similar to the hybrid sequence used in Rothblum's work) and show that each neighboring pair of hybrids in the sequence is indistinguishable to AC^0 leakage using the local sampleability property of our RandZero gadget. The complete proof is given in Section 5.

3 Preliminaries

Notation We will denote vectors by bold lowercase letters (e.g., $\mathbf{x}$) and matrices with bold uppercase letters (e.g., $\mathbf{M}$). By default, all vectors will be column vectors. We will denote the i-th entry of a vector $\mathbf{x}$ by x_i and the (i, j)-th entry of the matrix $\mathbf{M}$ by $M_{i,j}$. We use $\mathbf{e}_k \in \{0,1\}^n$ for the unit vector whose k-th coordinate is 1 and the rest of the coordinates to be 0.

We use the notation $\mathsf{W}[C]$ for the vector of wire values of a circuit C (under a canonical ordering consistent with the direction of evaluation), and $\mathsf{PAR}(n, b)$ for the distribution on n-bit strings that is chosen uniformly at random subject to having parity b.

3.1 Indistinguishability

Definition 3.1 (Statistical distance) *Let D_1 and D_2 be two distributions on a set S. The statistical distance between D_1 and D_2 is defined to be:*

$$\Delta(D_1, D_2) = \max_{T \subseteq S} |D_1(T) - D_2(T)| = \frac{1}{2} \sum_{s \in S} |\Pr[D_1 = s] - \Pr[D_2 = s]|$$

We say that D_1 is ε-close to D_2 if $\Delta(D_1, D_2) \leq \varepsilon$, and ε-far otherwise.

Lemma 3.2 (Triangle Inequality) $\Delta(D_1, D_3) \leq \Delta(D_1, D_2) + \Delta(D_2, D_3)$.

Definition 3.3 (ε-indistinguishability) *Let X and Y be two distribution over the same domain. We say that (X, Y) is ε-indistinguishable by a class of functions $\mathcal{C}$ if for every $C \in \mathcal{C}$, $\Delta(C(X), C(Y)) \leq \varepsilon$.*

3.2 Circuit complexity

A class of functions $\mathcal{C}$ is *closed under restriction* (resp., *negation*) if for every f in $\mathcal{C}$, the function obtained by fixing the value of any input (resp., negating it) is also in $\mathcal{C}$.

The composition $\mathcal{C} \circ \mathcal{C}'$ consists of all functions $(f \circ f')(x) = f(f'(x))$, where $f \in \mathcal{C}$ and $f' \in \mathcal{C}'$.

We use $\mathsf{NC}^0[c]$ for the class of all multi-input, multi-output Boolean functions in which every output depends on at most c inputs, $\mathsf{AC}^0(d, s, m)$ for the class of circuits that use unbounded fan-in AND-OR-NOT gates, have depth d, size at most s and m output bits, and $\mathsf{AC}^0[B](d, s, m)$ for circuits that may have other types of basis gates B that are closed under negation. If the input or output length is unrestricted or clear from context it is left out of the notation. The following claim follows directly from the definition.

Claim 3.4 $\mathsf{NC}^0[c] \circ \mathsf{NC}^0[c'] \subseteq \mathsf{NC}^0[cc']$, $\mathsf{AC}^0(d, s, m) \circ \mathsf{NC}^0[c] \subseteq \mathsf{AC}^0(d+1, s+n \cdot 2^c)$, *and* $\mathsf{AC}^0[B](d, s, m) \circ \mathsf{NC}^0[c] \subseteq \mathsf{AC}^0[B](d+2, s+n \cdot 2^c)$ *where n is the output length of the* $\mathsf{NC}^0[c]$ *circuit.*

A 2-adaptive circuit over $\mathcal{C}$ is a collection of functions $(A, B_y) \in \mathcal{C}$, where y ranges over all possible output values of A. The value of the circuit on input x is $(A(x), B_{A(x)}(x))$.

Claim 3.5 *If (D_1, D_2) is ε-indistinguishable by* $\mathsf{AC}^0(2d+1, (2^m+1)(s+O(1)), 2m)$ *(resp.,* $\mathsf{AC}^0[B](2d+1, (2^m+1)(s+O(1)), 2m)$*), then it is ε-indistinguishable by all 2-adaptive circuits over* $\mathsf{AC}^0(d, s, m)$ *(resp.,* $\mathsf{AC}^0[B](d, s)$*).*

Proof Given a 2-adaptive distinguisher (A, B_y), let C be the circuit that on input x, outputs $(A(x), B_y(x))$ for $y = A(x)$. Then C has the desired depth and size, as y can be selected by a DNF or a CNF of size 2^m, and distinguishes at least as well as (A, B_y). ■

Claim 3.6 *If (D_0, D_1) is ε-indistinguishable by* $\mathsf{AC}^0(d, s, 1)$ *(resp., $AC^0[B](d, s, 1)$) then it is $3^m\varepsilon/2$-indistinguishable by* $\mathsf{AC}^0(d, 2s, m)$ *(resp., $AC^0[B](d+1, s, m)$).*

Proof Let C be the distinguishing circuit with m outputs $C_1, \dots, C_m$. For AC^0 circuits we may assume without loss of generality that all the topmost gates of $C_1, \dots, C_m$ are ANDs. Modifying the distinguisher to have this form only affects the size by a factor of two.

By closure, for all subsets S of $\{1, \dots, m\}$, the circuit $\mathsf{AND}_{i \in S} C_i$ distinguishes D_0 and D_1 with advantage at most ε. Any point function of $C_1, \dots, C_m$ has the form $(\mathsf{AND}_{i \in T} C_i)(\mathsf{AND}_{i \notin T} \overline{C_i})$ for some subset T of $[m]$ indicating the positive literals. Using the Möbius transform (i.e., inclusion-exclusion) the point function can be expanded as

$$\mathsf{AND}_{i \in T} C_i \ \mathsf{AND}_{i \notin T} \overline{C_i} = \mathsf{AND}_{i \in T} C_i \ (1 - \mathsf{OR}_{i \notin T} C_i) = \sum_{S \subseteq \overline{T}} (-1)^{|S|} \mathsf{AND}_{i \in T \cup S} C_i.$$

By linearity it follows that the distinguishing advantage of a point function with t positive literals can be at most $2^{m-t}\varepsilon$. As the statistical distance between $C(D_0)$ and $C(D_1)$ is half the sum of the distinguishing advantages of all point functions, it can be at most $\frac{1}{2} \sum \binom{m}{t} 2^{m-t} \alpha = 3^m \alpha / 2$. ■

We conclude with Håstad's unconditional result on indistinguishability of parity by constant-depth circuits.

Theorem 3.7 **([Hås14])** *For any $d, s \in \mathbb{N}$ there exists a constant c_d that depends only on d such that $(\mathsf{PAR}(n,0), \mathsf{PAR}(n,1))$ is $2^{-c_d n/(\log s)^{d-1}}$-indistinguishable by $\mathsf{AC}^0(d, s, 1)$*

Corollary 3.8 *There exists a constant c_d such that $(\mathsf{PAR}(n,0), \mathsf{PAR}(n,1))$ are $2^{-c_d n/(\log s)^{d-1}}$-indistinguishable by $\mathsf{AC}^0(d, s/2, c_d n/(\log s)^{d-1})$ and $2^{-n^{O(1/d)}}$-indistinguishable by 2-adaptive circuits over $\mathsf{AC}^0(d/2 - 1, 2^{n^{O(1/d)}}, n^{O(1/d)})$.*

3.3 Leakage Resilient Circuit Compilers

In this subsection, we give the definitions of leakage resilient circuit compiler (abbreviated as LRCC) for stateful and stateless circuits.

LRCC for Stateful Circuits. We first recall the notion of stateful circuits. This description is taken verbatim from [ISW03]. A stateful circuit is a circuit augmented with *memory cells*. A memory cell is a stateful gate with fan-in 1: on any invocation of the circuit, it outputs the previous input to the gate, and stores the current input for the next invocation. Thus, memory cells act as delay elements. We extend the usual definition of a circuit by allowing stateful circuits to possibly contain cycles, so long as every cycle traverses at least one memory cell. When specifying a stateful circuit, we must also specify an initial state for the memory cells. When C denotes a circuit with memory cells and s_0 an initial state for the memory cells, we write $C[s_0]$ for the circuit C with memory cells initially filled with s_0. Stateful circuits can also have external input and output wires. For instance, in an AES circuit the internal memory cells contain the secret key, the input wires a plaintext, and the output wires produce the corresponding ciphertext. The computation of $C[s]$ on an input x results in a wire assignment W (a wire assignment is a string that is obtained by concatenating the values carried by all the wires in C), the output y and an updated state s_1. In the rest of the paper, we sometimes denote $\mathsf{W}[C(s)(x)]$ to be assignment to all the wires in C when the value of the memory cells is s and it is run with input x.

Definition 3.9 **($(\mathcal{L}, \tau, \varepsilon)$-leakage resilient implementation)** *Let C be a deterministic stateful circuit, $\mathcal{L}$ be a leakage class, τ be a round parameter and ε be an error parameter. We say that $(\widehat{C}, \mathsf{Setup})$ is an $(\mathcal{L}, \tau, \varepsilon)$-leakage resilient implementation of C if:*

- *$\widehat{C}$ is a randomized, stateful circuit.*
- Setup *is a randomized mapping from the initial state s_0 of C to an initial state $\widehat{s}_0$ of $\widehat{C}$.*
- ***Correctness.*** *For every $k \in \mathbb{N}$ and every sequence of inputs $x_1, \ldots, x_k$, we require that probability (over the random coins of* Setup *and $\widehat{C}$) that the same outputs are obtained by (stateful) invocations of $C[s_0]$ and $\widehat{C}[\widehat{s}_0]$ on this input sequence is 1.*
- ***Security.*** *For every (possibly unbounded) stateful adversary $\mathcal{A}$, there exists a (stateful) simulator $\mathcal{S}$ such that for every initial state s_0 :*

$$\left| \Pr[\mathsf{Real}_{\mathcal{A},\widehat{C},\mathsf{Setup},\mathcal{L}}(s_0, \tau) = 1] - \Pr[\mathsf{Ideal}_{\mathcal{A},\widehat{C},\mathsf{Setup},\mathcal{S},\mathcal{L}}(s_0, \tau) = 1] \right| \leq \varepsilon$$

where Real *and* Ideal *experiments are defined in Figure 1.*

```
$\mathsf{Real}_{\mathcal{A},\widehat{C},\mathsf{Setup},\mathcal{L}}(s_0, \tau)$

1. $\widehat{s}_0 \leftarrow \mathsf{Setup}(s_0)$.
2. Set $y_0, z_0 = \perp$.
3. for every round $t$ from 1 to $\tau$:
     • $x_t, \ell_t \leftarrow \mathcal{A}(\widehat{C}, y_{t-1}, z_{t-1})$ where $\ell_t \in \mathcal{L}$.
     • $(\widehat{\mathsf{W}}_t, y_t, \widehat{s}_t) \Leftarrow \widehat{C}[\widehat{s}_{t-1}](x_t)$.
     • $z_t = \ell_t(\widehat{\mathsf{W}}_t)$.
4. Output whatever $\mathcal{A}$ outputs.
```

```
$\mathsf{Ideal}_{\mathcal{A},\widehat{C},\mathsf{Setup},\mathcal{S},\mathcal{L}}(s_0, \tau)$

1. Set $y_0, z_0 = \perp$.
2. for every round $t$ from 1 to $\tau$:
     • $x_t, \ell_t \leftarrow \mathcal{A}(\widehat{C}, y_{t-1}, z_{t-1})$ where $\ell_t \in \mathcal{L}$.
     • $(\mathsf{W}_t, y_t, s_t) \Leftarrow C[s_{t-1}](x_t)$
     • $z_t = \ell_t(\mathcal{S}(C, x_t, y_t))$.
3. Output whatever $\mathcal{A}$ outputs.
```

