Unifying Freedom and Separation for Tight Probing-Secure Composition

Sonia Belaïd[1], Gaëtan Cassiers[3], Matthieu Rivain[1], and Abdul Rahman Taleb[1,2]

[1] CryptoExperts, France
[2] Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
[3] TU Graz, Austria
{sonia.belaid,matthieu.rivain,abdul.taleb}@cryptoexperts.com
gaetan.cassiers@iaik.tugraz.at

Abstract. The masking countermeasure is often analyzed in the probing model. Proving the probing security of large circuits at high masking orders is achieved by composing gadgets that satisfy security definitions such as non-interference (NI), strong non-interference (SNI) or free SNI. The region probing model is a variant of the probing model, where the probing capabilities of the adversary scale with the number of regions in a masked circuit. This model is of interest as it allows better reductions to the more realistic noisy leakage model. The efficiency of composable region probing secure masking has been recently improved with the introduction of the input-output separation (IOS) definition.
In this paper, we first establish equivalences between the non-interference framework and the IOS formalism. We also generalize the security definitions to multiple-input gadgets and systematically show implications and separations between these notions. Then, we study which gadgets from the literature satisfy these. We give new security proofs for some well-known arbitrary-order gadgets, and also some automated proofs for fixed-order, special-case gadgets. To this end, we introduce a new automated formal verification algorithm that solves the open problem of verifying free SNI, which is not a purely simulation-based definition. Using the relationships between the security notions, we adapt this algorithm to further verify IOS. Finally, we look at composition theorems. In the probing model, we use the link between free SNI and the IOS formalism to generalize and improve the efficiency of the tight private circuit (Asiacrypt 2018) construction, also fixing a flaw in the original proof. In the region probing model, we relax the assumptions for IOS composition (TCHES 2021), which allows to save many refresh gadgets, hence improving the efficiency.

Keywords: Masking · Probing model · Region probing model · Non-interference · Input output separation · Tight private circuit

1 Introduction

The security of cryptographic algorithms is mainly studied in the *black-box* model, where an adversary is restricted to the knowledge of inputs and outputs to recover the secret key. However, the discovery of *side-channel attacks* in the late nineties [31] opened the door for a new field of security study. These attacks can break cryptosystems by exploiting the device's physical leakage (power consumption, electromagnetic emanations, ...) that executes the implementation.

Many countermeasures have since been studied to protect against this class of attacks. The *masking* countermeasure is the most widely deployed one. The concept was introduced in 1999 by Chari et al. [21] and by Goubin and Patarin [25], and suggests splitting a sensitive variable x of the implementation into n shares, such that an adversary must get information on all the shares to recover the secret. This can be easily achieved by generating $n-1$ shares uniformly at random $x_1, \ldots, x_{n-1}$ and computing the last share x_n so that $x = x_1 * \ldots * x_{n-1} * x_n$ according to some group law $*$. An implementation then manipulates the sharing $\vec{x} = (x_1, \ldots, x_n)$ instead of the secret itself. While being easy to manipulate for affine operations, which can be computed on each share separately (sharewise), it is not trivial to securely manipulate sharings for non-linear operations without recombining the secrets.

To theoretically evaluate the security of a circuit against side-channel attacks, Ishai, Sahai, and Wagner (ISW) [29] introduced the *t-probing leakage model.* A circuit is secure in the t-probing model if the exact values of any set of t intermediate variables do not reveal any information on the secrets. This definition is motivated by the increasing difficulty of recovering information on the combination of $t+1$ variables from the leakage traces that contain noisy functions of the manipulated data, as t grows. Furthermore, this model corresponds to the practically-relevant notion of masking security order [6]. Many works have since built and analyzed masking schemes with respect to their security in the probing model [35,22,10].

Meanwhile, probing security is only partially satisfactory. Indeed, there is a mismatch between the fixed number of probes and the amount of real-world leakage, which grows with the number of performed computations. For instance, the authors of [7] show that the repeated manipulation of identical values can be exploited to retrieve the secrets in the context of *horizontal side-channel attacks.* This led to the formalization of the *noisy leakage model* in [34] as a specialization of the *only computation leaks* model [33]. In this model, the adversary can retrieve a noisy function of each intermediate variable of the computation. Because of the inconvenience of the noisy model for security proofs, Duc, Dziembowski, and Faust [24] proposed a reduction from the noisy model to the t-probing model. Reducing to the t-region probing model, where the whole computation is split into regions and the adversary can chose t probes in each region, improves significantly the security of this bound since this model increases the number of probes with respect to the size of the circuit/region.

Even in the simple probing model, proving the security of large masked circuits is a challenging problem, due to the combinatorial number of possible sets of probes. The most common solution is to build circuits from smaller sub-circuits named gadgets that implement a simple computation on masked data. Then, these gadgets can be composed to implement more complex computations, and the security proof only needs to care about the properties of the gadgets and their composition. In their seminal work [29], ISW introduced such gadgets for a field multiplication and addition. These gadgets can be arbitrarily composed, but this scheme requires masking with $n = 2t + 1$ shares. For performance reasons, the follow-up works mostly focused on using the optimal number of shares $n = t + 1$. The first composition security notion which is known as *t-strong non-interference* (SNI) was introduced by Barthe, Belaïd, Dupressoir, Fouque, Grégoire, Strub and Zucchini in [5]. In a nutshell, t-SNI ensures that probes inside a gadget and on its output shares can be perfectly simulated with knowledge of a limited number of input shares of a gadget. It follows that one can simulate all the probes in a composition of t-SNI gadgets by knowing t shares of each input sharing, which is independent of the secret input values. The SNI gadgets for affine operations are however not very efficient, was solved by the t-PINI definition (at the expense of slightly less efficient multiplication) [19].

Another direction to improve efficiency is to drop the requirement of arbitrary composition: while direct application of the ISW construction with $n = t + 1$ is not secure [35], it can be fixed by adding refresh gadgets (which implement the identity function but re-randomize the sharings). This has instance been proposed in [5] with the `maskComp` tool that can compose weaker non-interferent (NI) gadgets by inserting SNI refresh gadgets (which can be derived from the ISW multiplication) in the circuit. Later, Belaïd, Goudarzi and Rivain [12], introduced tight private circuits (TPC), which is another variant of the ISW scheme with $n = t+1$ and additional refresh gadgets. The set of refresh gadgets inserted by their `tightProve` tool is tight in the sense that removing one of these gadgets is guaranteed to break t-probing security.

A stronger version of the probing model has been considered in the literature which is known as the *region probing model* [1]. In this model, the adversary is not limited to t probes on the circuit but gets t probes per gadget (or *region*) of the circuit. This model is relevant in practice as being closer to actual side-channel leakages (providing information on all the gadgets of an implementation) which is formally captured by a reduction to the noisy leakage model [34,24]. In a recent work [27], Goudarzi, Prest, Rivain and Vergnaud introduce the *input-output separation* (IOS) notion for simple composition in the region probing model. This notion acts as probing-composition scissors: the circuit is divided into regions separated by IOS gadgets whose probes can then be simulated separately.

A complementary line of research has been considering the optimization of the masked gadgets themselves. This culminated in the design of a $(n - 1)$-NI multiplication gadget with randomness usage $n^2/4 + \mathcal{O}(n)$ [9] (while the ISW multiplication is $(n - 1)$-SNI and has $n^2/2 + \mathcal{O}(n)$ randomness usage) and a $(n - 1)$-SNI refresh gadget with complexity $\mathcal{O}(n \log n)$ [8]. Besides these arbitrary-order gadgets, there have been many optimizations for low-order gadgets. Manually verifying the security properties of such small gadgets is tedious and error-prone [22], which naturally led to the development of automatic formal verification tools for these security properties [4,5,2,30,16,13].

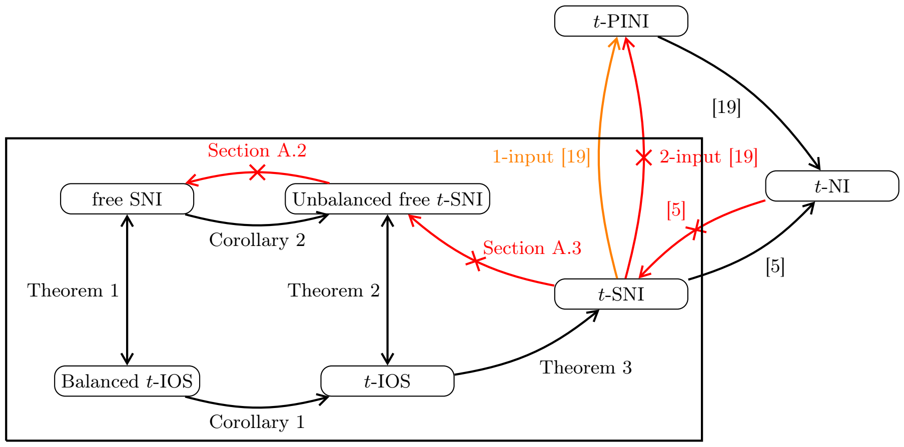

Fig. 1. Relations between security notions. Plain black arrows represent mathematical implications whereas red crossed out arrows mean that a counterexample demonstrates the absence of implication. The orange plain arrow represents an implication from t-SNI to t-PINI only for 1-input gadgets. The black rectangle frames the security notions that are focused on in this paper. F.V. means that a counterexample is given in the full version of the paper.

Our contributions. In this paper, we unify and extend existing probing composition notions, we analyze gadget constructions under these unified notions, we provide efficient verification methods for these notions, and we extend existing composition approaches in the probing and region probing model. In more details, our contributions are as follows:

- We show that the composition approach of tight private circuits (TPC) introduced in [12] actually requires a stronger notion than the SNI notion initially considered by the authors. The required notion happens to be the *free SNI* notion introduced by Coron and Spignoli in [23]. In a nutshell, the free SNI notion requires that the non-simulated output shares of a gadget be uniformly distributed and mutually independent of the simulated wires. We generalize the free SNI notion to two input gadgets and patch the composition proof of TPC.[4] Our proof also generalizes the TPC composition approach to any gadget circuit based on $(n-2)$-free SNI multiplication gadgets, $(n-1)$-free SNI refresh gadget and sharewise affine gadgets (which encompasses a wide number of masked circuits used in practice).
- We show strong connections between the free SNI notion and the IOS notion introduced in [27] for easy composition in the region probing model. Specif-

[4] We note that, while the composition proof is flawed, the TPC construction considered in [12] which relies on ISW multiplication and refresh gadgets is still secure since these gadgets achieve the necessary free SNI notion as we further show in the present paper.

ically, we show that these notions are essentially equivalent. More precisely, the IOS notion is equivalent to a *unbalanced* version of the free SNI notion which relaxes some constraints on the input and output shares involved in the simulation. On the other hand, the free SNI notion is equivalent to a *balanced* version of the IOS notion which adds the latter constraints to IOS. We further show that free SNI and IOS (for either the balanced or unbalanced version) both imply the SNI notion. This notably answers the questions left open in [28]: (1) IOS (resp. free SNI) is strictly stronger than SNI: IOS implies SNI while the converse is not true, (2) IOS (resp. free SNI) is strictly stronger than PINI for one-input gadget but it is separated from PINI for two-input gadgets. These relations are depicted on Figure 1 which summarizes the current state of affairs in terms of (most common) probing composition notions.

- We then investigate gadget constructions under the strong unified notions of free SNI and balanced IOS. We propose generic constructions of gadgets achieving these notions from simpler building blocks. We further demonstrate that common gadgets satisfy these properties. Specifically, we prove that the well known ISW multiplication gadget is free-$(n-2)$-SNI (or equivalently balanced $(n-2)$-IOS) and not free-$(n-1)$-SNI. We further show that the ISW refresh gadget as well as the quasilinear refresh gadget from [8] both satisfy the free-$(n-1)$-SNI security notion (or equivalently balanced $(n-1)$-IOS).
- While the verification of probing composition notions such as, *e.g.*, NI, SNI, PINI, is today relatively well understood and engineered [4,2,13], it is not yet clear how to verify notions such as free SNI or IOS which are different in nature. Verifying such notions not only means checking that a given tuple of wires can be perfectly simulated from at most t input shares, it further requires showing that a set of output wires is uniform and mutually independent of this simulation. In this paper, we tackle this issue and provide an efficient verification method for these notions. We present a set of algorithms to efficiently verify common gadgets under these notions which we implemented in the IronMask tool.[5] We further report applications of these algorithms to several gadgets of the literature and hence (in)validate their free SNI / IOS features.
- Finally, we extend the IOS composition framework proposed in [27]. We show a general composition theorem for circuits made of IOS gadgets and share-wise affine gadgets. Compared to the original IOS composition framework, we consider IOS gadgets which are not necessarily refresh gadgets, which allows us to take advantage of the IOS property of *e.g.* the ISW multiplication gadgets. Moreover, our composition result does not require to insert an IOS refresh gadget between any two non-IOS gadgets which allows to save many refresh gadgets, hence improving the global efficiency of the underlying masked circuit.

[5] This augmented version of IronMask is available at `https://github.com/CryptoExperts/IronMask`.

2 Preliminaries

Along the paper, $\mathbb{K}$ shall denote a finite field. For any tuple $\vec{x} = (x_1, \ldots, x_n) \in \mathbb{K}^n$ and any set $I \subseteq [1:n]$, we shall denote $\vec{x}|_I = (x_i)_{i \in I}$. We use $\overset{\text{id}}{=}$ to refer to the equality of the distributions of random variables.

2.1 Additive Sharing, Circuits and Gadgets

In the following, the *n-additive decoding* mapping, denoted AddDec, refers to the function $\bigcup_n \mathbb{K}^n \to \mathbb{K}$ defined as

$$\mathsf{AddDec} : (x_1, \ldots, x_n) \mapsto x_1 + \cdots + x_n \ ,$$

for every $n \in \mathbb{N}$ and $(x_1, \ldots, x_n) \in \mathbb{K}^n$. We shall further consider that, for every $n, \ell \in \mathbb{N}$, on input $(\vec{x}_1, \ldots, \vec{x}_\ell) \in (\mathbb{K}^n)^\ell$ the n-additive decoding mapping acts as

$$\mathsf{AddDec} : (\vec{x}_1, \ldots, \vec{x}_\ell) \mapsto (\mathsf{AddDec}(\vec{x}_1), \ldots, \mathsf{AddDec}(\vec{x}_\ell)) \ .$$

Definition 1 (Additive Sharing). *Let $n, \ell \in \mathbb{N}$. For any $x \in \mathbb{K}$, an n-*additive sharing *of x is a random vector $\vec{x} \in \mathbb{K}^n$ such that $\mathsf{AddDec}(\vec{x}) = x$. It is said to be* uniform *if for any set $I \subseteq [1:n]$ with $|I| < n$ the tuple $\vec{x}|_I$ is uniformly distributed over $\mathbb{K}^{|I|}$. A n-*additive encoding *is a probabilistic algorithm AddEnc which on input a tuple $(x_1, \ldots, x_\ell) \in \mathbb{K}^\ell$ outputs a tuple $(\vec{x}_1, \ldots, \vec{x}_\ell) \in (\mathbb{K}^n)^\ell$ such that $\vec{x}_i$ is a uniform n-additive sharing of x_i for every $i \in [1:\ell]$.*

An *arithmetic circuit* on a field $\mathbb{K}$ is a labeled directed acyclic graph whose edges are *wires* and vertices are *arithmetic gates* processing operations on $\mathbb{K}$. We consider circuits composed of addition gates, $(x_1, x_2) \mapsto x_1 + x_2$, multiplication gates, $(x_1, x_2) \mapsto x_1 \cdot x_2$, and copy gates, $x \mapsto (x, x)$. A *randomized arithmetic circuit* is equipped with an arithmetic circuit additional random gate which outputs a fresh uniform random value of $\mathbb{K}$.