Figure 1: Real and Ideal Experiments

Definition 3.10 (LRCC for Stateful Circuits) *Let n be the security parameter. A leakage resilient stateful circuit compiler for the (stateful) circuit class $\mathcal{C}$ is a pair of polynomial-time algorithms* $(\mathsf{Tr}, \mathsf{St})$ *such that:*

- Tr *is a deterministic algorithm that maps a deterministic stateful circuit in $C \in \mathcal{C}$ and the security parameter 1^n to another stateful, randomized circuit $\widehat{C}$.*
- St *is a randomized algorithm that maps an initial state s_0 of C and the security parameter 1^n to an initial state $\widehat{s}_0$ of $\widehat{C}$.*

For a leakage class $\mathcal{L}(n)$, round parameter $\tau(n)$ and error parameter $\varepsilon(n)$, we say that $(\mathsf{Tr}, \mathsf{St})$ *is a $(\mathcal{L}(n), \tau(n), \varepsilon(n))$-leakage resilient circuit compiler for $\mathcal{C}$, if for every stateful circuit $C \in \mathcal{C}$,* $(\mathsf{Tr}(C, 1^n), \mathsf{St}(\star, 1^n))$ *is a $(\mathcal{L}(n), \tau(n), \varepsilon(n))$-leakage resilient implementation of C.*

LRCC for Stateless Circuits. We now define a leakage-resilient circuit compiler for stateless circuits.

Definition 3.11 (($\mathcal{L}, \varepsilon$)-leakage resilient implementation) *Let $C : \{0,1\}^k \to \{0,1\}^m$ be a deterministic stateless circuit, $\mathcal{L}$ be a leakage class, and ε be an error parameter. We say that $(I, \widehat{C}, O)$ is a $(\mathcal{L}, \varepsilon)$-leakage resilient implementation of C if:*

- *$I : \{0,1\}^k \to \{0,1\}^{\widehat{k}}$ is a randomized input encoder which maps an input x to an encoded input $\widehat{x}$.*
- *$\widehat{C}$ is a randomized circuit that maps an encoded input $\widehat{x}$ to an encoded output $\widehat{y} \in \{0,1\}^{\widehat{m}}$.*
- *$O : \{0,1\}^{\widehat{m}} \to \{0,1\}^m$ is the deterministic output decoder that maps an encoded output $\widehat{y}$ to y.*
- ***Correctness:** For every input $x \in \{0,1\}^k$, $\Pr[O(\widehat{C}(I(x))) = f(x)] = 1$ where the probability is over the random coins of I and $\widehat{C}$.*
- ***Security:** For any two inputs $x_0, x_1 \in \{0,1\}^k$, let $(\mathsf{W}_0, \widehat{y}_0) \Leftarrow \widehat{C}(I(x_0))$ and $(\mathsf{W}_1, \widehat{y}_1) \Leftarrow \widehat{C}(I(x_1))$ where W_0 (resp. W_1) represents the assignment to every wire of $\widehat{C}$ on input $I(x_0)$ (resp. $I(x_1)$). For any leakage function $\ell \in \mathcal{L}$, the statistical distance between $\ell(\mathsf{W}_0)$ and $\ell(\mathsf{W}_1)$ is at most ε.*

Definition 3.12 (LRCC for Stateless Circuits) *Let n be the security parameter and let $\mathcal{C}$ be a class of stateless circuits taking k input bits and having m output bits. A leakage resilient stateless circuit compiler for the class $\mathcal{C}$ is a tuple of polynomial-time algorithms $(\mathsf{Enc}, \mathsf{Tr}, \mathsf{Dec})$ where*

- *Enc is a randomized input encoder which maps an input $x \in \{0,1\}^k$ and the security parameter 1^n to an encoded input $\widehat{x}$.*
- *Tr is a deterministic algorithm that maps a deterministic stateless circuit in $C \in \mathcal{C}$ and the security parameter 1^n to another stateful, randomized circuit $\widehat{C}$. $\widehat{C}$ maps an encoded input $\widehat{x}$ to an encoded output $\widehat{y}$.*
- *Dec is the deterministic output decoder that maps an encoded output $\widehat{y}$ to $y \in \{0,1\}^m$.*

For a leakage class $\mathcal{L}(n)$ and the error parameter $\varepsilon(n)$, we say that $(\mathsf{Enc}, \mathsf{Tr}, \mathsf{Dec})$ is a $(\mathcal{L}(n), \varepsilon(n))$-leakage resilient circuit compiler for $\mathcal{C}$ if for every $C \in \mathcal{C}$, $(\mathsf{Enc}(\star, 1^n), \mathsf{Tr}(C, 1^n), \mathsf{Dec})$ is a $(\mathcal{L}(n), \varepsilon(n))$-leakage resilient implementation of C.

4 Improved Analysis of the ISW Construction

The leakage-resilient circuit transformer of Ishai, Sahai, and Wagner [ISW03] is shown in Figure 2. Ishai et al. proved it is correct and perfectly secure against leakage functions that depend on at most $n/2 - 1$ wires.

The transformer maintains the invariant that every wire w of C is represented by a wire bundle $\mathbf{w}$ that XORs to the bit value w, ensuring correctness; for details of the correctness proof see [ISW03].

The input encoder $\mathsf{Enc}(1^n, x)$: Every input bit $x_i \in \{0,1\}$ is encoded independently by $\mathbf{x}_i \in \{0,1\}^n$ which is random conditioned on its parity being equal to x_i.

The transformer $\mathsf{Tr}(1^n, C)$:

Every wire $w \in \{0,1\}$ of C is replaced by a wire bundle $\mathbf{w} \in \{0,1\}^n$.

Every addition gate $a + b$ in C is implemented by $\mathbf{a} + \mathbf{b}$, where $\mathbf{a}, \mathbf{b}$ are the wire bundles representing a, b, respectively.

Every multiplication gate $a \times b$ is implemented as follows. Compute the matrix $\mathbf{Z} \in \{0,1\}^{n\times n}$ given by

$$Z_{ij} = \begin{cases} \text{a random bit}, & \text{if } i < j \\ a_i b_j, & \text{if } i = j \\ Z_{ji} + a_i b_j + a_j b_i, & \text{if } i > j \end{cases}$$

and output the matrix-vector product $\mathbf{Z} \cdot \mathbf{1}$ computed from left to right.

Every copy gate taking a wire a as input in C is implemented by copying each bit of $\mathbf{a}$.

The output decoder $\mathsf{Dec}(1^n, \mathbf{y}_1 \cdots \mathbf{y}_m)$: Replace every encoded output wire bundle $\mathbf{y}_j$ by its parity $y_{j1} + \cdots + y_{jn}$.

Figure 2: The Ishai-Sahai-Wagner circuit compiler [ISW03].

Theorem 4.1 *Let $\mathcal{C}$ be any class of functions that is closed under restriction and negation of inputs. Assume $(\mathsf{PAR}(n,0), \mathsf{PAR}(n,1))$ is ε-indistinguishable by $\mathcal{C} \circ \mathsf{NC}^0[2]$. Then the ISW circuit compiler is $(\mathcal{C}, k\varepsilon)$-leakage resilient stateless compiler where k is the input size of the circuit.*

Let $\widehat{C}(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ represent the transformed circuit when it is given wire bundles $\mathbf{x}_1, \ldots, \mathbf{x}_k$ as its inputs. The following lemma is key to the proof of Theorem 4.1.

Lemma 4.2 *For every circuit C of size S on k inputs, every k strings $\mathbf{w}_1, \ldots, \mathbf{w}_k \in \{0,1\}^n$, and every k bits $c_1, \ldots, c_k$, there exists an $\mathsf{NC}^0[2]$ circuit that takes as input $\mathbf{x}$ and outputs a distribution that is identical to $\widehat{C}(\mathbf{w}_1 + c_1 \cdot \mathbf{x}, \ldots, \mathbf{w}_k + c_k \cdot \mathbf{x})$.*

Proof of Theorem 4.1: Fix a leakage function $\ell \in \mathcal{C}$. We need to show that $\ell(\mathsf{W})$ is statistically close to $\ell(\mathsf{W}')$ where W and W' are the wires of $\widehat{C}(\mathsf{Enc}(x))$ and $\widehat{C}(\mathsf{Enc}(x'))$ for any $x, x' \in \{0,1\}^k$. First consider the case when x and x' differ in a single bit, say the i-th bit. Hardwiring all encoded inputs except for $\mathbf{x}_i$ into $\widehat{C}$ and applying Lemma 4.2 with $\mathbf{w}_j = \mathbf{x}_j, c_j = 0$ for $j \neq i$, and $\mathbf{w}_i = \mathbf{0}, c_i = 1, \mathbf{x} = \mathbf{x}_i$, we infer there exists an $\mathsf{NC}^0[2]$ circuit that generates the distribution of $\widehat{C}(\mathsf{Enc}(x))$ or $\widehat{C}(\mathsf{Enc}(x'))$ (depending on the parity of $\mathbf{x}_i$). Thus, $\ell \circ \mathsf{NC}^0[2]$ can be used to distinguish the case where $\mathbf{x}_i$ has parity 0 and the case where $\mathbf{x}_i$ has parity 1 and thus, contradicting the assumption in Theorem 4.1.

For the general case, consider the hybrid wire distributions $\widehat{C}(\mathsf{Enc}(x^i))$, where $x^0 = x$, $x^k = x'$, and x^{i-1}, x^i differ in at most one bit. By what was just proved $\widehat{C}(\mathsf{Enc}(x^{i-1}))$ and $\widehat{C}(\mathsf{Enc}(x^i))$ are ε-

indistnguishable, so by the triangle inequality $\widehat{C}(\mathsf{Enc}(x))$ and $\widehat{C}(\mathsf{Enc}(x'))$ must be $k\varepsilon$-indistinguishable. ■

The main idea in the proof of Lemma 4.2 is the following claim, which states that the wire distribution of any single gate in the transformed circuit can be described locally, and moreover the output of the gate obeys the same type of distribution as its inputs.