In the following, we shall call an *(n-share, ℓ-to-m) gadget*, a randomized arithmetic circuit that maps an input $(\vec{x_1}, \ldots, \vec{x_\ell}) \in (\mathbb{K}^n)^\ell$ to an output $(\vec{y_1}, \ldots, \vec{y_m}) \in (\mathbb{K}^n)^m$ such that $(x_1, \ldots, x_\ell) = \mathsf{AddDec}(\vec{x_1}, \ldots, \vec{x_\ell}) \in \mathbb{K}^\ell$ and $(y_1, \ldots, y_m) = \mathsf{AddDec}(\vec{y_1}, \ldots, \vec{y_m}) \in \mathbb{K}^m$ satisfy $(y_1, \ldots, y_m) = g(x_1, \ldots, x_\ell)$ for some function g. A *refresh gadget* is a gadget for which g is the identity function, while a *multiplication gadget* implements the multiplication function. Affine functions g can be implemented with *sharewise affine gadgets*: such gadgets apply the underlying linear function to all the shares, except for the last one where the affine function is applied.

Some gadgets are said *probing complete*: for all the combinations of one share from each of their input sharings they contain a wire which depends on all the shares of the combination. For example, a multiplication gadget that computes $a_i \cdot b_j$ (if $\vec{a}$ and $\vec{b}$ are its input sharings) for all i and j is probing complete. For single-input gadgets, probing completeness is a trivial notion.

2.2 Probing Security

An *assign-wires sampler* takes as input a randomized arithmetic circuit C, a set of wire labels W (subset of the wire labels of C), and an input $(\vec{x_1}, \dots, \vec{x_\ell})$, and it outputs a $|W|$-tuple $(w_1, \dots, w_{|W|}) \in (\mathbb{K} \cup \{\perp\})^{|W|}$, denoted as

$$(w_1, \dots, w_{|W|}) \leftarrow \mathsf{AssignWires}(C, W, (\vec{x_1}, \dots, \vec{x_\ell})) \ ,$$

where $(w_1, \dots, w_{|W|})$ corresponds to the assignments of the wires of C with label in W for an evaluation on input $(\vec{x_1}, \dots, \vec{x_\ell})$.

Definition 2 (Probing Security). *A randomized arithmetic circuit C equipped with an encoding* Enc[6] *is t-probing secure if there exists a simulator* Sim *which, for any input $(x_1, \dots, x_\ell) \in \mathbb{K}^\ell$, for every set of wires W such that $|W| \leq t$, satisfies*

$$\mathsf{Sim}(C, W) \overset{\mathrm{id}}{=} \mathsf{AssignWires}(C, W, \mathsf{Enc}(x_1, \dots, x_\ell)).$$

Definition 3 (Region Probing Security). *Let r be an integer. A randomized arithmetic circuit C equipped with an encoding* Enc *is r-region probing secure if there exists a circuit partition $C = (C_1, \dots, C_m)$ such that for any input $(x_1, \dots, x_\ell) \in \mathbb{K}^\ell$ and for any sets of wires $W_1 \subseteq W_{C_1}$, $\dots$, $W_m \subseteq W_{C_m}$ such that $|W_1| \leq \lceil r|C_1| \rceil$, $\dots$, $|W_m| \leq \lceil r|C_m| \rceil$, there exists a simulator* Sim *which satisfies*

$$\mathsf{Sim}(C, W) \overset{\mathrm{id}}{=} \mathsf{AssignWires}(C, W, \mathsf{Enc}(x_1, \dots, x_\ell))$$

where $W = W_1 \cup \cdots \cup W_m$.

3 Advanced Probing Composition Notions

Several security notions have been introduced in the literature to efficiently compose gadgets in the (region) probing model. This section aims to recall them, to specify or relax them for our purposes, and to demonstrate how they are connected to each other.

3.1 Existing Notions

First, most common gadgets that use randomness are expected to be uniform in the sense of Definition 4 (to not be confused with the uniformity definition of Threshold Implementations [?]), which is the most basic requirement for a refresh gadget.

Definition 4 (Uniformity from [27]). *An (n-share, ℓ-to-m) gadget G implementing a function g is* uniform *if, for every $(\vec{x_1}, \dots, \vec{x_\ell}) \in (\mathbb{K}^n)^\ell$, the output $G(\vec{x_1}, \dots, \vec{x_\ell})$ is a uniform additive sharing (seeDefinition 1) of $g(x_1, \dots, x_\ell)$.*

[6] In this paper, we restrict ourselves to additive encodings as recalled in Definition 1.

Then, we recall the notions of strong non-interference (or SNI), input-output separative (or IOS) and free SNI that were chronologically introduced in various contexts to compose gadgets in the (region) probing model.

SNI was the first security notion introduced to securely compose n-share gadgets into a $(n-1)$-probing secure circuit. In a nutshell, a gadget is t-SNI if any set of t_i internal probes and t_o output probes can be perfectly simulated from at most t_i shares of each input sharing for any $t_i + t_o \leq t$. Composition of SNI gadgets is then straightforward. Probes of one gadget can be simulated by its input shares which are the output shares of the incoming gadget(s). The latter inherited probes can then be simulated *for free*.

Definition 5 (Strong Non-Interference [5]). *Let G be an (n-share, ℓ-to-1) gadget. G is said t-Strong Non-Interferent (t-SNI), if for every set W of internal wires of G such that $|W| \leq t_i$, and every set $J \subseteq [1:n]$ of output share indices such that $|J| \leq t_o$ and $t_i + t_o \leq t$, there exists a (two-stage) simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that for every input $(\vec{x_1}, \ldots, \vec{x_\ell}) \in (\mathbb{K}^n)^\ell$,*

1. $\mathsf{Sim}_1(W, J) = (I_1, \ldots, I_\ell)$ *where* $I_1, \ldots, I_\ell \subseteq [1:n]$, *with* $|I_1|, \ldots, |I_\ell| \leq |W|$,
2. $\mathsf{Sim}_2(W, J, (\vec{x_1}|_{I_1}, \ldots, \vec{x_\ell}|_{I_\ell})) \overset{\text{id}}{=} \big(\mathsf{AssignWires}(G, W, (\vec{x_1}, \ldots, \vec{x_\ell})), \vec{y}|_J\big)$

where $\vec{y} = G(\vec{x_1}, \ldots, \vec{x_\ell})$. A gadget is simply said to be SNI *if it is* $(n-1)$-SNI.

A few years after SNI, the IOS security notion was introduced for 1-to-1 refresh gadgets. The latter are meant to be inserted between gadgets satisfying the classical probing security notion (which does not yield a secure composition on its own) in order to obtain a region probing secure circuit. In a nutshell, a gadget is t-IOS if it is uniform and if any set of t probes can be perfectly simulated from at most t input shares and t output shares.

In the following, we shall say that a pair of vector $(\vec{x}, \vec{y}) \in (\mathbb{K}^n)^2$ is *admissible* for a gadget G if there exists a random tape $\vec{\rho}$ (*i.e.* an assignment of the random gates' outputs) such that $\vec{y} = G^{\vec{\rho}}(\vec{x})$[7]. For an admissible pair $(\vec{x}, \vec{y})$ and a set W of wires of G, the wire assignment distribution of G in W *induced* by $(\vec{x}, \vec{y})$, denoted $\mathsf{AssignWires}(G, W, \vec{x}, \vec{y}) \in \mathbb{K}^{|W|}$, is the random vector $\mathsf{AssignWires}(G, W, \vec{x})$ constrained to $\vec{y} = G^{\vec{\rho}}(\vec{x})$, *i.e.* the wire assignment distribution obtained for a uniform drawing of $\vec{\rho}$ among $\{\vec{\rho}; G_W^{\vec{\rho}}(\vec{x}) = \vec{y}\}$. We note that for a uniform gadget, an admissible pair is any $(\vec{x}, \vec{y}) \in (\mathbb{K}^n)^2$ such that $\sum_{i=1}^n x_i = \sum_{i=1}^n y_i$. Based on this definition, we recall the IOS security notion for any 1-to-1 gadget.

Definition 6 (IOS [27]). *Let G be an (n-share, 1-to-1) gadget. G is said t-*input-output separative (t-IOS)*, if it is uniform and if there exists a (two-stage) simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that for every admissible pair $(\vec{x}, \vec{y})$ for G and for every set of wires W of G with $|W| \leq t$, we have*

1. $\mathsf{Sim}_1(W) = (I, J)$ *where* $I, J \subseteq [1:n]$, *with* $|I| \leq |W|$ *and* $|J| \leq |W|$, *and*

[7] This notion of *admissible pair* can be trivially extended to the notion of *admissible tuple* for any number of inputs.

2. $\mathsf{Sim}_2(W, \vec{x}|_I, \vec{y}|_J) \overset{\text{id}}{=} \mathsf{AssignWires}(G, W, \vec{x}, \vec{y})$.

A gadget is simply said to be IOS *if it is n-*IOS*.*

Finally, the free SNI notion was introduced for 1-to-1 refresh gadgets as well to strengthen the existing SNI property and produce probing secure circuits in the stateful model. In addition to the SNI features, the free SNI security notion ensures that a strict subset of the output shares which are not indexed by the input shares involved in the probes' simulation is uniformly and independently distributed, even conditioned on the probes and the other output shares.

Definition 7 (Free SNI [23]). *Let G be an (n-share, 1-to-1) gadget. G is said* free t-SNI*, if for every set W of internal wires of G such that $|W| \leq t$, there exists a (two-stage) simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that for every input $\vec{x} \in \mathbb{K}^n$,*

1. $\mathsf{Sim}_1(W) = I$ *where* $I \subseteq [1 : n]$*, with* $|I| \leq |W|$*,*
2. $\mathsf{Sim}_2(W, \vec{x}|_I) \overset{\text{id}}{=} (\mathsf{AssignWires}(G, W, \vec{x}), \vec{y}|_I)$*,*

where $\vec{y} = G(\vec{x})$, and for every set $O \subsetneq [1 : n] \setminus I$, $\vec{y}|_O$ is uniformly and independently distributed, conditioned on $\mathsf{AssignWires}(G, W, \vec{x})$ and $\vec{y}|_I$. A gadget is simply said to be free SNI *if it is* free $(n-1)$-SNI.

3.2 Extending and Balancing IOS

We now generalize the IOS property for 2-input gadgets[8]. In a nutshell, any set of at most t probes is now simulated from *two* subsets of input shares and a subset of output shares.

Definition 8 (Two-Input IOS). *Let G be an (n-share, 2-to-1) gadget. G is said* t-input-output separative (t-IOS)*, if it is uniform and if there exists a (two-stage) simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that for every admissible triple $(\vec{x_1}, \vec{x_2}, \vec{y})$ for G and for every set of wires W of G with $|W| \leq t$, we have*

1. $\mathsf{Sim}_1(W) = (I_1, I_2, J)$ *where* $I_1, I_2, J \subseteq [1 : n]$*, with* $|I_1| \leq |W|$*,* $|I_2| \leq |W|$ *and* $|J| \leq |W|$*, and*
2. $\mathsf{Sim}_2(W, \vec{x_1}|_{I_1}, \vec{x_2}|_{I_2}, \vec{y}|_J) \overset{\text{id}}{=} \mathsf{AssignWires}(G, W, (\vec{x_1}, \vec{x_2}), \vec{y})$*.*

We then define a balanced version of the IOS property in which the output set of shares used for the simulation is entirely defined together with the input sets of shares used for the simulation. This tweaked notion is already satisfied by the refresh gadget introduced in [27] and is advantageously equivalent to the free SNI notion (as proven later in Theorem 1).

Definition 9 (Two-Input Balanced IOS). *Let G be an (n-share, 2-to-1) gadget. G is said* balanced t-input-output separative (balanced t-IOS)*, if it is uniform and if there exists a (two-stage) simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that for every admissible triple $(\vec{x_1}, \vec{x_2}, \vec{y})$ for G and for every set of wires W of G with $|W| \leq t$, we have*

[8] The definition for ℓ-input gadgets is given in Section A.1.

1. $\mathsf{Sim}_1(W) = (I_1, I_2)$ *where* $I_1, I_2 \subseteq [1:n]$, *with* $|I_1| \leq |W|$, $|I_2| \leq |W|$ *and*
2. $\mathsf{Sim}_2(W, \vec{x_1}|_{I_1}, \vec{x_2}|_{I_2}, \vec{y}|_{I_1 \cap I_2}) \overset{\text{id}}{=} \mathsf{AssignWires}(G, W, (\vec{x_1}, \vec{x_2}), \vec{y})$.

A gadget is simply said to be balanced IOS *if it is* balanced $(n-1)$-IOS.

Corollary 1. *An (n-share,* 2*-to-*1*) balanced t-IOS gadget is t-IOS.*

For the above corollary, it is indeed enough to use the first simulator of the balanced IOS definition and to fix the J set to the intersection of I_1 and I_2.

The (balanced) IOS notion can also be naturally extended to 1-to-2 gadgets, *e.g.* to cover copy gadgets which are useful in the context of region probing composition (see Section 7). We give a formal definition of IOS for 1-to-2 gadgets in Appendix B.2.

3.3 Extending and Unbalancing Free SNI

As for the IOS property, we generalize the free SNI security notion for 2-input gadgets[9]. In a nutshell, *two* subsets of input shares are now involved in the simulation and the output shares to be simulated are defined from the intersection of their indices.

Definition 10 (Two-Input Free SNI). *Let G be an (n-share,* 2*-to-*1*) gadget. G is said* free t-SNI, *if for every set W of internal wires of G such that $|W| \leq t$, there exists a (two-stage) simulator* $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that for every input* $(\vec{x_1}, \vec{x_2}) \in \mathbb{K}^n \times \mathbb{K}^n$,

1. $\mathsf{Sim}_1(W) = (I_1, I_2)$ *where* $I_1, I_2 \subseteq [1:n]$, *with* $|I_1| \leq |W|$ *and* $|I_2| \leq |W|$,
2. $\mathsf{Sim}_2(W, \vec{x_1}|_{I_1}, \vec{x_2}|_{I_2}) \overset{\text{id}}{=} \big(\mathsf{AssignWires}(G, W, (\vec{x_1}, \vec{x_2})), \vec{y}|_{I_1 \cap I_2}\big)$,

where $\vec{y} = G(\vec{x_1}, \vec{x_2})$, *and for every set* $O \subsetneq [1:n] \setminus (I_1 \cap I_2)$, $\vec{y}|_O$ *is uniformly and independently distributed, conditioned on* $\mathsf{AssignWires}(G, W, (\vec{x_1}, \vec{x_2}))$ *and* $\vec{y}|_{I_1 \cap I_2}$. *A gadget is simply said to be* free SNI *if it is* free $(n-1)$-SNI.