Claim 4.3 *For all $g \in \{+, \times\}$ and $\mathbf{w}, \mathbf{w}', c, c'$ there exists a simulator Sim such that*

1. *The wires of $\mathsf{Sim}(\mathbf{x})$ and $\hat{g}(\mathbf{w} + c \cdot \mathbf{x}, \mathbf{w}' + c' \cdot \mathbf{x})$ are identically distributed even conditioned on $\mathbf{x}$.*
2. *The value $\mathbf{y}$ assigned to the output bundle by $\mathsf{Sim}(\mathbf{x})$ equals $\mathbf{w}'' + c'' \cdot \mathbf{x}$ for some $\mathbf{w}''$ and c'' that depend on the internal randomness of Sim only.*
3. *Every wire of $\mathsf{Sim}(\mathbf{x})$ depends on at most two bits of $\mathbf{x}$.*

Proof of Lemma 4.2: The proof is by induction on S which is the size of the circuit C. When $S = 0$, there are no internal gates so the $\mathsf{NC}^0[2]$ circuit has to generate $(\mathbf{w}_1 + c_1 \cdot \mathbf{x}, \ldots, \mathbf{w}_k + c_k \cdot \mathbf{x})$ and this can be trivially done.

Now suppose the lemma holds for all circuits of size $S - 1$. Given a circuit C of size S, let g be a bottom gate of C and x_i, x_j its (possibly identical) inputs. We need to show an $\mathsf{NC}^0[2]$ circuit that takes $\mathbf{x}$ as input and has to generate the wires of $\widehat{g}(\mathbf{w}_i + c_i \cdot \mathbf{x}, \mathbf{w}_j + c_j \cdot \mathbf{x})$, and the wires of $\widehat{C^-}(\mathbf{w}_1 + c_1 \cdot \mathbf{x}, \ldots, \mathbf{w}_k + c_k \cdot \mathbf{x}, \mathbf{y})$, where C^- is the circuit obtained by removing gate g from C and replacing its output by y.

By part 1 of Claim 4.3, the wires of $\widehat{g}(\mathbf{w}_i + c_i \cdot \mathbf{x}, \mathbf{w}_j + c_j \cdot \mathbf{x})$ can be replaced by those of $\mathsf{Sim}(\mathbf{x})$ without affecting the distinguisher's advantage. By part 3 they are 2-local functions of $\mathbf{x}$. Therefore, there is an $\mathsf{NC}^0[2]$ circuit that generates the wires of $\widehat{g}$. The Lemma now follows from part 2 of Claim 4.3 and the inductive hypothesis applied to the circuit C^-. ■

Proof of Claim 4.3: If g is an addition gate, set Sim to be the circuit that computes in parallel $\mathbf{w} + \mathbf{w}' + (c + c')\mathbf{x}$. The output is the sum of its two inputs confirming part 2, and there are no wires other than the output wires, from where part 3 follows.

Let $\mathbf{a} = \mathbf{w} + c\mathbf{x}$ and $\mathbf{b} = \mathbf{w}' + c'\mathbf{x}$. If g is a multiplication gate, the simulator $\mathsf{Sim}(\mathbf{x})$ works like $\widehat{\times}$, but uses the following alternative implementation of the matrix $\mathbf{Z}$:

$$Z_{ij} = \begin{cases} \text{a random bit} + a_i w'_j + b_i w_j, & \text{if } i < j \\ a_i b_j, & \text{if } i = j \\ Z_{ji} + a_i b_j + a_j b_i, & \text{if } i > j \end{cases}$$

This alternative implementation of $\mathbf{Z}$ does not affect the distribution of the entries of $\mathbf{Z}$ and therefore of the wires of the transformed circuit. We now argue properties 2 and 3 of Claim 4.3.

When $i = j$ and $i < j$, Z_{ij} only depends on the i-th bit of $\mathbf{a}$ and $\mathbf{b}$, which are independent of all but possibly the i-th bit of $\mathbf{x}$. When $i > j$, $Z_{ij} = Z_{ji} + a_i b_j + a_j b_i$ and this equals randomness plus the bit

$$(a_j w'_i + b_j w_i) + (a_i b_j + a_j b_i).$$

The first bracketed term equals $cx_jw'_i + c'x_jw_i$ plus a term that only depends on $\mathbf{w}$. The second one equals

$$(w_i + cx_i)(w'_j + c'x_j) + (w'_i + c'x_i)(w_j + cx_j) \\ = (cx_iw'_j + c'x_iw_j) + (cx_jw'_i + c'x_jw_i) + (w_iw'_j + w_jw_i).$$

Therefore the sum of the two equals $cx_iw'_j + c'x_iw_j$ plus a term that only depends on $\mathbf{w}$. It follows that for any i, j, Z_{ij} can only depend on the i-th bit of $\mathbf{x}$ and the wires in the computation of $Z_{i,j}$ for $i > j$ is a 2-local function of $\mathbf{x}$. We thus conclude that the wires in the computation of $\mathbf{Z} \cdot \mathbf{1}$ is a 1-local function in $\mathbf{x}$ and the output is of the form $\mathbf{w}'' + c'' \cdot \mathbf{x}$. ■

Corollary 1.1 follows directly from Theorem 4.1, Claim 3.4 and Corollary 3.8.

5 LRCC for Stateful Circuits

In this section, we give a construction of leakage resilient circuit compiler and prove its security against leakage classes that have low correlation with parity.

The class $\mathcal{C} \circ \mathsf{NC}^0[c]$ consists of all composed functions $f \circ g$ where $f \in \mathcal{C}$ and every output of g depends on at most c inputs.

Theorem 5.1 *Let c be a universal constant and $\mathcal{C}$ be any class of functions that is closed under restriction. If $(\mathsf{PAR}(n, 0), \mathsf{PAR}(n, 1))$ is ε-indistinguishable by 2-adaptive functions in $\mathcal{C} \circ \mathsf{NC}^0[c]$, then the construction in Figure 3 is a $(\mathcal{C}, T, O(\varepsilon T(S+n)))$-leakage resilient stateful circuit compiler for the class of stateful circuits of size S and T is the number of rounds.*

Organization. In Section 5.1, we will describe a building block that generates a random encoding of 0 and prove some useful properties. In Section 5.2, we give the description of the transformer $(\mathsf{Tr}, \mathsf{St})$. In Sections 5.3-5.5, we prove the security of the construction.

5.1 The Zero-Encoder

In this subsection, we describe and analyze a circuit $\mathsf{RandZero}$ that produces random encodings of the bit zero.

Construction 5.2 $\mathsf{RandZero}$*: On input matrix $\mathbf{G} \in \{0,1\}^{n \times n}$ and vector $\mathbf{r} \in \{0,1\}^n$, calculate*

$$Z_{ij} = \begin{cases} \textit{a random bit}, & \textit{if } i < j, \\ G_{ii}r_i, & \textit{if } i = j, \\ Z_{ji} + G_{ij}r_j + G_{ji}r_i, & \textit{if } i > j, \end{cases}$$

and output the matrix-vector product $\mathbf{Z} \cdot \mathbf{1}$ computed from left to right.

We denote by $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}; \mathbf{z})]$ the wire assignment of the circuit on input $\mathbf{G}, \mathbf{r}$ and internal randomness $\mathbf{z}$. The dependence on internal randomness is hidden when irrelevant.

For an n-by-m matrix $\mathbf{R}$ with columns $\mathbf{r}_1, \ldots, \mathbf{r}_m$, we write $\mathsf{RandZero}(\mathbf{G}, \mathbf{R})$ for the multi-output circuit $(\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_1; \mathbf{z}_1), \ldots, \mathsf{RandZero}(\mathbf{G}, \mathbf{r}_m; \mathbf{z}_m))$, where $\mathbf{z}_i$ is chosen uniformly and independently.

Basic properties The following facts can be inferred directly from the construction.

Fact 5.3 (Output distribution) *For every* $\mathbf{G}$ *and* $\mathbf{r}$*,* $\mathbf{c} = \mathsf{RandZero}(\mathbf{G}, \mathbf{r})$ *is uniformly random conditioned on* $\mathbf{1}^T \cdot \mathbf{c} = \mathbf{1}^T \cdot \mathbf{G} \cdot \mathbf{r}$*.*

Proof The equation is satisfied as both the left and right-hand sides are equal to the sum of the entries of $\mathbf{Z}$. We show that $\mathbf{c}$ is $(n-1)$-wise independent. If we remove the i-th row of $\mathbf{Z}$ and reorder the rest in sequence $1, 2, \ldots, i-1$ followed by $n, n-1, \ldots, i+1$, then each row contains a random bit that does not appear in any of the previous ones: These are the bits $Z_{12}, Z_{12}, \ldots, Z_{(i-1)i}$ followed by $Z_{n(n-1)}, \ldots, Z_{(i+1)i}$. Therefore $c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_n$ is a uniformly random bit sequence. ■

Fact 5.4 (Linearity) $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_1 + \mathbf{r}_2)]$ *is identically distributed to* $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_1; \mathbf{z}_1)] + \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_2; \mathbf{z}_2)]$ *provided at least one of* $\mathbf{z}_1$*,* $\mathbf{z}_2$ *is uniformly random.*

Proof $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}; \mathbf{z})]$ is a linear function of $\mathbf{r}$ and $\mathbf{z}$, so even when say $\mathbf{z}_1$ is fixed, $\mathbf{z} = \mathbf{z}_1 + \mathbf{z}_2$ is uniform. ■

Simulation The following claims provide simulations of the $\mathsf{RandZero}$ that are in a suitable sense "independent" of its respective inputs $\mathbf{G}$ and $\mathbf{r}$. In the following claims, we define $\mathsf{Diagonal}(r_1, \ldots, r_n)$ to be the diagonal $n \times n$ matrix where the i-th diagonal entry is given by r_i.

Claim 5.5 *There exists a simulator circuit* Simr *such that*

1. *For every* $\mathbf{G}$ *and* $\mathbf{r}$*,* $\mathsf{W}[\mathsf{Simr}(\mathbf{G}, \mathbf{r})]$ *and* $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r})]$ *are identically distributed.*[2]
2. *The output of* $\mathsf{Simr}(\mathbf{G}, \mathbf{r}; \mathbf{z})$ *equals* $\mathsf{Diagonal}(r_1, \ldots, r_n)\mathbf{G}^T\mathbf{1}$ *plus some function that depends only on* $\mathbf{z}$*.*
3. *For fixed* $\mathbf{G}$ *and* $\mathbf{z}$*,* $\mathsf{W}[\mathsf{Simr}(\mathbf{G}, \mathbf{r}; \mathbf{z})]$ *is an* NC^0 *function of* $\mathbf{r}$*.*

Claim 5.6 *There exists a simulator* Simv *such that*

1. *For every* $\mathbf{G}$*,* $\mathbf{r}$*, and* $\mathbf{v} \in \{0,1\}^n$*,* $\mathsf{W}[\mathsf{Simv}(\mathbf{G}, \mathbf{v}, \mathbf{r})]$ *and* $\mathsf{W}[\mathsf{RandZero}(\mathbf{G} + \mathbf{v} \cdot \mathbf{1}^T, \mathbf{r})]$ *are identically distributed.*
2. $\mathsf{Simv}(\mathbf{G}, \mathbf{v}, \mathbf{r})$ *equals* $\mathbf{v}\mathbf{r}^T\mathbf{1}$ *plus some function that does not depend on* $\mathbf{v}$*.*
3. *For fixed* $\mathbf{G}, \mathbf{r}, \mathbf{z}$*,* $\mathsf{W}[\mathsf{Simr}(\mathbf{G}, \mathbf{v}, \mathbf{r}; \mathbf{z})]$ *is an* NC^0 *function of* $\mathbf{v}$*.*

Proof of Claim 5.5: The only difference between Simr and $\mathsf{RandZero}$ is in the choice of Z_{ij} when $i < j$:

$$Z_{ij} = \begin{cases} \text{a random bit} + G_{ji}r_i, & \text{if } i < j, \\ G_{ii}r_i, & \text{if } i = j, \\ Z_{ji} + G_{ij}r_j + G_{ji}r_i, & \text{if } i > j, \end{cases}$$

[2]The simulator circuit Simr is the composition of $\mathsf{RandZero}$ and a preprocessing circuit. The irrelevant wires from preprocessing are discounted when comparing the two distributions.