In the reverse direction compared to IOS (see Figure 1), we define an unbalanced variant of free SNI for which the first simulator outputs different sets of indices. It has the advantage of being equivalent to the IOS security property (as proven later in Theorem 2).

Definition 11 (Two-Input Unbalanced Free SNI). *Let G be an (n-share,* 2*-to-*1*) gadget. G is said* unbalanced free t-SNI, *if for every set W of internal wires of G such that $|W| \leq t$, there exists a (two-stage) simulator* $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that for every input* $(\vec{x_1}, \vec{x_2}) \in \mathbb{K}^n \times \mathbb{K}^n$,

1. $\mathsf{Sim}_1(W) = (I_1, I_2, J)$ *where* $I_1, I_2, J \subseteq [1:n]$, *with* $|I_1| \leq |W|$, $|I_2| \leq |W|$ *and* $|J| \leq |W|$, *and*
2. $\mathsf{Sim}_2(W, \vec{x_1}|_{I_1}, \vec{x_2}|_{I_2}) \overset{\text{id}}{=} \big(\mathsf{AssignWires}(G, W, (\vec{x_1}, \vec{x_2})), \vec{y}|_J\big)$,

[9] The definition for ℓ-input gadgets is given in Appendix A.1..

where $\vec{y} = G(\vec{x_1}, \vec{x_2})$, and for every set $O \subsetneq [1:n] \setminus J$, $\vec{y}|_O$ is uniformly and independently distributed, conditioned on $\mathsf{AssignWires}(G, W, (\vec{x_1}, \vec{x_2}))$ *and $\vec{y}|_J$. A gadget is simply said to be* unbalanced free SNI *if it is* unbalanced free $(n-1)$-SNI.

Corollary 2. *An (n-share, 2-to-1) free t-SNI gadget is unbalanced free t-SNI.*

As for IOS (see Corollary 1), the free t-SNI property trivially implies the unbalanced free t-SNI property by fixing the output set of indices J to the intersection of sets I_1 and I_2.

Also, unbalanced free t-SNI implies t-SNI: the unbalanced free-SNI can simulate any strict subset of the output (more formally, this is a consequence of Theorem 3).

Corollary 3. *An (n-share, 2-to-1) unbalanced free t-SNI gadget is t-SNI.*

3.4 Relations between Security Notions

In this section, we draw and prove the relations between the different security notions recalled or introduced above for (n-share, 2-to-1) gadgets. They are summarized in Figure 1.

We first demonstrate the equivalence between the free t-SNI and balanced t-IOS notions. This result is somehow surprising given the different natures of the two notions: Free SNI requires the ability to simulate the probed wires from some input shares with the feature that non-simulated output wires are mutually independent of the simulation. On the other hand, IOS requires the ability to simulate the probed wires from some input *and* output shares where the simulation must be consistent with any given (admissible) values of the input and output sharings. The following theorem show that these two requirements are essentially equivalent.

Theorem 1. *Let G be a (n-share, 1-to-1 or 2-to-1) gadget and let t be an integer strictly less than n. G is free t-SNI if and only if G is t-balanced IOS.*

The complete proof is given in Section A.4. For both implications, we demonstrate how to build the new simulators from the existing ones. For the right-to-left implication, we additionally show that some part of the output sharing is uniform and independent conditioned on the probes and the remaining outputs using a contradiction. While the first simulators of both properties are always built the same way, the construction of the second simulators is trickier. For the left-to-right implication, we need to run the second free t-SNI simulator until its second output matches the third value of the admissible triple for the balanced IOS property. For the right-to-left implication, any uniformly random chunk of output forms an admissible triple to be used in the second balanced t-IOS simulator to build the free t-SNI one.

Similarly, Theorem 2 gives the equivalence between the unbalanced free SNI and IOS notions. Its proof follows exactly the same steps as the proof of Theorem 1 (see complete proof in Section A.4).

Theorem 2. *Let G be a (n-share, 1-to-1 or 2-to-1) gadget and let t be an integer strictly less than n. G is unbalanced free t-SNI if and only if G is t-IOS.*

Finally, the t-IOS security property implies the t-SNI security property, as stated in Theorem 3 and formally proven in Section A.5. In a nutshell, the first t-SNI simulator is built from the two first outputs of the first t-IOS simulator. Then, the second t-SNI simulator is built from the second t-IOS simulator using an admissible triple in which the output additionally integrates the output probes authorized by the SNI property.

Theorem 3. *Let G be an (n-share, 1-to-1 or 2-to-1) gadget and let t be an integer strictly less than n. If G is t-IOS then G is t-SNI.*

From Theorem 1 on the equivalence between t-IOS and unbalanced free t-SNI, it follows that unbalanced free t-SNI also directly implies t-SNI.

Meanwhile, unbalanced free t-SNI (or equivalently t-IOS) does not imply free t-SNI (or equivalently balanced t-IOS) and t-SNI does not imply t-unbalanced SNI (or equivalently t-IOS). Counterexamples are given in Section A.2 and Section A.3.

3.5 Additional Relations for ZE-Refresh Gadgets

In the literature, it can be noticed that most efficient refresh gadgets are built from a refresh on a sharing of zero. We thus properly define such constructions and we show how to use them to build free SNI gadgets.

In the following, we call a *ZE-refresh gadget* (*ZE* for *zero-encoding*) a gadget G which, for any input $\vec{x} \in \mathbb{K}^n$ computes $\vec{y} = G(\vec{x})$ as follows (*i.e.*, respecting the order of operations)

$$\vec{y} \leftarrow \vec{x} + \mathsf{ZeroEnc}()\ ,$$

where ZeroEnc is a randomized circuit which takes no input and outputs a sharing of 0. We note that G is uniform if and only if ZeroEnc outputs a uniform sharing of 0, *i.e.* $\vec{z} \leftarrow \mathsf{ZeroEnc}()$ is such that $\vec{z}|_O$ is uniformly distributed over $\mathbb{K}^{|O|}$ for every $O \subsetneq [1:n]$. Definition 12 introduces a security property on ZeroEnc that, if satisfied, yields free t-SNI ZE-refresh gadgets.

Definition 12 (Free Encoding of Zero). *Let* ZeroEnc *be an (n-share, 0-to-1) gadget performing a refresh on a sharing of zero.* ZeroEnc *is said to be t-free if for every set W of (internal and output) wires on* ZeroEnc *with $|W| \leq t$, there exists a set J of cardinality $|J| \leq |W|$ such that for every set $O \subsetneq [1:n] \setminus J$, $\vec{z}|_O$ is uniformly distributed and mutually independent of* $\mathsf{AssignWires}(\mathsf{ZeroEnc}, W, \emptyset, \vec{z})$, *and $\vec{z}|_J$, where $\vec{z} = \mathsf{ZeroEnc}()$.*

Then we obtain the following result, whose proof is given in Section A.6.

Proposition 1. *Let G be an n-share uniform ZE-refresh gadget. The inner gadget* ZeroEnc *is t-free if and only if G is free t-SNI.*

Let us remark that a result similar to Proposition 1 for SNI refresh gadgets has been introduced by Cassiers et al. [18]. Indeed, our ZE-refresh gadgets are the same construction as their "off-path" refresh gadgets. Moreover, they define the notion of Strong Output Independence (t-SOI) for $\mathsf{ZeroEnc}$ gadgets, which is a weaker variant of free encoding: $\mathsf{ZeroEnc}$ is t-SOI if, for every $t_1 + t_2 \leq t$, every set of wires in $\mathsf{ZeroEnc}$ with $|W| \leq t_1$ and every set $O \subset [1 : n]$ such that $|O| \leq t_2$, there exists a set $J \subseteq O$ such that $|J| \leq t_1$ and $\vec{z}|_{O \setminus J}$ is uniformly distributed and mutually independent of $\mathsf{AssignWires}(\mathsf{ZeroEnc}, W, \emptyset, \vec{z})$, and $\vec{z}|_J$, where $\vec{z} = \mathsf{ZeroEnc}()$. A ZE-Refresh gadget is t-SNI if and only if the inner $\mathsf{ZeroEnc}$ is t-SOI.

4 Gadgets under New Notions for Arbitrary Orders

This section is dedicated to the construction of free t-SNI or t-IOS gadgets at arbitrary orders. We first demonstrate useful propositions to build strong generic gadgets from ZE-refresh gadgets, and then we explore the properties satisfied by the most deployed gadgets from the literature.

4.1 Generic Constructions

ZE-refresh gadgets can be used to build free t-SNI and t-IOS gadgets with larger t, like multiplication or copy gadgets for instance, as illustrated on Figure 2.

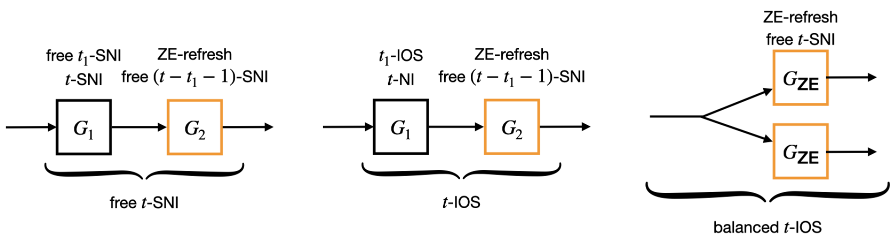

Fig. 2. Illustration of Generic Constructions of Propositions 2, 3 and 4

In particular, Proposition 2 and Proposition 3 demonstrate how to compose free t_1-SNI gadgets which are additionally t-SNI or t-NI (with $t > t_1$) with free $(t - t_1)$-SNI ZE-refresh gadgets to obtain free t-SNI refresh gadgets. Their proofs are given in Section B.1.

Proposition 2. *Let G_1 be an (n-share, 2-to-1) gadget and let G_2 be an n-share ZE-refresh gadget. Let $G = G_2 \circ G_1$. For $t > t_1 \geq 1$, if G_1 is t-SNI and free t_1-SNI, and G_2 is free $(t - t_1 - 1)$-SNI, then G is free t-SNI.*

Proposition 3. *Let G_1 be an (n-share, 2-to-1) gadget and let G_2 be an n-share ZE-refresh gadget. Let $G = G_2 \circ G_1$. For $t > t_1 \geq 1$, if G_1 is t-NI and t_1-IOS, and G_2 is free $(t - t_1 - 1)$-SNI, then G is t-IOS.*

In the same vein, Proposition 4 claims that a copy gadget built from the double application of a free t-SNI ZE-refresh gadget is balanced t-IOS. Its proof is given in Section B.2.

Proposition 4. *Let G_{ZE} be an n-share ZE-refresh gadget. Let $G_{\mathsf{cp}} = (G_{ZE}, G_{ZE})$. If G_{ZE} is free t-SNI, then G_{cp} is balanced t-IOS.*

From Theorem 1, this copy gadget is equivalently free t-SNI (and trivially unbalanced free t-SNI and t-IOS). We state the result for the IOS notion here since such copy gadgets are useful in the context of the IOS composition framework for region probing security (see Section 7).

4.2 Known Gadgets

We now demonstrate that common gadgets satisfy the strong unified notions of free SNI and IOS. Specifically, we prove that the well known ISW multiplication gadget (named after its authors Ishai, Sahai, and Wagner, from [29]) is free-$(n-2)$-SNI (or equivalently balanced $(n-2)$-IOS) and not free-$(n-1)$-SNI and that the subsequent ISW refresh gadget is free-$(n-1)$-SNI. Then, we show that the $\mathcal{O}(n \log n)$ refresh gadget from [8,32] and the copy gadget built from it are both free $(n-1)$-SNI.

4.3 ISW Multiplication and Refresh Gadgets

We recall the ISW multiplication gadget in Algorithm 1.

Algorithm 1: ISW Multiplication [29]

```
Input  : $(a_1, \dots, a_n)$,$(b_1, \dots, b_n)$ input sharings, $\{r_{i,j}\}_{1 \leq i < j \leq n}$ random values
Output: $(c_1, \dots, c_n)$ sharing of $a \cdot b$
for $i \leftarrow 1$ to $n$ do
    $c_i \leftarrow a_i \cdot b_i$;
end
for $i \leftarrow 1$ to $n$ do
    for $j \leftarrow i + 1$ to $n$ do
        $c_i \leftarrow c_i + r_{i,j}$;
        $r_{ji} \leftarrow (a_i \cdot b_j - r_{i,j}) + a_j \cdot b_i$;
        $c_j \leftarrow c_j + r_{j,i}$;
    end
end
return $(c_1, \dots, c_n)$;
```

Our main result for the n-share ISW multiplication gadget is provided in Proposition 5 whose proof is given in Section B.3. The n-share ISW multiplication gadget can thus be used as a building block of a tight composition in the probing model, as proven in Section 6.

Proposition 5. *The n-share ISW multiplication gadget is free-$(n-2)$-SNI.*

Note that from Theorem 1, the ISW multiplication is equivalently balanced $(n-2)$-IOS and it is also $(n-2)$-IOS, which is enough to act as a building block of a tight composition in the region probing model described in Section 7. However it is not free-$(n-1)$-SNI, as claimed in Proposition 6 whose proof is given in Section B.3. In a nutshell, the proof exhibits the following set of $n-1$ probes on the gadget $W = \{a_1 \cdot b_2 - r_{1,2}, \ldots, a_1 \cdot b_n - r_{1,n}\}$ as a counterexample. One can check that a linear combination between all the probes in W and the output share c_1 gives the value $a_1 \cdot (b_1 + \ldots + b_n)$ which involves all of the shares of the second input, leading to a failure.

Proposition 6. *The n-share ISW multiplication gadget is not free $(n-1)$-SNI.*

Remark 1. Note that the ISW multiplication gadget satisfies a security notion slighlty weaker than free $(n-1)$-SNI for which the output shares indexed by $I_1 \cap I_2$ do not need to be simulated.

Even though the ISW multiplication gadget is not free $(n-1)$-SNI, we can use the fact that it is free $(n-2)$-SNI and $(n-1)$-SNI to construct a new free $(n-1)$-SNI gadget, by applying the result of Proposition 2. Since the ISW gadget is free $(n-2)$-SNI, then the ZE-refresh gadget must be free 1-SNI. Hence, we can use any refresh gadget which simply outputs a uniform sharing. For this, we can choose a linear refresh gadget, for instance Algorithm 3 from [20] or the circular refresh gadget from [6].

Corollary 4. *The composition of the n-share ISW multiplication gadget with a free 0-SNI ZE-refresh gadget (like Algorithm 3 from [20] or the circular refresh gadget from [6]) is free $(n-1)$-SNI.*

The proof of Corollary 4 follows directly from Proposition 2.