This does not affect the wire distribution, confirming property 1. For property 2, it is direct from the redefinition of $\mathbf{Z}$ that $\mathbf{Z} \cdot \mathbf{1}$ now equals $\mathsf{Diagonal}(r_1, \ldots, r_n)\mathbf{G}^T\mathbf{1}$ plus some linear function in the randomness $\mathbf{z}$. For property 3, observe that all the wires in Simr are affine functions of at most two bits of $\mathbf{r}$. ■

Proof of Claim 5.6: Set $\mathbf{G}' = \mathbf{G} + \mathbf{v}\mathbf{1}^T$ and modify $\mathbf{Z}$ so that

$$Z_{ij} = \begin{cases} \text{a random bit} + G'_{ij} r_j, & \text{if } i < j, \\ G'_{ii} r_i, & \text{if } i = j, \\ Z_{ji} + G'_{ij} r_j + G'_{ji} r_i, & \text{if } i > j, \end{cases} .$$

The changes do not affect the wire distribution, confirming property 1. For property 2, Z_{ij} now equals $v_i r_j$ plus some affine function of the other inputs, so $\mathbf{Z} \cdot \mathbf{1} = \mathbf{v}\mathbf{r}^T\mathbf{1}$ as desired. As for property 3, observe that Z_{ij} does not depend on any entries of $\mathbf{v}$ except possibly v_i, so the wires of Simv are affine functions of at most two bits of $\mathbf{v}$. ■

5.2 Construction

We give the description of our leakage resilient circuit compiler $(\mathsf{Tr}, \mathsf{St})$ in Figure 3.

Correctness. The invariant maintained by the implementation is that the value of each wire w of C equals the parity of the wire bundle $\mathbf{w}$ in $\widehat{C}$ representing it. By construction this is true for the input wires and the state wires. In all applications of $\mathsf{RandZero}$, the parity of the output of $\mathsf{RandZero}$ equals zero by Fact 5.3. It follows that the output of addition has parity $\mathbf{1}^T\mathbf{a}+\mathbf{1}^T\mathbf{b} = \sum(a_i+b_i)$, the output of multplication has parity $\mathbf{1}^T\mathbf{a}^T\mathbf{b}\mathbf{1} = (\sum a_i)(\sum b_i)$, and the state wire updates, including those to $\mathbf{G}$, preserve parity. Finally, the output gates equal the parity of the corresponding wires, establishing correctness.

Security. We now prove the security part of Theorem 5.1. We will show that for every (possibly unbounded) stateful adversary $\mathcal{A}$, there exists a (stateful) simulator $\mathcal{S}$ such that for every initial state, the adversary's view in the real and ideal experiment described in Figure 1 are statistically close.

In Section 5.3, we give the description of our simulator. The security proof consists of two steps, following the structure in the works of Faust et al. [FRR+10] and Rothblum [Rot12] (a pictorial representation of the structure of the proof is given in Figure 4.). First, in Section 5.4, we describe a local *internal reconstruction procedure* that represents the adversary's view as a local (NC^0) function of an *external wire distribution.* This distribution contains explicit descriptions for all the wires in all evaluation rounds of $\widehat{C}$, as well as some additional information for the multiplication gates and state updates.

Then in Section 5.5, we gradually modify the components of the external wire distribution until the wire values in $\widehat{C}$ observed by the adversary become independent of the wires of C and so the adversary's view can be simulated, unless various circuits obtained by restricting inputs in the composition of the leakage and the internal reconstruction procedure can compute parity.

5.3 Description of the Simulator

We give the description of the simulator $\mathcal{S}$ in Figure 5.

The construction. Given a security parameter 1^n, a circuit C, and an initial state $s \in \{0,1\}^k$:

Initialization:

The encoded state consists of k wire bundles $\mathbf{s}$, where the i-th one is a random n-bit string of parity s_i. In addition the state contains an $n \times n$ matrix $\mathbf{G}$ that is random conditioned on $\mathbf{1}^T\mathbf{G} = \mathbf{0}^T$.

Every wire w of C is represented by an n-wire bundle $\mathbf{w}$ in the transformed circuit $\widehat{C} = \mathsf{Tr}(C, 1^n)$. In the following, the boldface vector notation will be used to denote n-bit strings.

Computation:

Every input gate x in C is implemented by the wire bundle $x \cdot \mathbf{e}_1$, where $\mathbf{e}_1 = (1, 0, \ldots, 0)$.

Every addition gate $a + b$ in C is implemented as $\mathbf{a} + \mathbf{b} + \mathsf{RandZero}(\mathbf{G}, \mathbf{r})$, where $\mathbf{a}, \mathbf{b}$ are the bundles representing a, b and $\mathbf{r}$ is a random string.

Every multiplication gate $a \times b$ in C is implemented as $(\mathbf{a} \cdot \mathbf{b}^T + \mathsf{RandZero}(\mathbf{G}, \mathbf{R})) \cdot \mathbf{1}$, where where $\mathbf{a}, \mathbf{b}$ are the bundles representing a, b, $\mathbf{R}$ is a random $n \times n$ matrix, and matrix-vector multiplication is implemented left-to-right.

Every copy gate taking a wire a as input in C is implemented by copying each bit of $\mathbf{a}$ in $\widehat{C}$.

For every output gate in C represented by wire bundle $\mathbf{out}$, compute $\mathbf{out'} = (\mathbf{out} + \mathsf{RandZero}(\mathbf{G}, \mathbf{r}))$ for a random $\mathbf{r}$ and decode the output as $\mathbf{1}^T \cdot \mathbf{out'}$.

State update:

Replace every bundle $\mathbf{s}_i$ of the state by $\mathbf{s}_i + \mathsf{RandZero}(\mathbf{G}, \mathbf{r}_i)$ for a random $\mathbf{r}_i$. Replace $\mathbf{G}$ by $\mathsf{RandZero}(\mathbf{G}, \mathbf{R})$ for a random n by n matrix $\mathbf{R}$.

Figure 3: LRCC $(\mathsf{Tr}, \mathsf{St})$ for stateful circuits.

5.4 External Data Sampler

The external data associated to a circuit wire (of C in a given round) consists of the wires of a copy of the circuit $\mathsf{RandZero}$. The external data associated to a gate consists of the external data of all its incident wires, plus some auxiliary data specific to the gate. We give the description of the external data sampler in Figure 6.

The external wire distribution denotes the induced distribution on the output of the external data sampler when it run by sampling $\mathbf{G}_1$ (which is the generator matrix of the first round) and for every round, sampling $\{\mathbf{r}_i\}, \mathbf{R}$ from some distribution.

Internal Reconstruction Procedures. We now prove the following lemmas.

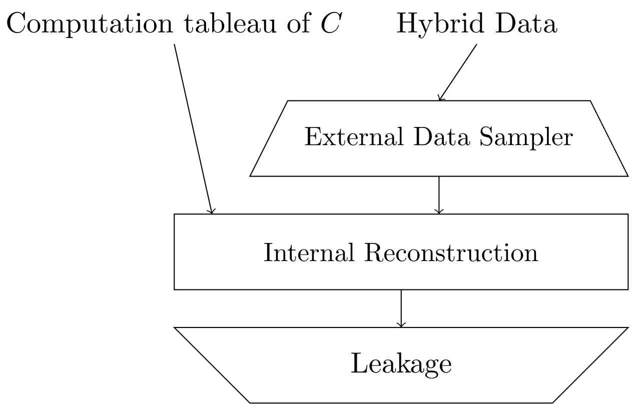

Figure 4: Components of the security proof. When the external sampler is given the input data for Hybrid_0, the adversary's view is identical to the output of the transformed circuit as in the real world. In Hybrid_3, the adversary's view is identical to the output of the simulator as in the ideal world. Indistinguishability of consecutive hybrids is argued by analyzing the view of the the leakage function composed with Internal Reconstruction.

Lemma 5.7 (Addition Reconstruction Procedure) *Fix a round and let* $\mathbf{G}$ *be the generator matrix for this round. There exists an* NC^0 *circuit* IR_+ *that, given inputs* $a, b \in \{0,1\}$ *and the external gate data* $\mathsf{W}_a = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_a; \mathbf{z}_a)]$, $\mathsf{W}_b = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_b; \mathbf{z}_b)]$, $\mathsf{W}_c = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_c; \mathbf{z}_c)]$ *outputs an assignment to the wires of a transformed addition gate* $\widehat{+}$ *such that if* $\mathbf{r}_c$ *and* $\mathbf{z}_c$ *are uniformly random,*

1. *The output of* IR_+ *is identically distributed to the wires of* $\widehat{+}((a\mathbf{e}_1 + \mathbf{o}_a), (b\mathbf{e}_1 + \mathbf{o}_b))$, *where* $\mathbf{o}_a$, $\mathbf{o}_b$ *are the outputs of* W_a, W_b, *respectively.*

2. *The output bundle of* IR_+ *is equal to* $\mathbf{o}_c + (a+b)\mathbf{e}_1$ *where* $\mathbf{o}_c$ *is the output of* W_c.

Proof The circuit IR_+ outputs the values $(a\mathbf{e}_1 + \mathbf{o}_a)$ and $(b\mathbf{e}_1 + \mathbf{o}_b)$ for the input wires $\mathbf{a}$ and $\mathbf{b}$ and $\mathsf{W}_a + \mathsf{W}_b + \mathsf{W}_c$ for the wires of the $\mathsf{RandZero}(\mathbf{G}, \mathbf{r})$ circuit used in the implementation of $\widehat{+}$, and obtains the output by adding $\mathbf{a}$, $\mathbf{b}$ and the output wires of $\mathsf{W}_a + \mathsf{W}_b + \mathsf{W}_c$. This can be computed in NC^0.

By Fact 5.4, $\mathsf{W}_a + \mathsf{W}_b + \mathsf{W}_c$ is identically distributed to $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r})]$ for a random $\mathbf{r}$, from where the identical distribution of the wires follows. ■

Note that updating the state can be expressed as special case of the addition circuit (i.e., setting one of the input vectors as $\mathbf{0}$). Hence, we get the following corollary.