From the ISW multiplication gadget, one can build the ISW refresh gadget by simply fixing the second input to the constant vector $(1, 0, \ldots, 0)$. Following Proposition 5, the gadget is trivially free $(n-2)$-SNI. Interestingly, it is even free $(n-1)$-SNI as stated in Lemma 1. In fact, the failure sets of probes of size $n-1$ in the case of the ISW multiplication gadget essentially failed because of the manipulation of two input sharings. This is no longer the case in the ISW refresh gadget, which is why it becomes free $(n-1)$-SNI. The full proof is given in Section B.5.

Lemma 1. *The n-share ISW refresh gadget is free-$(n-1)$-SNI.*

Like the ISW multiplication gadget, the ISW refresh gadget can be used as a building block of a tight composition in the probing or in the region probing model (as shown in Sections 6 and 7).

4.4 $\mathcal{O}(n \log n)$ Refresh and Copy Gadgets

In the following, we present the optimized version of the $\mathcal{O}(n \log n)$ refresh gadget from [8], as improved in [32], by removing one layer of randomness. This ZE-refresh gadget relies on the ZeroEnc gadget that we recall in Algorithm 2.

Algorithm 2: QuasiLinearZeroEnc

```
Input   : Number of shares $n$
Output: $(d_1, \dots, d_n)$ such that $d_1 + \cdots + d_n = 0$
1  if $n = 1$ then return 0;
2  if $n = 2$ then
3      $r \leftarrow \$$;
4      return $(r, -r)$;
5  end
6  $(c_1, \dots, c_{\lfloor n/2 \rfloor}) \leftarrow$ QuasiLinearZeroEnc($\lfloor n/2 \rfloor$);
7  $(c_{\lfloor n/2 \rfloor + 1}, \dots, c_n) \leftarrow$ QuasiLinearZeroEnc($\lceil n/2 \rceil$);
8  for $i \leftarrow 1$ to $\lfloor n/2 \rfloor$ do
9      $r \leftarrow \$$;
10     $d_i \leftarrow c_i + r$;
11     $d_{\lfloor n/2 \rfloor + i} \leftarrow c_{\lfloor n/2 \rfloor + i} - r$;
12 end
13 if $n \mod 2 = 1$ then $d_n \leftarrow c_n$;
14 return $(d_1, \dots, d_n)$;
```

Our next proposition states that the ZeroEnc gadget recalled in Algorithm 2 is a $(n-1)$-free encoding of zero. The proof is given in Section B.6.

Proposition 7. *Algorithm 2 is an $(n-1)$-free encoding of zero.*

From Proposition 1 and Proposition 7, we conclude that the corresponding ZE-refresh gadget is free $(n-1)$-SNI.

Corollary 5. *The $\mathcal{O}(n \log n)$ n-share ZE-refresh gadget instantiated with Algorithm 2 as* ZeroEnc *is free $(n-1)$-SNI.*

From Theorem 1, this $\mathcal{O}(n \log n)$ n-share ZE-refresh gadget is equivalently balanced IOS and thus trivially IOS (see Corollary 1). This result confirms (and even generalizes) the proven statement from [28] stating that the gadget is IOS when the number n of shares is a power of two.

Following this result and Proposition 4, the copy gadget built from the double application of the $\mathcal{O}(n \log n)$ ZE-refresh gadget described above is directly balanced t-IOS (or equivalently free t-SNI).

Corollary 6. *Let $G_{ZE,\mathcal{O}(n \log n)}$ be the $\mathcal{O}(n \log n)$ n-share ZE-refresh gadget recalled above. Then the copy gadget defined as $G_{\boldsymbol{cp}} = (G_{ZE,\mathcal{O}(n \log n)}, G_{ZE,\mathcal{O}(n \log n)})$ is balanced t-IOS.*

5 Efficient Verification of Free SNI and IOS

Verification tools for probing security/composition notions such as maskVerif [4,2] or IronMask [13] usually work by enumerating possible tuples of probed wires (for a given number of probes depending on the security notion) and for each of them analyze the number of input shares which are necessary for a perfect simulation. The latter analysis is done by considering linear combinations of the probed variables to eliminate the randomness (assuming linearly introduced randomness) and by listing the input shares involved in the random-free symbolic expressions (which are hence necessary for a perfect simulation). While this approach is sound to verify usual probing composition notions (*e.g.* NI, SNI, PINI) and the random probing notions (RPC, RPE) recently introduced in [11], it is not clear how to extend it to notions such as free SNI or IOS. Indeed, free SNI requires the ability to simulate the probed wires but also to demonstrate that each subset of non-simulated output wires is mutually independent of the simulation. On the other hand, IOS constrains the simulation to be consistent with any given (admissible) values of the input and output sharings and to simulate the probed wires not only from input shares but also with some output shares.

In this section, we show how to extend the existing probing verification approach to the free SNI and IOS notions recalled and generalized in Section 3. We present a set of algorithms to efficiently verify common gadgets under these notions. We then show a few applications of these algorithms as implemented in IronMask.

5.1 Verification Algorithms

Notations. In the following, we consider (n-share, 2-to-1) gadgets, and denote $\overrightarrow{x_1}, \overrightarrow{x_2}$ the input sharings, $\overrightarrow{r}$ the internal randomness, and $\overrightarrow{y}$ the output sharing. Throughout this section, the coordinates of $\overrightarrow{x_1}, \overrightarrow{x_2}$ and $\overrightarrow{r}$ are considered as symbolic variables and we denote $\mathcal{W}$ the set of symbolic *arithmetic* expressions in these symbolic variables (by *arithmetic* we mean with operations and/or constants from $\mathbb{K}$). Under this formalism, any wire w_i in the gadget and any output share y_j lies in W. We note that the verification algorithms presented hereafter can easily be adapted to cover gadgets with a single input.

We exclusively focus on LR-gadgets (as defined in [14]), *i.e.* gadgets for which all random values are additively introduced to compute the output shares. For these common gadgets (see *e.g.* [29,22,9,10]), each wire computes a variable of the form:

$$w = f_w(\overrightarrow{x_1}, \overrightarrow{x_2}) + \overrightarrow{r}^T \cdot \overrightarrow{s_w},$$

for some arithmetic function $f_w : (\mathbb{K}^n)^2 \to \mathbb{K}$, the input sharings $\overrightarrow{x_1}$, $\overrightarrow{x_2} \in \mathbb{K}^n$, the vector $\overrightarrow{r}$ of all random values used by the gadget which is uniformly drawn from $\mathbb{K}^\rho$ (with ρ the number of random gates in the gadget), and some constant vector $\overrightarrow{s_w} \in \mathbb{K}^\rho$.

In the following, we assume that we have access to the following function:

$$\begin{aligned}\mathsf{Gaussian} : \mathcal{W}^k &\to \mathcal{W}^k\\ \overrightarrow{W} = (w_1, \ldots, w_k)^T &\mapsto \overrightarrow{W'} = (w'_1, \ldots, w'_k)^T\end{aligned}$$

such that $\overrightarrow{W'} = N \cdot \overrightarrow{W}$ where N is an invertible matrix in $\mathbb{K}^{k \times k}$ such that $S' = N \cdot S$ with S' the row reduced form (after Gaussian elimination) of the matrix $S := (\overrightarrow{s_{w_1}} | \overrightarrow{s_{w_2}} | \ldots | \overrightarrow{s_{w_k}})^T$ which satistifes $S' = (\overrightarrow{s_{w'_1}} | \overrightarrow{s_{w'_2}} | \ldots | \overrightarrow{s_{w'_v}} | \overrightarrow{0} | \ldots | \overrightarrow{0})^T$ for some $v \in [1 : k]$.

After Gaussian elimination, the expression of each w'_i can be written as

$$w'_i = c_1 \cdot w_{j_1} + \ldots + c_t \cdot w_{j_t} = f_{w'_i}(\overrightarrow{x_1}, \overrightarrow{x_2}) + \overrightarrow{r}^T \cdot \overrightarrow{s_{w'_i}}$$

for $(c_1, \ldots, c_t) \in \mathbb{K}^t$, such that $N_{i,j} \neq 0$ for any $j \in \{j_1, \ldots, j_t\}$ and $N_{i,j} = 0$ for all other coefficients on the same row i of matrix N.

We denote by I_{1,w'_i} (resp. I_{2,w'_i}) the indices of the first input (resp. second input) shares that are contained in the symbolic expression of w'_i, *i.e.* in $f_{w'_i}(\overrightarrow{x_1}, \overrightarrow{x_2})$.

We also denote $\overline{I_{1,w'_i}}$ (resp. $\overline{I_{2,w'_i}}$) the indices of the first input (resp. second input) shares that are involved in the symbolic expression of $f_{w'_i}(\overrightarrow{x_1}, \overrightarrow{x_2}) - g(\overrightarrow{x_1}, \overrightarrow{x_2})$, where $g(\overrightarrow{x_1}, \overrightarrow{x_2}) := \sum_{i=1}^n y_i$ is the (unshared) output of the considered gadget G. For instance, if G is a single input refresh gadget, then we have, $g(\overrightarrow{x_1}) = \sum_{i=1}^n x_{1,i}$, and $\overline{I_{1,w'_i}} = [1 : n] \setminus I_{1,w'_i}$. In the case where G is a multiplication gadget, the products of input shares must be considered in each symbolic expression, *i.e.* $g(\overrightarrow{x_1}, \overrightarrow{x_2}) = \big(\sum_{i=1}^n x_{1,i}\big)\big(\sum_{i=1}^n x_{2,i}\big)$.

Finally, given $\overrightarrow{W} = (w_1, \ldots, w_k)^T$ and $\overrightarrow{W'} = (w'_1, \ldots, w'_k)^T$ such that $\overrightarrow{W} = N \cdot \overrightarrow{W}$ (obtained through a call to $\mathsf{Gaussian}$), we define

$$O_{w'_i} := \{j \in [1 : n] \mid \exists\, \ell \in [1 : k] \text{ s.t. } N_{i,\ell} \neq 0 \text{ and } w_\ell = y_j\}\ .$$

Namely, following the Gaussian elimination, the output share $w_\ell = y_j$ appears in the linear combination defining w'_i. We further define $\overline{O_{w'_i}} := [1 : n] \setminus O_{w'_i}$.

Preliminary results. We present hereafter the sub-algorithms used to verify the main security notions defined or recalled in Section 3. These sub-algorithms take as input a given set of probes $\overrightarrow{W}$ and verify the free SNI or IOS notion for this set. These sub-algorithms have been implemented in the IronMask tool (see [13]) and called on all possible sets of probes $\overrightarrow{W}$ of the target gadget to check that it satisfies the free SNI or IOS notion.

First, we present an efficient method to check that an n-share gadget G is uniform. Namely, this method checks that for any $O \subsetneq [1 : n]$, the output shares in $\overrightarrow{y}|_O$ are all independent and uniform (with $\overrightarrow{y}$ denoting the output sharing of the gadget). The method is described in Algorithm 3.

Algorithm 3: Verification of Output independence and Uniformity

```
Input  : Symbolic expressions $\overrightarrow{y} \in W^n$ of the output sharing of an
         LR-gadget $G$.
Output: success if $G$ is uniform, failure otherwise.
1 $(w'_1, \ldots, w'_n) \leftarrow \mathsf{Gaussian}(y_1, \ldots, y_n)$;
2 $v \leftarrow$ index in $[1:n]$ such that $\forall j > v,\ \overrightarrow{s_{w'_j}} = \overrightarrow{0}$;
3 if $v \neq n-1$ then
4 |   return failure;
5 end
6 return success;
```

Proposition 8. *Algorithm 3 is correct.*

The proof of Proposition 8 is given in Section C.1. In a nutshell, it states that if the gadget is indeed uniform and we perform a Gaussian elimination on the vectors of randomness for the output shares $\overrightarrow{y}$, then we get at most one row which is a linear combination of all of the others outputs in the matrix of the Gaussian procedure. This is exactly the check performed by Algorithm 3.

In the rest of this section, we assume a gadget G on which Algorithm 3 does not fail. If Algorithm 3 fails, then we do not need to go further in the verification since both the free SNI and IOS properties require that the gadget G satisfy uniformity.

Before presenting the full verification algorithm, we state the following useful result. Namely, a set $I \subseteq [1:n]$ satisfies the *free property* for some probes $\overrightarrow{W} \in \mathcal{W}^k$ if, for any $O \subsetneq [1:n] \setminus I$, the output shares $\overrightarrow{y}|_O$ are independent and uniform conditioned on the probes in $\overrightarrow{W}$ and output shares $\overrightarrow{y}|_I$. The proof is given in Section C.4.

Lemma 2. *Let G be a uniform n-share LR-gadget for the base field $\mathbb{K} = \mathbb{F}_2$. Let $\overrightarrow{W} = (w_1, \ldots, w_k)$ be a tuple of internal probes on G and let*

$$(f_1, \ldots, f_{n+k-1}) \leftarrow \mathit{Gaussian}(w_1, \ldots, w_k, y_1, \ldots, y_{n-1}) \; .$$

Let $v \in [0 : n+k-1]$ such that $\overrightarrow{s_{f_i}} \neq \overrightarrow{0}$ for all $i \leq v$ and $\overrightarrow{s_{f_i}} = \overrightarrow{0}$ for all $i > v$ and denote $P = \{f_{v+1}, \ldots, f_{n+k-1}\}$.

For any partition $P_1 \cup P_2 = P$ (with $P_1 \cap P_2 = \emptyset$), the set

$$I = \Big(\bigcup_{f_i \in P_1} O_{f_i} \Big) \cup \Big(\bigcup_{f_i \in P_2} \overline{O_{f_i}} \Big) \; , \tag{1}$$

with $\overline{O_{f_i}} = [1:n] \setminus O_{f_i}$, satisfies the free property for $\overrightarrow{W}$.

Moreover, any set I' which is not a superset of a set I in the form of (1) does not satisfy the free property for $\overrightarrow{W}$.

Verification algorithms. We present hereafter the verification algorithms which check whether a given set of probes on a gadget represents a failure for free SNI or IOS property. The complete procedure is described in Algorithms 4, 5, and 6. They are implemented in the IronMask verification tool, which iterates over all possible sets of probes on the gadget and calls the procedure on each of them. The exploration of the different sets of probes uses the optimizations already integrated in the tool and presented in [13]. As in Lemma 2, the description assumes that the base field is $\mathbb{K} = \mathbb{F}_2$. We later discuss the general case of any base field $\mathbb{K}$.