Corollary 5.8 (State Update Reconstruction Procedure) *Fix a round and let* $\mathbf{G}$ *be the generator matrix for this round. There exists an* NC^0 *circuit* IR_{st} *that, given inputs* $a \in \{0,1\}$ *and the external gate data* $\mathsf{W}_a = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_a; \mathbf{z}_a)]$, $\mathsf{W}_c = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_c; \mathbf{z}_c)]$, *outputs an assignment to all the wires in the transformed state update gate* $\widehat{st}$ *such that if* $\mathbf{r}_c$ *and* $\mathbf{z}_c$ *are uniformly random,*

1. *The output of* IR_{st} *is identically distributed to the wires of* $\widehat{st}(a\mathbf{e}_1 + \mathbf{o}_a)$, *where* $\mathbf{o}_a$ *is the output in* W_a.

The (stateful) simulator. Given a security parameter 1^n, a circuit C, inputs x, and outputs y:

Initialization. The encoded state consists of k wire bundles $\mathbf{s}$, where the i-th one is uniformly random n-bit string. The matrix $\mathbf{G}$ is random conditioned on $\mathbf{1}^T\mathbf{G} = \mathbf{1}^T$.

The simulator runs the circuit $\widehat{C}$ using the input, addition, and multiplication gates implementation from Figure 3.

For every output gate of $\widehat{C}$ with input $\mathbf{x}$ whose value in y is *out*, the output of the gate $\widehat{C}$ is simulated as $\mathbf{1}^T(\mathbf{x} + \mathsf{RandZero}(\mathbf{G}, \mathbf{r}))$, where $\mathbf{r}$ is a random string of parity $out + \mathbf{1}^T\mathbf{x}$.

State update: The state update is the same as in 3, except that the matrix $\mathbf{R}$ used in $\mathbf{G}$'s update is random conditioned on $\mathbf{1}^T\mathbf{R} = \mathbf{1}^T$.

Figure 5: The simulator $\mathcal{S}$.

The External Data Sampler. The input to the Sampler consists of the round number t, a generator matrix for this round $\mathbf{G}_t$, a wire update seed $\mathbf{r}_{wt}$ for each wire bundle w that is the output of a gate, a state update seed matrix $\mathbf{R}_t$, the seeds $\mathbf{r}_{wt}$ and the wire bundle corresponding to the updated state from the previous round.

To sample the external wires for the round number t do the following:

1. For every input wire, sample the external data as $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{0}; \mathbf{0})]$.
2. For all other wires w which do not correspond to the updated state, sample the external data $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt})]$.
3. For wires w that correspond to the updated state, sample z' uniformly such that $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt}; z')]$ is equal to the updated state from the previous round.
4. For every multiplication gate, sample the auxiliary data $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{F})]$, where $\mathbf{F}$ is a uniformly random $n \times (n-1)$ matrix.
5. For the state update, sample $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{R}_t)]$ and update $\mathbf{G}_{t+1}$ to equal the output of this circuit.

Figure 6: External Data Sampler

2. *The output bundle of IR_{st} is equal to $\mathbf{o}_c + a\mathbf{e}_1$ where $\mathbf{o}_c$ is the output of W_c.*

Lemma 5.9 (Output Reconstruction Procedure) *Fix a round and let $\mathbf{G}$ be the generator matrix for this round. There exists an NC^0 circuit IR_{out} that, given inputs $a \in \{0,1\}$ and the external* *gate data* $\mathsf{W}_a = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_a; \mathbf{z}_a)]$, $\mathsf{W}_c = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_c; \mathbf{z}_c)]$, *outputs an assignment to all the wires except those in the final decoding step of a transformed output gate* $\widehat{\mathsf{out}}$ *such that:*

1. *If* $\mathbf{1}^T \cdot \mathbf{G} = \mathbf{0}^T$ *and* $\mathbf{r}_c$ *and* $\mathbf{z}_c$ *are uniformly random, the output of* IR_{out} *is identically distributed to these wires in* $\widehat{\mathsf{out}}(a\mathbf{e}_1 + \mathbf{o}_a)$, *where* $\mathbf{o}_a$ *is the output in* W_a.

2. *If* $\mathbf{1}^T \cdot \mathbf{G} = \mathbf{1}^T$ *and* $\mathbf{r}_c \sim \mathsf{PAR}(n, 0)$, $\mathbf{z}_c$ *is chosen uniformly random, the output of* IR_{out} *is identically distributed to these wires in the simulated distribution.*

Proof Consider the NC^0 circuit from Lemma 5.7 where we set $b = 0$ and $\mathbf{o}_b, \mathbf{r}_b$ and W_b to be all zeroes string. The first part of the lemma is a direct consequence of Lemma 5.7. To see the second part, note that the NC^0 circuit from Lemma 5.7 implicitly sets the randomness used in the gadget as $\mathbf{r} = \mathbf{r}_a + \mathbf{r}_c$. Thus, parity of $\mathbf{r}$ is equal to the parity of $\mathbf{G} \cdot (\mathbf{r}_a + \mathbf{r}_c)$ (since column parity of $\mathbf{G}$ is 1). This is equal to parity of $\mathbf{o}_a + \mathbf{o}_c$ (follows from Fact 5.3) which is in turn equal to the parity of $(a\mathbf{e}_1 + \mathbf{o}_a) + (a\mathbf{e}_1 + \mathbf{o}_c)$. Since $\mathbf{r}_c$ is chosen uniformly subject to its parity being 0, $\mathbf{r}$ is distributed uniformly subject to its parity being equal to the parity of $(a\mathbf{e}_1 + \mathbf{o}_a) + (a\mathbf{e}_1 + \mathbf{o}_c)$. This is precisely the simulated distribution. ■

Lemma 5.10 (Multiplication Reconstruction Procedure) *Fix a round and let* $\mathbf{G}$ *be the generator matrix for this round. There exists an* NC^0 *circuit* $IR_\times$ *that, given inputs* $a, b \in \{0, 1\}$ *and the external gate data* $\mathsf{W}_a = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_a; \mathbf{z}_a)]$, $\mathsf{W}_b = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_b; \mathbf{z}_b)]$, $\mathsf{W}_c = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_c; \mathbf{z}_c)]$, $\mathsf{W}_F = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{F}; z_F)]$ *outputs an assignment to the wires of a transformed multiplication gate* $\widehat{\times}$ *such that if* $\mathbf{r}_c, \mathbf{z}_c, \mathbf{F}, z_F$ *are uniformly random,*

1. *The output of* $IR_\times$ *is identically distributed to the wires of* $\widehat{\times}((a\mathbf{e}_1 + \mathbf{o}_a), (b\mathbf{e}_1 + \mathbf{o}_b))$, *where* $\mathbf{o}_a$, $\mathbf{o}_b$ *are the outputs of* W_a, W_b, *respectively.*

2. *The output bundle of* $IR_\times$ *is equal to* $\mathbf{o}_c + (ab)\mathbf{e}_1$ *where* $\mathbf{o}_c$ *is the output of* W_c.

Proof We start with the description of $IR_\times$. We set $\mathbf{a} = a\mathbf{e}_1 + \mathbf{o}_a$ and $\mathbf{b} = b\mathbf{e}_1 + \mathbf{o}_b$.

1. Compute in NC^0 the matrix $\mathbf{C} = (a\mathbf{e}_1 + \mathbf{o}_a) \cdot (b\mathbf{e}_1 + \mathbf{o}_b)^T$. The m-th column of this matrix is given by $\mathbf{c}_m$ and is equal to

$$\begin{aligned} \mathbf{c}_m &= b_m \cdot (\mathbf{o}_a + a \cdot \mathbf{e}_1) \text{ (where } b_m \text{ is the } m\text{-th entry of } \mathbf{b}) \\ &= b_m \cdot \mathbf{o}_a + b_m a \cdot \mathbf{e}_1 \\ &= \mathsf{RandZero}(\mathbf{G}, \mathbf{t}'_m; z'_m) + b_m a \cdot \mathbf{e}_1 \end{aligned}$$

where $(\mathbf{t}'_m, \mathbf{z}'_m) = \begin{cases} (\mathbf{r}_a, \mathbf{z}_a), & \text{if } b_m \neq 0, \\ (\mathbf{0}, \mathbf{0}), & \text{if } b_m = 0 \end{cases}$.

2. Let $(\mathbf{h}_1, \ldots, \mathbf{h}_{n-1}) = \mathsf{RandZero}(\mathbf{G}, \mathbf{F}; z_F)$. Compute a matrix $\mathbf{U} \in \{0, 1\}^{n \times n}$ in NC^0 defined as follows:

$$\mathbf{U} = (\mathbf{o}_c + a \cdot \mathbf{o}_b + \mathbf{h}_1, \mathbf{h}_1 + \mathbf{h}_2, \ldots, \mathbf{h}_{n-2} + \mathbf{h}_{n-1}, \mathbf{h}_{n-1}) + a \cdot \mathsf{Diagonal}(b_1, \ldots, b_n)$$

The sum of all the columns of this matrix is given by $\mathbf{o}_c + a \cdot \mathbf{o}_b + a \cdot (\mathbf{o}_b + b \cdot \mathbf{e}_0) = \mathbf{o}_c + ab\mathbf{e}_1$. Additionally, we note that we can compute $\mathbf{U} \cdot \mathbf{1}$ in NC^0. Now, observe that the m-th column of $\mathbf{U}$ given by $\mathbf{u}_m$ is of the form

$$\mathbf{u}_m = \mathsf{RandZero}(\mathbf{G}, \mathbf{t}_m; z_m) + ab_m \cdot \mathbf{e}_m$$

$$\text{where } (\mathbf{t}_m, z_m) = \begin{cases} (\mathbf{r}_c + a \cdot \mathbf{r}_b + \mathbf{f}_1, \mathbf{z}_c + a \cdot \mathbf{z}_b + \mathbf{z}_{F,1}), & \text{if } m = 1, \\ (\mathbf{f}_{m-1} + \mathbf{f}_m, \mathbf{z}_{F,m-1} + \mathbf{z}_{F,m}), & \text{if } 1 < m < n, \\ (\mathbf{f}_{n-1}, \mathbf{z}_{F,n-1}), & \text{if } m = n, \end{cases}$$

Note that $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{t}_m; z_m)]$ can be computed in NC^0.

3. Compute in NC^0, $\mathsf{W}_m = \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{t}'_m; z'_m)] + \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{t}_m; z_m)] + \mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{0}; z''_m)]$ where z''_m is an assignment such that $\mathsf{RandZero}(\mathbf{G}, \mathbf{0}; z''_m) = ab_m(\mathbf{e}_1 + \mathbf{e}_m)$.[3] Let $\mathbf{S}$ be the matrix whose m-th column is the output in W_m.

4. Output $\mathbf{C}$, the wire assignments in computing $\{\mathsf{W}_m\}$, $\mathbf{U} = \mathbf{S} + \mathbf{C}$ and the computation of $\mathbf{U} \cdot \mathbf{1}$.