Algorithm 4 performs the preliminary steps common to the verification of free SNI and IOS. Namely, it determines the sets of input shares necessary to simulate the given set of probes $\overrightarrow{W}$. This is done through the Gaussian elimination $(g_1, \dots, g_t) \leftarrow \mathsf{Gaussian}(w_1, \dots, w_t)$ performed on line 2 and then constructing the sets of input shares

$$I_1 = \bigcup_{i \in [v+1:t]} I_{1,g_i} \quad \text{and} \quad I_2 = \bigcup_{i \in [v+1:t]} I_{2,g_i}$$

on lines 3 to 6, where, as introduced above, I_{j,g_i} is the set of shares from $\overrightarrow{x_j}$ which are involved in the expression g_i. At this point, we already have a failure if at least one of the sets I_1 or I_2 is of size larger than $|\overrightarrow{W}|$. This indeed consists in an SNI failure, which is automatically a failure for free SNI and IOS. This test is performed on line 7. Next, on lines 8 and 9, the algorithm prepares the inputs to the inner algorithm, which either checks free SNI or IOS by performing the Gaussian elimination described in Lemma 2. Depending on the property, Algorithm 4 then calls one of Algorithms 5 or 6. In both algorithms, we perform a direct application of Lemma 2. In other words, we try to find a set of output shares that satisfies free SNI or IOS requirements using the Lemma result. The correctness of this procedure is proved in Lemmas 3 and 4. The full proof of the lemmas are given in Section C.2 and Section C.3.

Lemma 3. *Algorithms 4 and 5 are correct when checking free t-SNI property.*

Lemma 4. *Algorithms 4 and 6 are correct when checking t-IOS property.*

Optimization. We can optimize the execution time of Algorithms 5 and 6 by reducing the number of sets to test, *i.e.* the number of subsets to consider in the main loop on line 2 in both algorithms.

Observe that in Algorithm 5, for each $i \in C = \{v+1, \dots, n+t-1\}$, if $|I_{1,f_i} \cup O_{f_i}| > t$ or $|I_{2,f_i} \cup O_{f_i}| > t$, then for any subset C' of C such that $i \in C'$, the constructed sets I'_1 and I'_2 will be of size bigger than t, which means that we cannot use them to satisfy the free SNI property. In this case, we have no choice but to use $\overline{I_{1,f_i}} \cup \overline{O_{f_i}}$ and $\overline{I_{2,f_i}} \cup \overline{O_{f_i}}$ to add to the sets I'_1 and I'_2 respectively. The observation works analoguously in the case where $|\overline{I_{1,f_i}} \cup \overline{O_{f_i}}| > t$ or $|\overline{I_{2,f_i}} \cup \overline{O_{f_i}}| > t$. We hence apply this in a preprocessing

Algorithm 4: Verification Algorithm for free-t-SNI or t-IOS for a single set of probes

```
Input  : $\overrightarrow{W} = (w_1, \ldots, w_t) \in \mathcal{W}^t$ a tuple of $t$ internal probes on an
         ($n$-share, 2-to-1) LR-gadget $G$.
         property $\in$ {freeSNI, IOS} to check.
Output: false if $\overrightarrow{W}$ is a failure for property, true otherwise.
1  $I_1 \leftarrow \{\}$,   $I_2 \leftarrow \{\}$;
2  $(g_1, \ldots, g_t) \leftarrow \mathsf{Gaussian}(w_1, \ldots, w_t)$;
3  $v \leftarrow$ index in $[1 : t]$ such that $\forall j > v$, $\overrightarrow{s_{g_j}} = \overrightarrow{0}$;
4  for $i \in [v + 1 : t]$ do
5      $I_1 \leftarrow I_1 \cup I_{1,g_i}$,    $I_2 \leftarrow I_2 \cup I_{2,g_i}$;
6  end
7  if $|I_1| > t$ or $|I_2| > t$ then return false;
8  $(f_1, \ldots, f_{n+t-1}) \leftarrow \mathsf{Gaussian}(w_1, \ldots, w_t, y_1, \ldots, y_{n-1})$;
9  $v \leftarrow$ index in $[1 : n + t - 1]$ such that $\forall j > v$, $\overrightarrow{s_{f_j}} = \overrightarrow{0}$;
10 return check_{property}($t$, $I_1$, $I_2$, $(f_{v+1}, \ldots, f_{n+t-1})$);
```

Algorithm 5: check_freeSNI

```
Input  : $t$, $I_1$, $I_2$, $(f_{v+1}, ..., f_{n+t-1})$
         from Algorithm 4
Output: false if failure for
         free-$t$-SNI, true otherwise.
1  $C \leftarrow \{v + 1, \ldots, n + t - 1\}$;
2  for each $C'$ such that $C' \subseteq C$ do
3      $I'_1 \leftarrow I_1$,    $I'_2 \leftarrow I_2$;
4      for $i \in C'$ do
5          $I'_1 \leftarrow I'_1 \cup I_{1,f_i} \cup O_{f_i}$;
6          $I'_2 \leftarrow I'_2 \cup I_{2,f_i} \cup O_{f_i}$;
7      end
8      for $i \in C \setminus C'$ do
9          $I'_1 \leftarrow I'_1 \cup \overline{I_{1,f_i}} \cup \overline{O_{f_i}}$;
10         $I'_2 \leftarrow I'_2 \cup \overline{I_{2,f_i}} \cup \overline{O_{f_i}}$;
11     end
12     if $|I'_1| \le t$ and $|I'_2| \le t$ then
           return true;
13 end
14 return false;
```

Algorithm 6: check_IOS

```
Input  : $t$, $I_1$, $I_2$, $(f_{v+1}, ..., f_{n+t-1})$
         from Algorithm 4
Output: false if failure for $t$-IOS,
         true otherwise.
1  $C \leftarrow \{v + 1, \ldots, n + t - 1\}$;
2  for each $C'$ such that $C' \subseteq C$ do
3      $I'_1 \leftarrow I_1$,    $I'_2 \leftarrow I_2$,    $J' \leftarrow \{\}$;
4      for $i \in C'$ do
5          $I'_1 \leftarrow I'_1 \cup I_{1,f_i}$;
6          $I'_2 \leftarrow I'_2 \cup I_{2,f_i}$;
7          $J' \leftarrow J' \cup O_{f_i}$;
8      end
9      for $i \in C \setminus C'$ do
10         $I'_1 \leftarrow I'_1 \cup \overline{I_{1,f_i}}$;
11         $I'_2 \leftarrow I'_2 \cup \overline{I_{2,f_i}}$;
12         $J' \leftarrow J' \cup \overline{O_{f_i}}$;
13     end
14     if $|I'_1| \le t$ and $|I'_2| \le t$ and
        $|J'| \le t$ then return true;
15 end
16 return false;
```

phase to the algorithm, where for each such $i \in C = \{v+1, \ldots, n+t-1\}$, we update the sets I_1 and I_2 correspondingly, and remove the index i from C. Then, in the loop afterwards, we only consider a subset of C. In fact, in the case where $t < n/2$, we always have either $|I_{1,f_i} \cup O_{f_i}| \leq t$ or $|\overline{I_{2,f_i}} \cup \overline{O_{f_i}}| \leq t$ for each $i \in C$, hence there is only one possible partition that could work for the property. This means that we do not need to loop on all possible subsets of C anymore but only go through all indices once. In the case where $t \geq n/2$, the experimental results in the next section show that with the optimization, we end up considering, on average, only 1 or 2 subsets of C in the loop before finding the partition that satisfies the property. In other words, Algorithm 5 for checking the free SNI property is in $\mathcal{O}(n+t)$ whenever $t < n/2$ while it might be exponential in $\mathcal{O}(2^{n+t})$ otherwise to explore all the subsets $C' \subseteq C$. But interestingly in practice, this exploration does not explode for tested gadgets for which the overhead is at most a factor 2.

Finally, we can also apply the same optimization for Algorithm 6 where instead of considering the sizes of the sets $I_{1,f_i} \cup O_{f_i}$ and $I_{2,f_i} \cup O_{f_i}$, we have to consider the sets I_{1,f_i}, I_{2,f_i}, and O_{f_i} independently.

Generalization to any base field $\mathbb{K}$. The verification technique presented in this section can be generalized from $\mathbb{F}_2$ to any base field $\mathbb{K}$. First, Algorithm 3 which tests if the output sharing of the gadget is uniform can be applied in the exact same way. Then, to verify free SNI and IOS properties, we can also apply Algorithm 4 without any changes by correctness of the verification procedure of IronMask on any base field as proven in [14]. Next, we need to slightly modify Algorithms 5 and 6. For each $i \in C$, where C is the set defined on the first line of the algorithms, instead of considering the equation f_i (*i.e.* the sets I_{1,f_i}, I_{2,f_i} and O_{f_i}), and $f_i + \sum_j y_j$ (*i.e.* the sets $\overline{I_{1,f_i}}$, $\overline{I_{2,f_i}}$ and $\overline{O_{f_i}}$), we need to consider substracting from f_i different multiples of $\sum_j y_j$ by constant factors of $\mathbb{K}$. In general, we need to consider the $|\mathbb{K}|$ different multiples of $\sum_j y_j$ (where $|\mathbb{K}|$ is the size of the field). However, we can observe that each f_i can be written as

$$f_i = c_{1,f_1} \cdot w_1 + \ldots + c_{t,f_1} \cdot w_t + d_{1,f_1} \cdot y_1 + \ldots d_{n-1,f_1} \cdot y_{n-1}$$

for coefficients $(c_{1,f_1}, \ldots, c_{t,f_1}, d_{1,f_1}, \ldots, d_{n-1,f_1}) \in \mathbb{K}^{t+n-1}$ (on $\mathbb{F}_2$, all of these coefficients are either 0 or 1). Hence, for each $i \in C$, we only need to consider the n equations $f_i, f_i - d_{1,f_1} \cdot \sum_j y_j, \ldots, f_i - d_{n-1,f_1} \cdot \sum_j y_j$. The correctness of this procedure can be proven in a similar way to the result of Lemma 2. Indeed, the cost of Algorithms 5 and 6 becomes more exponential: in the case of $\mathbb{F}_2$, the algorithms perform at most $2^{n+t-v-2}$ iterations of the main loop, while on a larger base field $\mathbb{K}$, the algorithms perform at most $n^{n+t-v-2}$ iterations to find the sets satisfying the free SNI or IOS property.

5.2 Application to Concrete Gadgets

We now demonstrate the verification of the recalled or newly defined properties of Section 3 for the most common multiplication and refresh gadgets used in the literature for a reasonable number of shares. We execute the IronMask tool extended with Algorithm 4 on a single core of a 2.4 GHz Intel Core i9 8 cores with a 16 GB RAM. Our results are given in Table 1.

The first column displays the number of shares n (or the range when applicable) for which the gadget is tested. The second column gives the complexity of the gadget with its number of random variables and its number of intermediate variables. Then, the three properties SNI, IOS, and free SNI are tested for $t = n-1$, and the verification result is given in the `Res.` column. Finally, the last column displays the verification time and the highest t for which the property is verified when it is not for $t = n-1$.

First, Table 1 confirms for small numbers of shares that refresh gadgets from Section 4, *i.e.* the ISW refresh gadget and the optimized $\mathcal{O}(n \log n)$ ZE-refresh gadget, are free $(n-1)$-SNI and $(n-1)$-IOS. The table also shows the verification result for the ISW multiplication gadget, which is $(n-1)$-SNI for up to 7 shares, and confirms that the gadget is only free $(n-2)$-SNI and $(n-2)$-IOS. Conversely, the multiplication gadget from Corollary 4 is free $(n-1)$-SNI for up to 7 shares, as expected.

As an additional multiplication gadget, we test the parallel construction from [6, SNI gadgets from Table 4] for up to 8 shares.[10] The results in the table show that the gadget is always $(n-1)$-SNI, but depending on the number of shares, is free t-SNI and t'-IOS for different values of $t, t' \geq 1$. Interestingly, the values of t and t' change differently depending on the number of shares in the construction. Also, the gadget is only free $(n-1)$-SNI and $(n-1)$-IOS for $n = 4$.

Finally, we test the optimized 8-shares refresh gadget from [18], which is slightly more efficient than the 8-shares $\mathcal{O}(n \log n)$ refresh gadget from Corollary 5, and uses one less random value. The gadget is confirmed to be free 7-SNI and 7-IOS. We also test the circular refresh gadget as constructed in [3] as a ZE-refresh and confirm that it is free $(n-1)$-SNI and $(n-1)$-IOS for n up to 11 shares.

In all of the tests, we evaluate the impact of the optimization on Algorithms 5 and 6. In order to do this, for each checked gadget, we compute the total number of iterations performed in the loops of Algorithm 5 for free SNI, or Algorithm 6 for IOS, for all of the tuples, divided by the total number of tuples which require at least one iteration of the loop. We observe that for any of the tested gadgets, this ratio does not exceed 1.2, which means that for each tuple of probes, we need, on average, one or at most two iterations of the loop of Algorithm 5 or 6 before finding the sets that satisfy the respective property. This shows that with the optimization introduced to the algorithms, the cost is not exponential anymore, meaning that we do not need to test all possible partitions as described in Lemma 2, but only at most one or two before finding the right one.

[10] When we implemented the parallel multiplication gadgets from [6, SNI gadgets from Table 4], we detected a correctness flaw for the case $n \mod 4 = 2$. That is why we could not test such a gadget with six shares.

Table 1. IronMask verification results and execution time.

# shares	Complexity # rand. - # var.	Property	Res.	Verification time
Multiplication Gadgets				
ISW multiplication from [29]				
$n \leq 7$	≤ 21, ≤ 161	SNI	✓	$\leq$ 10sec
		IOS	✗	$\leq$ 10sec ($t_{\max} = n-2$)
		free SNI	✗	$\leq$ 10sec ($t_{\max} = n-2$)
Multiplication from Corollary 4				
$n \leq 7$	≤ 27, ≤ 179	SNI	✓	$\leq$ 1min30sec
		IOS	✓	$\leq$ 3min
		free SNI	✓	$\leq$ 3min
Parallel multiplication from [6, SNI gadgets from Table 4]				
3	3, 27	SNI	✓	$\leq$ 1sec
		IOS	✗	$\leq$ 1sec ($t_{\max} = 1$)
		free SNI	✗	$\leq$ 1sec ($t_{\max} = 1$)
4	8, 56	SNI	✓	$\leq$ 1sec
		IOS	✓	$\leq$ 1sec
		free SNI	✓	$\leq$ 1sec
5	10, 80	SNI	✓	$\leq$ 1sec
		IOS	✓	$\leq$ 1sec
		free SNI	✗	$\leq$ 1sec ($t_{\max} = 3$)
7	21, 161	SNI	✓	7sec
		IOS	✗	4sec ($t_{\max} = 5$)
		free SNI	✗	4sec ($t_{\max} = 5$)
8	24, 200	SNI	✓	6min
		IOS	✗	11sec ($t_{\max} = 5$)
		free SNI	✗	$\leq$ 1sec ($t_{\max} = 3$)
Refresh Gadgets				
ISW refresh gadget from [29]				
$n \leq 8$	≤ 28, ≤ 84	SNI	✓	$\leq$ 30sec
		IOS	✓	$\leq$ 2min30sec
		free SNI	✓	$\leq$ 2min30sec
$\mathcal{O}(n \log n)$ refresh gadget from Corollary 5				
$n \leq 11$	≤ 17, ≤ 51	SNI	✓	$\leq$ 1min
		IOS	✓	$\leq$ 3min
		free SNI	✓	$\leq$ 3min
Optimized 8-share refresh gadget from [18]				
8	11, 33	SNI	✓	$\leq$ 1sec
		IOS	✓	$\leq$ 1sec
		free SNI	✓	$\leq$ 1sec
Circular refresh gadget (`RefreshBlock1; RefreshBlock3`) from [3]				
$n \leq 11$	≤ 22, ≤ 66	SNI	✓	$\leq$ 6min
		IOS	✓	$\leq$ 76min
		free SNI	✓	$\leq$ 71min

6 Probing Model Composition

In this section, we revisit the security proof of the tight private circuits (TPC) [12]. We begin with an overview of this proof, and then show that there is a flaw in one of its lemmas. We however remark that the security of the standard circuits verified by tightPROVE still holds, since the ISW multiplication gadgets are free $(n-2)$-SNI. We are actually able to leverage the new security notions to generalize the tight private circuits to circuits using other types of gadgets (*i.e.* not only ISW multiplication/refresh and additions). This generalization allows, for example, to use the $\mathcal{O}(n \log n)$ refresh gadget from Algorithm 2 in TPCs, instead of the less efficient ISW refresh gadget.

6.1 TPC security proof

The TPCs [12] are *standard shared circuits*, that is, masked circuits obtained by composing sharewise addition gadgets, ISW multiplication gadgets and ISW refresh gadgets (next collectively denoted as ISW gadgets). The security proof for TPCs is composed of two parts: the equivalence between t-probing security and a simpler leakage model (Game 3), and a technique to verify the security of a circuit in this model (instantiated with the tightPROVE tool).

In this paper, we focus on the first part of the proof, which is carried by showing the successive equivalence of four different security games. In each of these games, an adversary $\mathcal{A}$ selects some probes and secret inputs, and a simulator $\mathcal{S}$ has to perfectly simulate the set of probes in the circuit. A simulator wins the game if the simulation $\mathsf{ExpSim}_i(\mathcal{A}, \mathcal{S}, C)$ has the same distribution as the true set of probes $\mathsf{ExpReal}_i(\mathcal{A}, C)$, and a circuit is secure in a game if for all adversaries, there exists a simulator that wins the game.

The first game of the reduction (Game 0, see Figure 4) is the t-probing security game. Next, Game 1 is a variant of Game 0 where the adversary cannot put probes inside RNL ("refresh or non-linear") gadgets, and instead of a probe in a ISW gadget, can get a probe on one share of each of its input sharings. The two following games (see Figure 5) operate on a flattened circuit $C' = \mathsf{Flatten}(C)$, in which the output sharing of each ISW gadget is replaced by a new input sharing of the circuit. In Game 2, the adversary probes C', and the constraints on the probes are the same as in Game 1. Finally, in Game 3, the set of probes follow the same rules, but is additionally limited to probes on the input shares of two-input ISW gadgets (i.e., the multiplication gadgets).

The reduction between the games is illustrated in Figure 3, and we next present the main ideas of the reductions. First, the equivalence between Game 0 and Game 1 is based, in one direction, on the simulation of the internal probes using knowledge of the input shares, and in the other direction by exploiting the probing completeness of the ISW gadgets. Next, the equivalence between Game 1 and Game 2 is a consequence of [12, Lemma 1], which states that the output sharing of any SNI gadget, hence of ISW gadgets, is uniform. Finally, the equivalence of Game 2 and Game 3 relies on the observation that without probes on the multiplication gadgets, any standard circuit would be secure, and we can actually show that the other probes are redundant with the ones in multiplication gadgets.

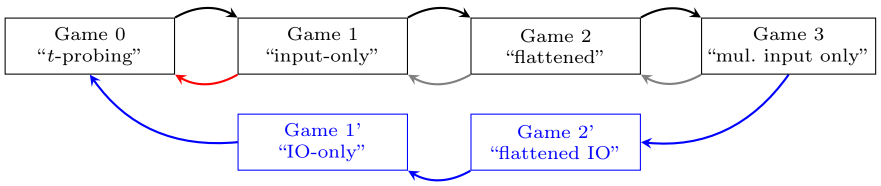

Fig. 3. Representation of the flawed and fixed TPC proofs (an arrow Game $i \to$ Game j means "Security for Game i implies security for Game j."). The flawed implication is in red and the new reductions are in blue.

```
$\mathsf{ExpReal}_i(\mathcal{A}, C)$
  1. $(\mathcal{P}_i, x_1, \ldots, x_m) \leftarrow \mathcal{A}()$
  2. $\overrightarrow{x_1} \leftarrow \mathsf{Enc}(x_1), \ldots, \overrightarrow{x_m} \leftarrow \mathsf{Enc}(x_m)$
  3. $(v_1, \ldots, v_q) \leftarrow C(\overrightarrow{x_1}, \ldots, \overrightarrow{x_m})_{\mathcal{P}_i}$
  4. Return $(v_1, \ldots, v_q)$
```

```
$\mathsf{ExpSim}_i(\mathcal{A}, \mathcal{S}, C)$
  1. $(\mathcal{P}_i, x_1, \ldots, x_m) \leftarrow \mathcal{A}()$
  2. $(v_1, \ldots, v_q) \leftarrow \mathcal{S}(\mathcal{P}_i)$
  3. Return $(v_1, \ldots, v_q)$
```

Fig. 4. Game 0, Game 1 and Game 1' ($i = 0, 1, 1'$ respectively). For Game 0, $\mathcal{P}_0$ can be any set of $q = t$ probes in C. For Game 1, the probes in ISW or RNL gadgets are moved to their input shares while for Game 1', the corresponding output probes (following the balanced IOS definition) are also included.

6.2 Proof flaw

The reduction of security w.r.t. Game 0 to the security w.r.t. Game 1 (security implication from Game 1 to Game 0 in Figure 3, [12, Proposition 4]) has a flawed proof. This proof relies on a simulation argument. At a gadget level, a probe in a ISW multiplication gadget can be simulated when one share of each input is known, since the gadget is SNI. However, when handling the composition, the simulation may need the values of output shares of a multiplication gadget. Thanks to [12, Lemma 1]), any set of $n-1$ output shares of the ISW multiplication is uniform and independent of the input sharings. This observation is used in the proof of the proposition to claim that the output shares can be simulated as fresh randomness. However, this is not correct in presence of probes in the multiplication gadget, whose value may not be independent of the output shares.

Let us now show with a counter-example that the proof cannot be easily fixed, *i.e.* that it cannot be fixed while relying only on the SNI security of the ISW multiplication. We consider the circuit described in Algorithm 7. In the first multiplication gadget, the adversary can probe the last intermediate sum in

```
$\mathsf{ExpReal}_i(\mathcal{A}, C)$
  1. $C' \leftarrow \mathsf{Flatten}(C)$
  2. $(\mathcal{P}_i, x_1, \dots, x_M) \leftarrow \mathcal{A}()$
  3. $\overrightarrow{x_1} \leftarrow \mathsf{Enc}(x_1), \dots, \overrightarrow{x_M} \leftarrow \mathsf{Enc}(x_M)$
  4. $(v_1, \dots, v_q) \leftarrow C(\overrightarrow{x_1}, \dots, \overrightarrow{x_M})_{\mathcal{P}_i}$
  5. Return $(v_1, \dots, v_q)$
```

```
$\mathsf{ExpSim}_i(\mathcal{A}, \mathcal{S}, C)$
  1. $C' \leftarrow \mathsf{Flatten}(C)$
  2. $(\mathcal{P}_i, x_1, \dots, x_M) \leftarrow \mathcal{A}()$
  3. $(v_1, \dots, v_q) \leftarrow S(\mathcal{P}_i)$
  4. Return $(v_1, \dots, v_q)$
```

Fig. 5. Game 2, Game 2' and Game 3 ($i = 2, 2', 3$ respectively). For Game 2, the probes are the same as in Game 1, and for Game 2' we add probes on the new input shares that correspond to probes on output shares of RNL gadgets in $\mathcal{P}_{1'}$. For Game 3, $\mathcal{P}_3$ contains only probes on input shares of the two-input ISW/RNL gadgets.

Algorithm 7: Broken ISW composition

```
Input : sharings $(a_1, \dots, a_4)$ and $(b_1, \dots, b_4)$
1 $(c_1, \dots c_4) \leftarrow \mathsf{ISW\text{-}MUL}((a_1, \dots, a_4), (b_1, \dots, b_4))$ ;    // Uses randomness $r_{i,j}$
2 $(d_1, d_2, d_3, d_4) \leftarrow (c_2, c_3, c_4, c_1)$
3 for $i \leftarrow 1$ to 4 do
4     $e_i \leftarrow a_i \oplus d_i$;
5 end
6 $(f_1, \dots f_4) \leftarrow \mathsf{ISW\text{-}MUL}((e_1, \dots, e_4), (a_1, \dots, a_4))$
```

the computation of c_2: $p_1 = S$ with $c_2 = S \oplus r_{2,3}$, and $p_2 = (a_2 b_3 \oplus r_{2,3}) \oplus a_3 b_2$. It can also probe $p_3 = e_1 a_4$ in the second multiplication. Since $p_3 = (a_1 \oplus c_2) a_4$, the adversary may compute $p_1 \oplus p_2 \oplus p_3 = (S \oplus r_{2,3})(a_4 \oplus 1) \oplus a_2 b_3 \oplus a_3 b_2 \oplus a_1 a_4$. When S, b_2 and b_3 are uniform randomness, and (a_1, a_2, a_3, a_4) is a uniform sharing of a, the value of previous expression depends on a, therefore the circuit is not 3-probing secure.

Let us remark that the presence of the rotation between the sharing $(c_1, \dots, c_4)$ and the sharing $(d_1, \dots, d_4)$ is needed the exhibit the attack. If we consider this rotation as part of the first multiplication gadget, it remains $(n-1)$-SNI (as the size of a set of output shares is not impacted by the rotation), showing that, for the proof to work, we have to rely on a stronger property. However, a rotation break the free SNI property, and indeed, we next show that, since the ISW multiplication is free $(n-2)$-SNI, the TPC proof can be repaired.

6.3 Generalized proof

We now fix the above flaw and generalize the proof of [12] to cover more types of gadgets. Namely, we allow any probing complete gadget that is free $(n-2)$-SNI and $(n-1)$-NI instead of only ISW multiplication and refresh gadgets. Moreover, instead of only additions, our proof works with arbitrary affine sharewise gadgets.

Definition 13 (GTPC). *An n-*generalized tight private circuit *(GTPC) is an n-share masked circuit composed of SA gadgets and RNL gadgets (standing for refresh or non-linear). A gadget is SA if it is sharewise affine, and a gadget is*

RNL if it is $(n-1)$-probing complete, $(n-1)$-NI, free $(n-2)$-SNI, and has one or two input shares.

Let us remark that this definition covers the standard shared circuits, since ISW multiplication and refresh gadgets are free $(n-2)$-SNI and $(n-1)$-NI.

Theorem 4 (All games are equivalent). *Let G be any of the games of Figure 3. A n-GTPC is $(n-1)$-probing secure if and only if for every adversary $\mathcal{A}$ there exists a simulator $\mathcal{S}$ that wins G.*

The proof idea is as follows (the full proof is given in Section D): we introduce two new intermediate games (Game 1' and Game 2', shown in Figure 3) for the reduction of Game 0 security to Game 3 security, while for the tightness (reduction of Game 3 security to Game 0) we discuss the minor changes needed to generalize the proofs of [12] to the generalized circuit assumptions.

In a nutshell, with the two new games, we introduce the output probes needed to simulate the probes in the RNL gadgets, using a balanced IOS simulator, as shown in Figure 4 and Figure 5. Security in Game 1' therefore implies security in Game 0 thanks to the balanced IOS simulation, while, as previously, the reduction of Game 1' to Game 2' is a consequence of the uniformity of output sharings of RNL gadgets. Finally, the reduction of Game 2' to Game 3 has to deal with the additional probes on the circuit inputs and the more general affine gadgets, but the core proof idea remain the same: any attack in Game 2' has its source in a probe in a 2-input RNL gadget.

Let us remark that we only require free $(n-2)$-SNI and not free $(n-1)$-SNI, which allows us to cover ISW gadgets with our proof. This relaxed requirement is only significant when the adversary puts all its $n-1$ probes in a single gadget. In that case, the part of the circuit that uses the output of this gadget is not probed, hence we do not care about the output shares distribution, which is why $(n-1)$-NI is a sufficient requirement.

7 Region Probing Model Composition

In this section, we revisit the IOS composition framework introduced in [27] to achieve security in the region probing model. As recalled in Section 2, the region probing model is a strong version of the probing model in which the adversary gets to place t probes in each gadget (or region) of the circuit. This model is relevant in practice as being closer to actual side-channel leakages (providing information on all the gadgets of an implementation) which is formally captured by a reduction from the *noisy leakage model* [34,24].

We show hereafter that another generalized version of tight private circuits (TPC), which we name *region tight private circuit* (RTPC), enjoys a tight security in the region probing model.

Composing several sharewise affine gadgets yields a larger sharewise affine gadget which we shall call a *sharewise affine region* (or SA region) hereafter when it is delimited by IOS gadgets. Consider such an SA region

$$G : (\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell}) \mapsto (\overrightarrow{y_1}, \ldots, \overrightarrow{y_m})$$

computing a function $g : (x_1, \ldots, x_\ell) \to (g_1(x_1, \ldots, x_\ell), \ldots, g_m(x_1, \ldots, x_\ell))$. We say that this SA region is *full rank* if the computed coordinate functions g_i's are linearly independent, *i.e.*, there exist no constant $\alpha \in \mathbb{K}^m$ and $\beta \in \mathbb{K}$ such that $\langle \alpha, g(x_1, \ldots, x_\ell) \rangle = \beta$ (for all $x_1, \ldots, x_\ell$).

Definition 14 (RTPC). *An n-*region tight private circuit *(RTPC) is an n-share masked circuit composed of SA gadgets and RCNL gadgets (standing for refresh, copy or non-linear). A gadget is SA if it is sharewise affine while a gadget is RCNL if it is $\lfloor (n-1)/3 \rfloor$-IOS. Moreover, all the SA regions of an RTPC are full rank.*

We stress that for $n \geq 3$, a generalized tight private circuit (GTPC) under Definition 13 is also an RTPC under the above definition provided that its SA regions are full rank. Moreover, the common example of masked circuits composed of ISW multiplications, $\mathcal{O}(n \log n)$ refresh gadgets and sharewise affine gadgets are both GTPC and RTPC (once again provided that their SA regions are full rank).

Remark 2. We note that an SA region which is not full rank can be split into smaller full rank SA regions by introducing IOS refresh and/or copy gadgets. Consider for example the function:

$$g(x_1, x_2, x_3) \mapsto \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} x_1 + x_2 \\ x_2 + x_3 \\ x_1 + x_3 \end{pmatrix}.$$

An SA region composed of three addition gadgets computing this function is not full rank. By introducing IOS copies of x_1 and x_3 (*e.g.* using the copy gadget of Section 4.4), this SA region can be split into two full rank SA regions, the first one computing (y_1, y_2) and the second one computing y_3.