Clearly, $IR_\times$ is computable in NC^0 since each of the above steps can be computed in NC^0 (and there are a constant number of steps). To show that the distribution output by this circuit is identical to the real distribution of the internal wires of the multiplication gate, it is sufficient to show that each column of $\mathbf{S}$ is distributed identically to the output of $\mathsf{RandZero}(\mathbf{G}, \mathbf{r}; z)$ for a randomly chosen $\mathbf{r}$ and z. We argue this as follows. Observe that $(\mathbf{r}_k, z_k)$ for $k \in \{a, b, c\}$ and $(\mathbf{F}, z_F)$ are chosen uniformly at random and hence from the linearity property (Lemma 5.4) that for every column m, the distribution of the wire assignments in $\mathsf{RandZero}(\mathbf{G}, \mathbf{t}_m)$ computed by the above circuit is identical to the distribution of $\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_m; z)$ for a uniformly chosen $\mathbf{r}_m$ and randomly chosen z. Specifically, for the first column this follows since $\mathbf{r}_c$ and $\mathbf{z}_c$ are uniformly distributed and for each of the other columns, it follows since $\mathbf{f}_i, \mathbf{z}_{F,i}$ are chosen uniformly at random. By one more application of the linearity property, we deduce that W_m is distributed identically to $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_m; z)]$ for an uniform $\mathbf{r}_m, z_m$. Since the m-th column of $\mathbf{S}$ is same as the output of W_m, it follows that this column is identically distributed to $\mathsf{RandZero}(\mathbf{G}, \mathbf{r}; z)$. ■

Lemma 5.11 (Composition) *There exists a circuit IR such that for every round, given the tableau of C and the external data for that round, outputs an assignment to all the wires of the transformed circuit except for those wires involved in the final output decoding such that:*

1. *(Locality) IR is in NC^0, and moreover every gate in the output of IR only depends on the tableau of the gate and on external data for its incident wires and the gate.*

2. *(Real world distribution) If $\mathbf{G}$ for the first round is sampled randomly such that $\mathbf{1}^T \cdot \mathbf{G} = \mathbf{0}^T$ and for every round, if the external data is generated by giving the sampler $\{\mathbf{r}_i\}, \mathbf{R}$ that are chosen uniformly at random, the concatenated outputs of IR in every round is identical to the real distribution of these wires.*

[3]Note that for any vector $\mathbf{v}$ whose parity is 0, there exists an assignment z such that $\mathsf{RandZero}(\mathbf{G}, \mathbf{0}; z) = \mathbf{v}$.

3. *(Ideal world distribution) If* $\mathbf{G}$ *for the first round is sampled randomly such that* $\mathbf{1}^T \cdot \mathbf{G} = \mathbf{1}^T$ *and for every round, if the external data is generated by giving the sampler* $\mathbf{r}_i \xleftarrow{\$} \{0,1\}^n$ *for every wire* i *that is not an output wire and for every output wire* i*,* $\mathbf{r}_i \sim \mathsf{PAR}(n, 0)$ *and* $\mathbf{R} \xleftarrow{\$} \{0,1\}^{n \times n}$ *subject to* $\mathbf{1}^T\mathbf{R} = \mathbf{1}^T$ *then the concatenated outputs of* IR *for every round is identical to the simulated distribution of these wires.*

Proof IR on inputs, the tableau and the the external data, simply invokes the corresponding internal reconstruction procedure for every gate (Lemmas 5.7, 5.10, 5.9 and Corollary 5.8), concatenates their wire assignment and outputs it. It is clear that IR is local in both senses specified.

To prove the second (resp. third) part, we consider a sequence of hybrids starting from the real (resp. simulated) distribution and ending with the output of IR.

- Hybrid_1 : In the first hybrid of this sequence, the only change that we make is that in the first round of the experiment, we sample initial state encoding $\mathbf{s}_i$ as $\mathsf{RandZero}(G, \mathbf{r}) + s_i\mathbf{e}_1$ where s_i is actual value of the input and $\mathbf{r}$ is uniformly chosen. It follows from Fact 5.3 that this hybrid is identical to the real (resp. simulated) distribution since $\mathbf{1}^T \cdot \mathbf{G} = \mathbf{0}^T$ (resp. $\mathbf{1}^T$). Specifically, if $\mathbf{1}^T \cdot \mathbf{G} = \mathbf{0}^T$ (resp. $\mathbf{1}^T$), then the output of $\mathsf{RandZero}(G, \mathbf{r})$ is identically distributed to a uniform string whose parity is 0 (resp. a random bit).

- $\mathsf{Hybrid}_{t,i}$: Now, in the (t, i)-th hybrid in this sequence, for every output wire w of a gate in the i-th layer of C in the t-th round, the external data corresponding to this wire is generated by running the sampler on an uniformly chosen $\mathbf{r}_w$ (resp. sampling $\mathbf{r}_w \sim \mathsf{PAR}(n, 0)$ if w is an output wire of the circuit). Then, the internal wires of every gate in the i-th layer is generated by running their corresponding internal reconstruction procedure. We note that it now follows as a direct consequence of Lemmas 5.7, 5.10, 5.9 and Corollary 5.8 that the i-th hybrid in this sequence is identical to the previous hybrid and this completes the argument.

■

5.5 Proof of Indistinguishability

In this subsection, we complete the proof of security. For this purpose we describe four hybrid distributions $\mathsf{Hybrid}_0, \mathsf{Hybrid}_1, \mathsf{Hybrid}_2, \mathsf{Hybrid}_3$ observed by the leakage. We argue that Hybrid_0 and Hybrid_3 are identically distributed to the wires of the transformed circuit and the simulator's output, respectively, and that all pairs of consecutive distributions are computationally indistinguishable by the leakage.

The four distributions are sampled by instantiating the external data sampler with different inputs, and then applying the internal reconstruction in Lemma 5.11 to the output. The inputs used to instantiate the external data sampler are:

Hybrid_0: Initial $\mathbf{G}$ is random conditioned on having zero column-parity ($\mathbf{1}^T\mathbf{G} = \mathbf{0}^T$). all wire update seeds $\mathbf{r}_{wt}$ and all state update seeds $\mathbf{R}_t$ are uniformly random.

Hybrid_1: $\mathbf{G}$ is sampled as in Hybrid_0. All wire update seeds $\mathbf{r}_{wt}$ and all state update seeds $\mathbf{R}_t$ are random conditioned on having column-parity 0 ($\mathbf{1}^T\mathbf{r}_{wt} = 0, \mathbf{1}^T\mathbf{R}_t = \mathbf{0}^T$).

Hybrid_2: $\mathbf{r}_{wt}$ are sampled as in Hybrid_1. $\mathbf{G}$ and $\mathbf{R}_t$ are random conditioned on having column-parity 1 ($\mathbf{1}^T\mathbf{G} = \mathbf{1}^T, \mathbf{1}^T\mathbf{R}_t = \mathbf{1}^T$).

Hybrid_3: $\mathbf{G}$ and $\mathbf{R}_t$ are sampled as in Hybrid_2. $\mathbf{r}_{wt}$ are uniformly random except for the output wires, which remain unchanged.

We note that the assignment to the final output decoding wires is a deterministic function of the external data. Thus, it follows from part 2 of Lemma 5.11, the view of the leakage function in Hybrid_0 is identical to the real distribution of the transformed circuit's wires, and by part 3, its view in Hybrid_3 is identical to the output of the simulator. To finish the proof, we establish the following three claims.

Claim 5.12 *Under the assumptions of Theorem 5.1, the adversary's outputs on* Hybrid_0 *and* Hybrid_1 *are $O(\varepsilon T(S+n))$-statistically close.*

Proof We fix $\mathbf{G} = \mathbf{G}_1$ and modify the distribution of the relevant seeds $\mathbf{r}_{wt}$ and the columns of $\mathbf{R}_t$ one by one, in increasing order of the round t. As the effect of both types of seeds is the same, without loss of generality, we analyze the effect of changing a seed of type $\mathbf{r}_{wt}$ from being uniformly random to having parity zero, assuming all the other seeds are fixed to maximize the adversary's distinguishing advantage.

We can simulate the first $(t-1)$ rounds of the leakage experiment using the fixed seeds. In the t-th round, we can generate all the external data for this round non-uniformly except $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt})]$. As all random seeds from the previous rounds have been fixed, by Fact 5.3 $\mathbf{G}_t$ is a fixed matrix with column-parity zero. We now argue via a sequence of hybrids where we switch from $\mathbf{r}_{wt}$ being a uniform random string to one where $\mathbf{r}_{wt}$ is uniform conditioned on its parity being 0.

1. $\mathsf{Hybrid}_{wt,1}$: This is identical to the distribution where $\mathbf{r}_{wt}$ is uniformly chosen. If w corresponds to the updated state wire in round t, then in the next round, we sample a uniform z' such that the output bundle in $\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$ is equal to the updated state from round t. Note that such a z' always exists since the column parity of $\mathbf{G}_t$ and $\mathbf{G}_{t+1}$ is 0.

2. $\mathsf{Hybrid}_{wt,2}$: We first consider a sub-hybrid where the external data $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt})]$ is replaced by $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt})]$. By part 1 of Claim 5.5, this change is indistinguishable to the adversary.

3. $\mathsf{Hybrid}_{wt,3}$: In this hybrid, if w corresponds to the state update wire then we sample z' uniformly and compute $\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$. We then sample a random z conditioned on the output bundle in $\mathsf{Simr}((\mathbf{G}_t, \mathbf{r}_{wt}); z)$ is equal to the output bundle in $\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$. This hybrid is identical to the previous hybrid.

4. $\mathsf{Hybrid}_{wt,4}$: In this hybrid, if w is an updated state wire, we replace $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt})]$ with $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt})]$. By part 1 of Claim 5.5, this change is indistinguishable to the adversary.

5. $\mathsf{Hybrid}_{wt,5}$: In this hybrid, if w corresponds to an updated state wire, then we sample uniform z and compute $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt}; z)$ and then sample a uniform z' such that the output bundle in $\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$ is identical to the output bundle in $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt}; z)$. This hybrid is again identically distributed to the previous hybrid.

6. $\mathsf{Hybrid}_{wt,2}$: In this hybrid, we switch from sampling $\mathbf{r}_{wt}$ as a uniform string to sampling it uniformly conditioned on its parity being 0. We now argue below why this change is indistinguishable to the adversary except with advantage ε.

To simulate the t-th round, we first observe that by part 3 of Claim 5.5, we infer that $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt})]$ is NC^0 computable from $\mathbf{r}_{wt}$ and therefore, we can generate all the external data for the t-th round by an NC^0 circuit. Now, running the internal reconstruction procedure IR (which is again an NC^0 circuit) on this external data outputs an assignment to every wire of $\widehat{C}$ in the t-th round except those in the final output decoding step. Since $\mathbf{G}_t^T\mathbf{1} = \mathbf{0}$, by part 2 of Claim 5.5, the output of $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt})$ is statistically independent of $\mathbf{r}_{wt}$. Therefore, the wires of all the gates in the computation that are evaluated after w (including the final output decoding in case that w is an output wire) are independent of $\mathbf{r}_{wt}$ and can be fixed to maximize the adversary's advantage. Thus, we can generate the wire assignment to every wire of $\widehat{C}$ in the t-th round using an NC^0 circuit. If w corresponds to the updated state wire, then we need to simulate the $(t+1)$-th round as well. However, since $\mathbf{G}_t^T \cdot \mathbf{1} = \mathbf{G}_{t+1}^T \cdot \mathbf{1} = \mathbf{0}$, it follows from part 2 of claim 5.5 that the updated state from the t-th round is given by some fixed function of the z. Thus, we can sample z' such that the output bundle of $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')]$ (which is independent of $\mathbf{r}_{wt}$, again from part 2 of claim 5.5) is equal to this fixed function of z. This allows us to generate the external data for $(t+1)$-th round using a NC^0 circuit and simulate the input to the leakage function in round $t+1$ via an NC^0 internal reconstruction procedures. The subsequent rounds of the leakage experiment can be simulated from the fixed seeds and thus, the bundles which feed into the subsequent rounds are independent of $\mathbf{r}_{wt}$ and depend only on the fixed seeds.