An RTPC tolerates up to $t = \lfloor (n-1)/3 \rfloor$ probes per IOS gadget and per (full rank) SA region, which gives a probing rate r of $\lfloor (n-1)/3 \rfloor$ over the size of the largest region (IOS gadget or SA region). This is formalized in the following theorem (the proof is given in Appendix E).

Theorem 5. *An n-region tight private circuit is r-region probing secure with*

$$r = \min\left(\frac{\lfloor (n-1)/3 \rfloor}{|\mathsf{SAR}|}, \frac{\lfloor (n-1)/3 \rfloor}{|\mathsf{RCNL}|} \right)$$

with $|\mathsf{SAR}|$ the size of the largest SA region in the circuit and $|\mathsf{RCNL}|$ the size of the largest RCNL gadget in the circuit.

The above theorem generalizes the composition result from [27] to RTPC. Compared to this previous result, the above theorem is more general in several aspects:

- We consider IOS gadgets which are not necessarily refresh gadgets. We can thus take advantage of the IOS property of multiplication gadgets, such as the ISW multiplication gadget, and use IOS copy gadgets.
- We do not require the underlying circuit to insert an IOS refresh gadget between any two non-IOS gadgets whereas the *standard circuit compilers* defined in [27] impose this requirement (which is needed by the composition theorem). As a result, we can use large sharewise affine regions and save many refresh gadgets.
- The underlying circuit can use IOS copy gadgets where a standard circuit compiler from [27] would use a copy gadget surrounded by three IOS refreshes. Our construction of IOS copy gadget (see Section 4.4) is slightly more efficient (*i.e.* equivalent to 2 IOS refreshes).
- We do not consider probing secure gadgets (besides sharewise affine gadgets) which seemingly is a loss of generality but is actually not as we show hereafter.

Probing secure gadgets in our framework. The power of the IOS framework introduced in [27] resides in the fact that it enables to securely compose gadgets which only satisfy probing security and no further probing composition notions. Sharewise affine gadgets are an example of probing-secure gadgets (which are not *e.g.* SNI or IOS) but other probing secure gadgets exist which are not sharewise affine. An example is the multiplication gadget considered in [27] (previously introduced in [26]) which achieves quasilinear complexity $\mathcal{O}(n \log n)$ (with some constraints on the underlying field $\mathbb{K}$). While such gadgets are seemingly out of the scope of RTPC circuits they can be augmented to fit into it thanks to the following proposition (the proof is given in Section E).

Proposition 9. *Let $G : (x_1, x_2) \mapsto y$ be a t-probing secure gadget and G_{ref} be a t-IOS refresh gadget. The gadget G' defined as*

$$G' : (x_1, x_2) \mapsto G_{\mathsf{ref}}\big(G\big(G_{\mathsf{ref}}(\overrightarrow{x_1}), G_{\mathsf{ref}}(\overrightarrow{x_2})\big)\big)$$

is (balanced) t-IOS.

The above proposition shows that any probing secure gadget can be made IOS by surrounding it with IOS refreshes (which is the principle of the composition from [27]). This way, we can include non-sharwise affine probing secure gadgets (such as the quasilinear multiplication gadget) in our extended IOS framework by first composing them with IOS refreshes.

Acknowledgements. This work is partly supported by SGS and the French FUI-AAP25 VeriSiCC project. The authors would also like to thank Benjamin Grégoire for his insightful comments and constructive discussions.

References

1. Marcin Andrychowicz, Stefan Dziembowski, and Sebastian Faust. Circuit compilers with $O(1/\log(n))$ leakage rate. In Marc Fischlin and Jean-Sébastien Coron,

editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 586–615, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

2. Gilles Barthe, Sonia Belaïd, Gaëtan Cassiers, Pierre-Alain Fouque, Benjamin Grégoire, and François-Xavier Standaert. maskVerif: Automated verification of higher-order masking in presence of physical defaults. In Kazue Sako, Steve Schneider, and Peter Y. A. Ryan, editors, *ESORICS 2019: 24th European Symposium on Research in Computer Security, Part I*, volume 11735 of *Lecture Notes in Computer Science*, pages 300–318, Luxembourg, September 23–27, 2019. Springer, Heidelberg, Germany.
3. Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, François-Xavier Standaert, and Pierre-Yves Strub. Improved parallel mask refreshing algorithms: generic solutions with parametrized non-interference and automated optimizations. *J. Cryptogr. Eng.*, 10(1):17–26, 2020.
4. Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, and Pierre-Yves Strub. Verified proofs of higher-order masking. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 457–485, Sofia, Bulgaria, April 26–30, 2015. Springer, Heidelberg, Germany.
5. Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, Pierre-Yves Strub, and Rébecca Zucchini. Strong non-interference and type-directed higher-order masking. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, pages 116–129, Vienna, Austria, October 24–28, 2016. ACM Press.
6. Gilles Barthe, François Dupressoir, Sebastian Faust, Benjamin Grégoire, François-Xavier Standaert, and Pierre-Yves Strub. Parallel implementations of masking schemes and the bounded moment leakage model. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 535–566, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.
7. 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, *Cryptographic Hardware and Embedded Systems – CHES 2016*, volume 9813 of *Lecture Notes in Computer Science*, pages 23–39, Santa Barbara, CA, USA, August 17–19, 2016. Springer, Heidelberg, Germany.
8. Alberto Battistello, Jean-Sebastien Coron, Emmanuel Prouff, and Rina Zeitoun. Horizontal side-channel attacks and countermeasures on the ISW masking scheme. Cryptology ePrint Archive, Report 2016/540, 2016. `https://eprint.iacr.org/2016/540`.
9. 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, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 616–648, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.
10. Sonia Belaïd, Fabrice Benhamouda, Alain Passelègue, Emmanuel Prouff, Adrian Thillard, and Damien Vergnaud. Private multiplication over finite fields. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology –*

CRYPTO 2017, Part III, volume 10403 of *Lecture Notes in Computer Science*, pages 397–426, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

11. Sonia Belaïd, Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Abdul Rahman Taleb. Random probing security: Verification, composition, expansion and new constructions. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 339–368, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Heidelberg, Germany.
12. Sonia Belaïd, Dahmun Goudarzi, and Matthieu Rivain. Tight private circuits: Achieving probing security with the least refreshing. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part II*, volume 11273 of *Lecture Notes in Computer Science*, pages 343–372, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.
13. Sonia Belaïd, Darius Mercadier, Matthieu Rivain, and Abdul Rahman Taleb. Iron-mask: Versatile verification of masking security. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 142–160. IEEE, 2022.
14. Sonia Belaïd, Darius Mercadier, Matthieu Rivain, and Abdul Rahman Taleb. Iron-Mask: Versatile verification of masking security. In *2022 IEEE Symposium on Security and Privacy*, pages 142–160, San Francisco, CA, USA, May 22–26, 2022. IEEE Computer Society Press.
15. Roderick Bloem, Hannes Groß, Rinat Iusupov, Bettina Könighofer, Stefan Mangard, and Johannes Winter. Formal verification of masked hardware implementations in the presence of glitches. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 321–353, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.
16. Nicolas Bordes and Pierre Karpman. Fast verification of masking schemes in characteristic two. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 283–312, Zagreb, Croatia, October 17–21, 2021. Springer, Heidelberg, Germany.
17. Gaëtan Cassiers, Sebastian Faust, Maximilian Orlt, and François-Xavier Standaert. Towards tight random probing security. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 185–214, Virtual Event, August 16–20, 2021. Springer, Heidelberg, Germany.
18. Gaëtan Cassiers, Benjamin Grégoire, Itamar Levi, and François-Xavier Standaert. Hardware private circuits: From trivial composition to full verification. *IEEE Trans. Computers*, 70(10):1677–1690, 2021.
19. Gaëtan Cassiers and François-Xavier Standaert. Trivially and efficiently composing masked gadgets with probe isolating non-interference. *IEEE Trans. Inf. Forensics Secur.*, 15:2542–2555, 2020.
20. Gaëtan Cassiers and François-Xavier Standaert. Provably secure hardware masking in the transition- and glitch-robust probing model: Better safe than sorry. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(2):136–158, 2021. `https://tches.iacr.org/index.php/TCHES/article/view/8790`.
21. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener,

editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany.

22. Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Thomas Roche. Higher-order side channel security and mask refreshing. In Shiho Moriai, editor, *Fast Software Encryption – FSE 2013*, volume 8424 of *Lecture Notes in Computer Science*, pages 410–424, Singapore, March 11–13, 2014. Springer, Heidelberg, Germany.
23. Jean-Sébastien Coron and Lorenzo Spignoli. Secure wire shuffling in the probing model. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 215–244, Virtual Event, August 16–20, 2021. Springer, Heidelberg, Germany.
24. Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying leakage models: From probing attacks to noisy leakage. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 423–440, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.
25. Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES'99*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172, Worcester, Massachusetts, USA, August 12–13, 1999. Springer, Heidelberg, Germany.
26. Dahmun Goudarzi, Antoine Joux, and Matthieu Rivain. How to securely compute with noisy leakage in quasilinear complexity. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part II*, volume 11273 of *Lecture Notes in Computer Science*, pages 547–574, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.
27. Dahmun Goudarzi, Thomas Prest, Matthieu Rivain, and Damien Vergnaud. Probing security through input-output separation and revisited quasilinear masking. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(3):599–640, 2021. `https://tches.iacr.org/index.php/TCHES/article/view/8987`.
28. Dahmun Goudarzi, Thomas Prest, Matthieu Rivain, and Damien Vergnaud. Probing security through input-output separation and revisited quasilinear masking. Cryptology ePrint Archive, Report 2022/045, 2022. `https://eprint.iacr.org/2022/045`.
29. Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Heidelberg, Germany.
30. David Knichel, Pascal Sasdrich, and Amir Moradi. SILVER - statistical independence and leakage verification. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 787–816, Daejeon, South Korea, December 7–11, 2020. Springer, Heidelberg, Germany.
31. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany.
32. Axel Mathieu-Mahias. Securisation of implementations of cryptographic algorithms in the context of embedded systems, 2021.

33. Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 278–296, Cambridge, MA, USA, February 19–21, 2004. Springer, Heidelberg, Germany.
34. Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 142–159, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany.
35. Matthieu Rivain and Emmanuel Prouff. Provably secure higher-order masking of AES. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems – CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 413–427, Santa Barbara, CA, USA, August 17–20, 2010. Springer, Heidelberg, Germany.

A Proofs of Section 3

A.1 Generalized security notions

Definition 15 (ℓ-Input IOS). *Let G be an (n-share, ℓ-to-1) gadget. G is said* t-input-output separative (t-IOS)*, if it is uniform and if there exists a (two-stage) simulator* $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that for every admissible tuple* $(\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell}, \overrightarrow{y})$ *for G and for every set of wires W of G with $|W| \leq t$, we have*

1. $\mathsf{Sim}_1(W) = (I_1, \ldots, I_\ell, J)$ *where* $I_1, \ldots, I_\ell, J \subseteq [1:n]$*, with* $|I_1| \leq |W|, \ldots, |I_\ell| \leq |W|$ *and* $|J| \leq |W|$*, and*
2. $\mathsf{Sim}_2(W, \overrightarrow{x_1}|_{I_1}, \ldots, \overrightarrow{x_\ell}|_{I_\ell}, \overrightarrow{y}|_J) \overset{\text{id}}{=} \mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell}), \overrightarrow{y})$.

Definition 16 (ℓ-Input Balanced IOS). *Let G be an (n-share, ℓ-to-1) gadget. G is said balanced* t-input-output separative (balanced t-IOS)*, if it is uniform and if there exists a (two-stage) simulator* $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that for every admissible tuple* $(\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell}, \overrightarrow{y})$ *for G and for every set of wires W of G with $|W| \leq t$, we have*

1. $\mathsf{Sim}_1(W) = (I_1, \ldots, I_\ell)$ *where* $I_1, \ldots, I_\ell \subseteq [1:n]$*, with* $|I_1| \leq |W|, \ldots, |I_\ell| \leq |W|$ *and*
2. $\mathsf{Sim}_2(W, \overrightarrow{x_1}|_{I_1}, \ldots, \overrightarrow{x_\ell}|_{I_\ell}, \overrightarrow{y}|_{I_1 \cap \ldots \cap I_\ell}) \overset{\text{id}}{=} \mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell}), \overrightarrow{y})$.

A gadget is simply said to be balanced IOS *if it is* balanced $(n-1)$-IOS.

Definition 17 (ℓ-Input Free SNI). *Let G be an (n-share, ℓ-to-1) gadget. G is said* free t-SNI*, if for every set W of internal wires of G such that $|W| \leq t$, there exists a (two-stage) simulator* $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that for every input* $(\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell})$*, we have*

1. $\mathsf{Sim}_1(W) = (I_1, \ldots, I_\ell)$ *where* $I_1, \ldots, I_\ell \subseteq [1:n]$*, with* $|I_1| \leq |W|, \ldots, |I_\ell| \leq |W|$*,*
2. $\mathsf{Sim}_2(W, \overrightarrow{x_1}|_{I_1}, \ldots, \overrightarrow{x_\ell}|_{I_\ell}) \overset{\text{id}}{=} \big(\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell})), \overrightarrow{y}|_{I_1 \cap \ldots \cap I_\ell}\big)$,

where $\overrightarrow{y} = G(\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell})$*, and for every set* $O \subsetneq [1:n] \backslash (I_1 \cap \ldots \cap I_\ell)$*,* $\overrightarrow{y}|_O$ *is uniformly and independently distributed, conditioned on* $\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell}))$ *and* $\overrightarrow{y}|_{I_1 \cap \ldots \cap I_\ell}$*. A gadget is simply said to be* free SNI *if it is* free $(n-1)$-SNI.

Definition 18 (ℓ-Input Unbalanced Free SNI). *Let* G *be an (*n*-share,* ℓ*-to-*1*) gadget.* G *is said* unbalanced free t-SNI*, if for every set* W *of internal wires of* G *such that* $|W| \leq t$*, there exists a (two-stage) simulator* $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that for every input* $(\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell})$*, we have*

1. $\mathsf{Sim}_1(W) = (I_1, \ldots, I_\ell, J)$ *where* $I_1, \ldots, I_\ell, J \subseteq [1:n]$*, with* $|I_1| \leq |W|, \ldots, |I_\ell| \leq |W|$ *and* $|J| \leq |W|$*, and*
2. $\mathsf{Sim}_2(W, \overrightarrow{x_1}|_{I_1}, \ldots, \overrightarrow{x_\ell}|_{I_\ell}) \overset{\text{id}}{=} \big(\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell})), \overrightarrow{y}|_J\big)$*,*

where $\overrightarrow{y} = G(\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell})$*, and for every set* $O \subsetneq [1:n] \backslash J$*,* $\overrightarrow{y}|_O$ *is uniformly and independently distributed, conditioned on* $\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \ldots, \overrightarrow{x_\ell}))$ *and* $\overrightarrow{y}|_J$*. A gadget is simply said to be* unbalanced free SNI *if it is* unbalanced free $(n-1)$-SNI.