By the above argument, we deduced that (i) the first $(t-1)$ rounds of the leakage experiment can be simulated independent of $\mathbf{r}_{wt}$, (ii) the wire assignment in rounds t and $t+1$ are NC^0 computable from $\mathbf{r}_{wt}$, and (iii) the subsequent rounds of the experiment are independent of $\mathbf{r}_{wt}$. Therefore the adversary's advantage cannot exceed the maximal advantage of a 2-adaptive circuit from the class $\mathcal{C} \circ \mathsf{NC}^0$ in distinguishing a uniform random string from a parity-zero string. This is at most twice the advantage in distinguishing random parity-zero and parity-one strings, which is assumed to be ε.

By the triangle inequality, the adversary's advantage accumulated over all $O(T(S+n))$ changes is at most $O(\varepsilon T(S+n))$. ■

Claim 5.13 *Under the assumptions of Theorem 5.1, the adversary's outputs on* Hybrid_1 *and* Hybrid_2 *are $O(\varepsilon T)$-statistically close.*

Proof We modify the distribution on the matrices $\mathbf{G} = \mathbf{G}_1, \mathbf{R}_1, \ldots, \mathbf{R}_n$ one by one such that they are random subject to their column parity being 1.

Changing G. The change in the distribution of $\mathbf{G}$ can be implemented by setting $\mathbf{G} = \mathbf{G}' + \mathbf{v} \cdot \mathbf{1}^T$, where $\mathbf{G}'$ is a random column-parity zero matrix and $\mathbf{v}$ changes from a random parity-0 to a random parity-1 vector. As before, we consider a sequence of sub-hybrids where we will change $\mathbf{v}$ from being a random vector with parity 0 to a random vector with parity 1.

1. $\mathsf{Hybrid}_{1,1}$: This corresponds to the distribution where $\mathbf{v}$ is a random string with parity 0.

2. $\mathsf{Hybrid}_{1,2}$: In this hybrid, we replace all items of type $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{r}_{w1})]$, $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{F})]$, and $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}, \mathbf{R}_1)]$ in the external data for the first round by $\mathsf{W}[\mathsf{Simv}(\mathbf{G}', \mathbf{v}, \mathbf{r}_{\mathsf{w}1})]$, $\mathsf{W}[\mathsf{Simv}(\mathbf{G}', \mathbf{v}, \mathbf{F})]$, and $\mathsf{W}[\mathsf{Simv}(\mathbf{G}', \mathbf{v}, \mathbf{R}_1)]$. By part-1 of Claim 5.6, we infer that this is indistinguishable to the adversary.

3. $\mathsf{Hybrid}_{1,3}$: In this hybrid, we change $\mathbf{v}$ from a parity 0 vector to a random vector with parity 1. We argue below that the adversary cannot distinguish this change except with advantage ε.

Note that $\mathsf{W}[\mathsf{Simv}(\mathbf{G}', \mathbf{v}, \mathbf{r}_{\mathsf{w}1})]$, $\mathsf{W}[\mathsf{Simv}(\mathbf{G}', \mathbf{v}, \mathbf{F})]$, and $\mathsf{W}[\mathsf{Simv}(\mathbf{G}', \mathbf{v}, \mathbf{R}_1)]$ define the external data for the first round and by part 3 of claim 5.6, we can generate this by an NC^0 circuit. Now, applying the internal reconstructing procedure, IR (which is again an NC^0 circuit) from Lemma 5.11 on this external data allows us to generate all the wires in the computation of $\widehat{C}$ in the first round, except the assignment to the final output decoding wires. By part 2 of Claim 5.6, the output of $\mathsf{Simv}(\mathbf{G}', \mathbf{v}, \mathbf{r})$ is independent of $\mathbf{v}$ provided $\mathbf{r}$ has parity zero, which is true in all instantiations. Therefore all the wires in the final output decoding step of the first round are independent of $\mathbf{v}$ and can be non-uniformly computed. Thus, we have generated the assignment to every wire of $\widehat{C}$ in the first round by an NC^0 circuit. The subsequent rounds are independent of $\mathbf{v}$ as a direct consequence of part 2 of Claim 5.6. Thus, the assumption that random strings of parity zero and one are indistinguishable by $\mathcal{C} \circ \mathsf{NC}^0$, we obtain that the adversary's outputs when $\mathbf{G}$ is modified are ε-close.

Changing $\mathbf{R}_t$. We now analyze the change in advantage when $\mathbf{R}_t$ is modified from having column-parity zero to one. We represent $\mathbf{R}_t$ as $\mathbf{R}' + \mathbf{v} \cdot \mathbf{1}^T$, where $\mathbf{R}'$ s a random column-parity zero matrix and $\mathbf{v}$ changes from a random parity-0 to a random parity-1 vector. We fix all the random seeds given as input the external data sampler except for $\mathbf{v}$ such that adversary's distinguishing advantage is maximized conditioned on this fixing. This allows us to simulate the first $(t-1)$ rounds of the leakage experiment. As before, we consider sub-hybrids where we will replace $\mathbf{v}$ from a parity 0 string to a parity 1 string.

1. $\mathsf{Hybrid}_{t,1}$: This corresponds to the distribution where $\mathbf{v}$ is sampled uniformly at random subject to its parity is 0. For all the updated state wires in round t, we need to additionally generate the external date for this wire corresponding to round $t+1$. For every such wire w in round t, we sample a uniform z' such that the output bundle in the distribution $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt})]$ is equal to $\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$. Note that such a z' always exists since $1^T \mathbf{G}_t \cdot \mathbf{r}_{wt} = 1^T \cdot \mathbf{G}_{t+1} \cdot \mathbf{r}_{wt} = 1^T \cdot \mathbf{r}_{wt}$ from Fact 5.3.

2. $\mathsf{Hybrid}_{t,2}$: In this hybrid, we replace $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{R}_t)]$ in the external data with $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{R}_t)]$. By part 1 of Claim 5.5, this change is indistinguishable to the adversary.

3. $\mathsf{Hybrid}_{t,3}$: In this hybrid, for each wire w in round t that corresponds to the updated state, we sample a uniform z' and compute $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')]$ and sample a uniform z such that the output bundle in $\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt}; z)$ is equal to the output bundle in $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')]$. This hybrid is identical to the previous hybrid.

4. $\mathsf{Hybrid}_{t,4}$: In this hybrid, for each wire w in the updated state of round t, we replace $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt})]$ in the external data of the $(t+1)$-th round with $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt})]$. By part 1 of Claim 5.5, this change is indistinguishable to the adversary.

5. $\mathsf{Hybrid}_{t,5}$: In this hybrid, for each wire w in the updated state of round t, we again sample a uniform z and compute $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt}; z)]$ and sample a uniform z' such that the output bundle in $\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$ is equal to the output bundle in $\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt}; z)$. This hybrid is again identical to the previous hybrid.

6. $\mathsf{Hybrid}_{t,3}$: In this hybrid, we sample $\mathbf{v}$ as a uniform string with parity 1. We argue below that this change is indistinguishable to the adversary except with advantage ε.

Note that $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{R}_t)]$ defines the external data for the t-th round as well as the assignment to the final output decoding wires which are independent of $\mathbf{v}$ and hence can be non-uniformly fixed. As a consequence of part 3 of Claim 5.5 and part 1 of Lemma 5.11, we deduce that the assignment to all the wires of $\widehat{C}$ in the t-th round can be generated by an NC^0 circuit. Since the column parity of $\mathbf{G}_t$ is 1, by part 2 of Claim 5.5 $\mathbf{G}_{t+1} = \mathsf{RandZero}(\mathbf{G}_t, \mathbf{R}_t)$ can be expressed as $\mathbf{G}' + \mathbf{v} \cdot \mathbf{1}^T$ where $\mathbf{G}'$ is independent of $\mathbf{v}$. We may now use Claim 5.6 and Lemma 5.11 in an analogous manner to the first part of the proof to deduce that the assignment to all the wires of $\widehat{C}$ in the $(t+1)$-th round can be generated by an NC^0 circuit. Again, it follows from the part 2 of claim 5.6, the subsequent rounds of the leakage experiment can be simulated independent of $\mathbf{v}$.

We thus, conclude that the advantage of the adversary cannot exceed that of a 2-adaptive circuit in the class $C \circ \mathsf{NC}^0$ in distinguishing random strings $\mathbf{v}$ of parity zero and one. By assumption, this advantage is at most ε.

By the triangle inequality, the adversary's advantage accumulated by all T changes is at most εT. ■

Claim 5.14 *Under the assumptions of Theorem 5.1, the adversary's outputs on* Hybrid_2 *and* Hybrid_3 *are $O(\varepsilon TS)$-statistically close.*

Proof We fix $\mathbf{G}, \mathbf{R}_1, \ldots, \mathbf{R}_T$ and modify the distribution of $\mathbf{r}_{wt}$ one by one in the increasing order of round t if w is not an output wire. By a standard hybrid argument, it is sufficient to show that for some t, w, the effect of changing $\mathbf{r}_{wt}$ is ε-indistinguishable. We fix all other seeds except $\mathbf{r}_{wt}$ such that the adversary's distinguishing advantage is maximized conditioned on this fixing. As before, we consider a sequence of sub-hybrids where we will switch $\mathbf{r}_{wt}$ from a parity 0 string to a uniformly chosen random string.

1. $\mathsf{Hybrid}_{wt,1}$: This corresponds to the distribution where $\mathbf{r}_{wt}$ is chosen a random parity 0 string. If w is the updated state wire that feeds into the round $(t+1)$, then we need to additionally generate the external data corresponding to the round $(t+1)$ for this update wire. For this purpose, we sample a uniform z' such that the output bundle in the distribution $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt})]$ is equal to $\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$. Note that such a z' always exists since $1^T\mathbf{G}_t \cdot \mathbf{r}_{wt} = 1^T \cdot \mathbf{G}_{t+1} \cdot \mathbf{r}_{wt} = 1^T \cdot \mathbf{r}_{wt}$ from Fact 5.3.

2. $\mathsf{Hybrid}_{wt,2}$: In this hybrid, we replace $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_t, \mathbf{r}_{wt})]$ in the external data of the t-th round with $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt})]$. By part 1 of Claim 5.5, this change is indistinguishable to the adversary.

3. $\mathsf{Hybrid}_{wt,3}$: In this hybrid, we sample a uniform z' and compute $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')]$ and sample a z such that the output bundle in $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt}; z)$ is equal to the output bundle in $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')]$. This hybrid is identical to the previous hybrid.

4. $\mathsf{Hybrid}_{wt,4}$: In this hybrid, we replace $\mathsf{W}[\mathsf{RandZero}(\mathbf{G}_{t+1}, \mathbf{r}_{wt})]$ in the external data of the $(t+1)$-th round with $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt})]$. By part 1 of Claim 5.5, this change is indistinguishable to the adversary.

5. $\mathsf{Hybrid}_{wt,5}$: In this hybrid, we again sample a uniform z and compute $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt}; z)]$ and sample a uniform z' such that the output bundle in $\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z')$ is equal to the output bundle in $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt}; z)$.

6. $\mathsf{Hybrid}_{wt,6}$: In this hybrid, we sample $\mathbf{r}_{wt}$ uniformly at random. We now argue below that this change is indistinguishable to the adversary except with advantage ε.

We can simulate the first $(t-1)$ rounds of the leakage experiment using the fixed seeds. In the t-th round, we can generate all the external data for this round non-uniformly except $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt})]$. By part 3 of Claim 5.5, we infer that $\mathsf{W}[\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt})]$ is NC^0 computable from $\mathbf{r}_{wt}$ and therefore, we can generate in NC^0 all the external data for the t-th round along with the assignment to the final output decoding wires which are independent of $\mathbf{r}_{wt}$. Now, by running the internal reconstruction procedure IR from Lemma 5.11 on this external data, we can generate the assignment to every wire of $\widehat{C}$ in the t-th round using an NC^0 circuit. Since $\mathbf{G}_t^T\mathbf{1} = \mathbf{1}$, by part 2 of Claim 5.5, the output of $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt})$ is statistically independent of $\mathbf{G}_t$. In particular, the output bundle of $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt}; z) = \mathbf{r}_{wt} + f(z)$ for some fixed function f. Thus, we sample z' uniformly such that $f(z) = f(z')$. This ensures that the output bundles in both $\mathsf{Simr}(\mathbf{G}_t, \mathbf{r}_{wt}; z)$ and $\mathsf{Simr}(\mathbf{G}_{t+1}, \mathbf{r}_{wt}; z)$ are identical. Thus, even when w is an updated state wire, the external data for the $(t+1)$-th round can be generated by an NC^0 circuit (from part 3 of Claim 5.5). This can be used to obtain an assignment to every wire of $\widehat{C}$ in round $(t+1)$ by an NC^0 circuit (as a consequence of Lemma 5.11). The subsequent rounds of the leakage experiment can be simulated from the fixed seeds and are independent of $\mathbf{r}_{wt}$.

By the above argument, we deduced that (i) the first $(t-1)$ rounds of the leakage experiment can be simulated independent of $\mathbf{r}_{wt}$, (ii) the wire assignment in rounds t and $(t+1)$ are NC^0 computable from $\mathbf{r}_{wt}$, and (iii) the subsequent rounds of the experiment are independent of $\mathbf{r}_{wt}$. Therefore the adversary's advantage cannot exceed the advantage of 2-adaptive circuits from $\mathcal{C} \circ \mathsf{NC}^0$ in distinguishing a uniform random string from a parity-zero string. This is at most twice the advantage in distinguishing random parity-zero and parity-one strings, which is assumed to be ε.

■

From the above claims, we deduce that the real distribution is $O((S+n) \cdot \tau \cdot \varepsilon)$-close to the simulated distribution. This completes the proof of Theorem 5.1. Corollary 1.2 follows directly from Theorem 5.1, Claim 3.4 and Corollary 3.8.

Acknowledgements. We thank Mor Weiss for insightful discussions and the anonymous reviewers for helpful suggestions and comments. The first author's research is supported by Hong Kong RGC GRF CUHK14207618 and CUHK14209419. The second author's research is supported by ERC Project NTSC (742754), ISF grants 1709/14 and 2774/20, NSF-BSF grant 2015782, BSF grant 2018393, and a grant from the Ministry of Science and Technology, Israel and Department of Science and Technology, Government of India. The third author's research is supported in part from DARPA/ARL SAFEWARE Award W911NF15C0210, AFOSR Award FA9550-15-1-0274, AFOSR YIP Award, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for LongTerm Cybersecurity (CLTC, UC Berkeley). Work partially done when the first and the third authors were visiting the Technion.

References

[ABG$^+$14] Adi Akavia, Andrej Bogdanov, Siyao Guo, Akshay Kamath, and Alon Rosen. Candidate weak pseudorandom functions in AC^0 o MOD$_2$. In Moni Naor, editor, *ITCS 2014*, pages 251–260. ACM, January 2014.

[AIS18] Prabhanjan Ananth, Yuval Ishai, and Amit Sahai. Private circuits: A modular approach. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III*, pages 427–455, 2018.

[Ajt11] Miklós Ajtai. Secure computation with information leaking to an adversary. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 715–724, 2011.

[BBP$^+$16] Sonia Belaïd, Fabrice Benhamouda, Alain Passelègue, Emmanuel Prouff, Adrian Thillard, and Damien Vergnaud. Randomness complexity of private circuits for multiplication. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 616–648. Springer, Heidelberg, May 2016.

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

[BCPZ16] Alberto Battistello, Jean-Sébastien Coron, Emmanuel Prouff, and Rina Zeitoun. Horizontal side-channel attacks and countermeasures on the ISW masking scheme. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *CHES 2016*, volume 9813 of *LNCS*, pages 23–39. Springer, Heidelberg, August 2016.

[BDIR18] Fabrice Benhamouda, Akshay Degwekar, Yuval Ishai, and Tal Rabin. On the local leakage resilience of linear secret sharing schemes. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, pages 531–561, 2018.

[BDL14] Nir Bitansky, Dana Dachman-Soled, and Huijia Lin. Leakage-tolerant computation with input-independent preprocessing. In *CRYPTO*, pages 146–163, 2014.

[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*, pages 316–334, 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, 2012.

[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.

[BIS19] Andrej Bogdanov, Yuval Ishai, and Akshayaram Srinivasan. Unconditionally secure computation against low-complexity leakage. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 387–416. Springer, 2019.

[BIVW16] Andrej Bogdanov, Yuval Ishai, Emanuele Viola, and Christopher Williamson. Bounded indistinguishability and the complexity of recovering secrets. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 593–618. Springer, Heidelberg, August 2016.

[BKT19] Mark Bun, Robin Kothari, and Justin Thaler. Quantum algorithms and approximating polynomials for composed functions with shared inputs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 662–678, 2019.

[BMW$^+$18] Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F. Wenisch, Yuval Yarom, and Raoul Strackx. Foreshadow: Extracting the keys to the intel SGX kingdom with transient out-of-order execution. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018.*, pages 991–1008, 2018.

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

[Cor14] Jean-Sébastien Coron. Higher order masking of look-up tables. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 441–458. Springer, Heidelberg, May 2014.

[CPRR13] Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Thomas Roche. Higher-order side channel security and mask refreshing. In *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, pages 410–424, 2013.

[DDF14] Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying leakage models: From probing attacks to noisy leakage. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 423–440. Springer, Heidelberg, May 2014.

[DF12] Stefan Dziembowski and Sebastian Faust. Leakage-resilient circuits without computational assumptions. In *TCC 2012*, pages 230–247, 2012.

[DFS15] Stefan Dziembowski, Sebastian Faust, and Maciej Skorski. Noisy leakage revisited. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 159–188. Springer, Heidelberg, April 2015.

[DIK10] Ivan Damgård, Yuval Ishai, and Mikkel Krøigaard. Perfectly secure multiparty computation and the computational overhead of cryptography. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 445–465. Springer, Heidelberg, May 2010.

[DLZ15] Dana Dachman-Soled, Feng-Hao Liu, and Hong-Sheng Zhou. Leakage-resilient circuits revisited - optimal number of computing components without leak-free hardware. In *EUROCRYPT 2015*, pages 131–158, 2015.

[FIKK20] Yuval Filmus, Yuval Ishai, Avi Kaplan, and Guy Kindler. Limits of preprocessing. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 17:1–17:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[FPS17] Sebastian Faust, Clara Paglialonga, and Tobias Schneider. Amortizing randomness complexity in private circuits. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 781–810. Springer, Heidelberg, December 2017.

[FRR$^+$10] Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting circuits from leakage: the computationally-bounded and noisy cases. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 135–156. Springer, Heidelberg, May 2010.

[FRR$^+$14] Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting circuits from computationally bounded and noisy leakage. *SIAM J. Comput.*, 43(5):1564–1614, 2014. Extended abstract in Eurocrypt 2010.

[GIM$^+$16] Vipul Goyal, Yuval Ishai, Hemanta K. Maji, Amit Sahai, and Alexander A. Sherstov. Bounded-communication leakage resilience via parity-resilient circuits. In *FOCS 2016*, pages 1–10, 2016.

[GIW17] Daniel Genkin, Yuval Ishai, and Mor Weiss. How to construct a leakage-resilient (stateless) trusted party. In *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, pages 209–244, 2017.

[GJS11] Sanjam Garg, Abhishek Jain, and Amit Sahai. Leakage-resilient zero knowledge. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 297–315. Springer, Heidelberg, August 2011.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[GR10] Shafi Goldwasser and Guy N. Rothblum. Securing computation against continuous leakage. In *CRYPTO 2010*, pages 59–79, 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, 2012.

[Hås14] Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM J. Comput.*, 43(5):1699–1708, 2014.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 463–481. Springer, Heidelberg, August 2003.

[IWY16] Yuval Ishai, Mor Weiss, and Guang Yang. Making the best of a leaky situation: Zero-knowledge pcps from leakage-resilient circuits. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 3–32, 2016.

[JV10] Ali Juma and Yevgeniy Vahlis. Protecting cryptographic keys against continual leakage. In *CRYPTO 2010*, pages 41–58, 2010.

[KGG$^+$18] Paul Kocher, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. Spectre attacks: Exploiting speculative execution. *CoRR*, abs/1801.01203, 2018.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 388–397. Springer, Heidelberg, August 1999.

[Koc96] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 104–113. Springer, Heidelberg, August 1996.

[LSG$^+$18] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown: Reading kernel memory from user space. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018.*, pages 973–990, 2018.

[Mil14] Eric Miles. Iterated group products and leakage resilience against NC1. In Moni Naor, editor, *ITCS 2014*, pages 261–268. ACM, January 2014.

[MR04] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 278–296. Springer, Heidelberg, February 2004.

[MV13] Eric Miles and Emanuele Viola. Shielding circuits with groups. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 251–260. ACM Press, June 2013.

[Rot12] Guy N. Rothblum. How to compute under $\mathcal{AC}^0$ leakage without secure hardware. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 552–569. Springer, Heidelberg, August 2012.

[RP10] Matthieu Rivain and Emmanuel Prouff. Provably secure higher-order masking of AES. In Stefan Mangard and François-Xavier Standaert, editors, *CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 413–427. Springer, 2010.

[Wei19] Mor Weiss. Zero-knowledge PCPs from leakage-resilient circuits, revisited. In *CFAIL 2019*, 2019. Available at https://sites.google.com/site/morweissmor/.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.