A.2 Example of an unbalanced Free t-SNI (or equivalently t-IOS) gadget but not free t-SNI (or equivalently balanced t-IOS)

Consider the following 2-share refresh gadget with a single random value r:

$$c_1 = a_2 + r$$
$$c_2 = a_1 - r$$

IronMask demonstrates that the above gadget is not free 1-SNI. As a counterexample, it exhibits the set formed of a single probe $W = \{r\}$. Indeed, we only have three choices to fix the set I for the free 1-SNI property: $I = \emptyset$, $I = \{1\}$ or $I = \{2\}$.
In the case where we choose $I = \emptyset$, then for any $O \subsetneq \{1, 2\}$, the output shares $c_{|O}$ must be independent and uniform conditioned on W. This is clearly not the case, since neither c_1 nor c_2 is independent of the probe r.
In the case where we choose $I = \{1\}$, we need to be perfectly able to simulate the probe r and the output share c_1 using I. This is not the case, since to perfectly simulate the output share c_1, we need the input share a_2.
Finally, in the case where we choose $I = \{2\}$, we need to be perfectly able to simulate the probe r and the output share c_2 using I. This is not the case, since to perfectly simulate the output share c_2, we need the input share a_1.
Hence, for the probe $W = \{r\}$, there is no set I of size at most 1, which satisfies the free 1-SNI property. The gadget is not free 1-SNI.

Meanwhile, we can check that the above gadget is unbalanced free 1-SNI. Specifically, we consider three different cases depending on the probed variables among r, a_1 and a_2 for the unbalanced free 1-SNI property.
When $W = \emptyset$, then we can fix $I = J = \emptyset$. We can verify that for any $O \subsetneq \{1, 2\} \backslash J$, the corresponding output share is independent and uniform conditioned on W, which satisfies the property.

When $W = \{r\}$ or $W = \{-r\}$ or $W = \{a_1\}$, we fix $I = \{1\}$ and $J = \{2\}$. Using the input share a_1, we can perfectly simulate the probe in W and the output share $c_2 = a_1 - r$. In addition, since $\{1, 2\} \setminus J = \{1\}$, then we do not need to consider any output shares for the independence property (because $O \subsetneq \{1, 2\} \setminus J$ so $O = \emptyset$). This satisfies the unbalanced free 1-SNI property.
Finally, when $W = \{a_2\}$, we fix $I = \{2\}$ and $J = \{1\}$. Using the input share a_2, we can perfectly simulate the probe in W and the output share $c_1 = a_2 + r$. In addition, since $\{1, 2\} \setminus J = \{2\}$, then we do not need to consider any output shares for the independence property (because $O \subsetneq \{1, 2\} \setminus J$ so $O = \emptyset$). This satisfies the unbalanced free 1-SNI property.

We hence easily proved that the depicted gadget is unbalanced free 1-SNI, while it is not free 1-SNI.

A.3 Example of a *t*-SNI gadget but not unbalanced free *t*-SNI (or equivalently *t*-IOS)

Consider the following 4-share linear refresh gadget introduced in [35, Algorithm 4]:

$$\begin{aligned}
&c_1 = a_1 + r_1 \\
&c_2 = a_2 + r_2 \\
&c_3 = a_3 + r_3 \\
&c_4 = (((a_4 - r_1) - r_2) - r_3)
\end{aligned}$$

IronMask and maskVerif verification tools both demonstrate that the above gadget is 1-SNI, but not 2-SNI. However, the gadget is not unbalanced free 1-SNI. In fact, consider as a counterexample the single probe

$$w = a_4 - r_1 - r_2 \ .$$

It is clear that no input shares are needed to perfectly simulate the probe w. However, we can easily check that the output shares c_1 and c_2 are not independent and uniform conditioned on w. In fact, we can check that

$$c_1 + c_2 + w = a_1 + a_2 + a_4 \ .$$

Hence, in order to satisfy the unbalanced free 1-SNI, we have to consider at least $J = \{1, 2\}$, and consequently $I = \{1, 2, 4\}$ to be able to perfectly simulate w and the output shares c_1 and c_2 . Since $|I| > 1$ and $|J| > 1$, then this gadget cannot satisfy the unbalanced free 1-SNI.

We hence proved that the depicted gadget is 1-SNI but not unbalanced free 1-SNI.

A.4 Proof of Theorem 1 and Theorem 2

We prove the results for 2-to-1 gadgets. The 1-to-1 easily follow. We start with the proof of Theorem 1.

We start with the left-to-right implication. Namely, we assume that G is free t-SNI. First, the uniformity of G is straightforward from the free t-SNI property for $W = \emptyset$. In that case, any strict subset of the output is uniformly and independently distributed. Then, we denote $(\mathsf{Sim}_1^{\text{f-sni}}, \mathsf{Sim}_2^{\text{f-sni}})$ the corresponding two-stage simulator. Let $(\overrightarrow{x_1}, \overrightarrow{x_2}, \overrightarrow{y})$ be an admissible triple for G and W a set of wires such that $|W| \leq t$. We define the first simulator for the balanced t-IOS property as follows:

$$\mathsf{Sim}_1^{\text{b-ios}}(W) = \mathsf{Sim}_1^{\text{f-sni}}(W) = (I_1, I_2)$$

and we have $I_1, I_2 \subseteq [1:n]$, with $|I_1| \leq |W|$, $|I_2| \leq |W|$ as expected, and we define $J = I_1 \cap I_2$. Then, for the second simulator, we run the second free t-SNI simulator on the inputs $\mathsf{Sim}_2^{\text{f-sni}}(\overrightarrow{x_1}, \overrightarrow{x_2})$ until its second output is equal to $\overrightarrow{y}|_J$ as previously defined (observe that we need to do this since the output $\overrightarrow{y}$ is already fixed for the balanced IOS property, in the admissible triple $(\overrightarrow{x_1}, \overrightarrow{x_2}, \overrightarrow{y})$). Note that this step is certainly inefficient but it ends with probability one (since G is uniform and $(\overrightarrow{x_1}, \overrightarrow{x_2}, \overrightarrow{y})$ is admissible for G). Once done, we define the second simulator $\mathsf{Sim}_2^{\text{b-ios}}$ to output $\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \overrightarrow{x_2}))$ as simulated by the second free t-SNI simulator for the correct output. This corresponds to a perfect simulation of $\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \overrightarrow{x_2}), \overrightarrow{y})$ which shows the correctness of $\mathsf{Sim}_2^{\text{b-ios}}$. Hence G is balanced t-IOS.

We continue with the right-to-left implication. Now we assume that G is balanced t-IOS. Let W be a set of internal wires such that $|W| \leq t$. We define the simulators of the free t-SNI property. The first simulator $\mathsf{Sim}_1^{\text{f-sni}}$ is defined like the one of the balanced t-IOS property. We thus have that $\mathsf{Sim}_1^{\text{f-sni}}(W) = (I_1, I_2)$ where $I_1, I_2 \subseteq [1:n]$, with $|I_1|, |I_2| \leq |W|$. Since G is uniform and $|J| < n$, any sharing chunk $\overrightarrow{y}|_J$ generated uniformly at random forms an admissible triple for G with $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$. We thus generate such a sharing chunk $\overrightarrow{y}|_J$ and we use the second simulator of the balanced IOS property $\mathsf{Sim}_2^{\text{b-ios}}$ to build the second simulator for the free t-SNI property as follows:

$$\mathsf{Sim}_2^{\text{f-sni}}(\overrightarrow{x_1}|_{I_1}, \overrightarrow{x_2}|_{I_2}) \leftarrow \mathsf{Sim}_2^{\text{b-ios}}(\overrightarrow{x_1}|_{I_1}, \overrightarrow{x_2}|_{I_2}, \overrightarrow{y}|_J)$$

which gives us $\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \overrightarrow{x_2}))$. The output sharing $\overrightarrow{y}|_J$ is trivially simulated since it is already defined. Now, it remains to prove that for every set $O \subsetneq [1:n] \backslash (J)$, $\overrightarrow{y}|_O$ is uniformly and independently distributed, conditioned on $\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \overrightarrow{x_2}))$ and $\overrightarrow{y}|_J$. Assume this is not the case, then $\mathsf{Sim}_2^{\text{b-ios}}$ would necessarily require more shares from $\overrightarrow{y}|_J$ to output a perfect simulation of $\mathsf{AssignWires}(G, W, (\overrightarrow{x_1}, \overrightarrow{x_2}), \overrightarrow{y})$, which leads to a contradiction. $\square$

Finally, the proof of Theorem 2 is identical to the above proof, except for letting Sim_1 output (I_1, I_2, J) instead of using $J = I_1 \cap I_2$.

A.5 Proof of Theorem 3

We prove the result for 2-to-1 gadgets. The 1-to-1 easily follows.

We assume that G is t-IOS. Let W be a set of internal wires such that $|W| \leq t_i$ and J be a set of output share indices such that $|J| \leq t_o$ with $t_i + t_o \leq t$. From the t-IOS property, there exists $\mathsf{Sim}_1^{\text{ios}}$ such that $\mathsf{Sim}_1^{\text{ios}}(W) = (I_1, I_2, J^{\text{ios}})$. We define the first simulator for the SNI property as follows:

$$\mathsf{Sim}_1^{\text{sni}}(W) = (I_1, I_2).$$

We thus have that $|I_1|, |I_2| \leq |W|$. We then generate uniformly at random a sharing chunk $\overrightarrow{y}|_{J \cup J^{\text{ios}}}$. Since G is uniform and $|J \cup J^{\text{ios}}| \leq |W| + t_o \leq t < n$, $\overrightarrow{y}|_{J \cup J^{\text{ios}}}$ can be seen as a part of a $\overrightarrow{y}$ that forms an admissible triple with $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$. In other words, since G is uniform, then the subset $J \cup J^{\text{ios}}$ of at most $n-1$ shares of the output $\overrightarrow{y}$ can be seen as a set of $n-1$ uniform and mutually independent random values that do not depend on the inputs $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$. Hence, we can generate all shares in $\overrightarrow{y}|_{J \cup J^{\text{ios}}}$ uniformly at random, and they would still form an admissible triple with $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$.

Then, using the second simulator of the t-IOS property with this $\overrightarrow{y}$, we obtain a perfect simulation of the probes in W and we also have $\overrightarrow{y}|_J$, which together form the output of the second SNI simulator. □

A.6 Proof of Proposition 1

Let G be an n-share uniform ZE-refresh gadget. We will prove that the inner gadget $\mathsf{ZeroEnc}()$ is t-free if and only if G is free t-SNI. For this, let us denote $\overrightarrow{z}$ the output sharing of $\mathsf{ZeroEnc}()$, $\overrightarrow{y}$ the output sharing of G and $\overrightarrow{x}$ its input sharing.

We start with the left-to-right implication. Namely, let us first suppose that $\mathsf{ZeroEnc}$ is t-free. Let W be a set of probes on the gadget G such that $|W| \leq t$. We can split W into disjoint sets W_1 and W_2, where W_1 contains probes (internal and output) on $\mathsf{ZeroEnc}$, and W_2 contains probes on the input shares x_i, in other words $W_2 \subseteq \{x_1, \ldots, x_n\}$. Since $\mathsf{ZeroEnc}$ is t-free, there exists a set J such that $|J| \leq |W_1|$ and for any $O' \subsetneq [1:n] \setminus J$, the shares $\overrightarrow{z}|_{O'}$ are uniformly distributed and mutually independent from the probes in W_1 and $\overrightarrow{z}|_J$.
Next, we construct the two-stage simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ of G for free t-SNI. Namely, Sim_1 outputs the set $I = J \cup \{i \mid x_i \in W_2\}$. Observe that $|I| \leq |J| + |W_2| \leq |W|$. Now we show how Sim_2 perfectly simulates the probes in W and the output shares $\overrightarrow{y}|_I$. All probes in W_1 can be simply perfectly simulated by generating random values and performing the same operations as in $\mathsf{ZeroEnc}$. Then, the probes in W_2 which are input shares can be perfectly simulated using the input shares $\overrightarrow{x}|_I$ by construction of the set I. In addition, the output shares $\overrightarrow{y}|_I$ are also perfectly simulated by observing that each y_i for $i \in I$ is equal to $y_i = x_i + z_i$ where x_i is an input share and z_i is an output of $\mathsf{ZeroEnc}$. This proves the simulation part of the free t-SNI property.
Next, let $O \subsetneq [1:n] \setminus I$. We have that $O \subsetneq [1:n] \setminus I \subseteq [1:n] \setminus J$ since $J \subseteq I$. Since for any $O' \subsetneq [1:n] \setminus J$, the outputs of the internal block $\overrightarrow{z}|_{O'}$ are uniformly distributed and mutually independent from W_1 and $\overrightarrow{z}|_J$, then we also have that $\overrightarrow{z}|_O$ are uniformly distributed mutually independent from $W = W_1 \cup W_2$ and $\overrightarrow{z}|_I$ because $W_1 \subseteq W$ and $J \subseteq I$. Then, we get that $\overrightarrow{y}|_O$ are also mutually independent from W and $\overrightarrow{y}|_I$. This concludes the first part of the proof.

We continue with the right-to-left implication. Now we suppose that G is free t-SNI. Let W be a set of probes on $\mathsf{ZeroEnc}$ such that $|W| \leq t$. We will prove that $\mathsf{ZeroEnc}$ is t-free. For this, we can use the simulator of G for free t-SNI. Namely, we provide Sim_1 with the set of probes W. Notice that $\{x_1, \dots, x_n\} \cap W = \emptyset$. The simulator then outputs a set I such that $|I| \leq |W| \leq t$. From the free t-SNI property of G, we know that for any $O \subsetneq [1 : n] \setminus I$, the output shares $\overrightarrow{y}|_O$ are uniformly distributed and mutually independent from W and $\overrightarrow{y}|_I$. Hence, for the t-free property of $\mathsf{ZeroEnc}$, we can fix $J = I$. In fact, since for each $i \in [1 : n]$, we have $y_i = z_i + x_i$ and x_i is an input share, then we can check that for any $O \subsetneq [1 : n] \setminus J$, the output shares $\overrightarrow{z}|_O = \overrightarrow{y}|_O - \overrightarrow{x}|_O$ of $\mathsf{ZeroEnc}$ are uniformly distributed and mutually independent from W and $\overrightarrow{z}|_J = \overrightarrow{y}|_J - \overrightarrow{x}|_J$. This concludes the second part of the proof, which concludes the proof of the Lemma. $\square$

B Proofs of Section 4

B.1 Proofs of Proposition 2 and Proposition 3

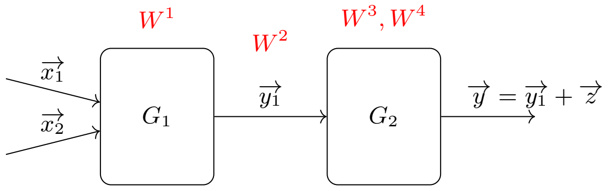

Fig. 6. Construction from Proposition 2.

We will prove Proposition 2 and Proposition 3. Namely, let G_1 be an (n-share, 2-to-1) gadget and G_2 be a ZE-refresh gadget with $G = G_2 \circ G_1$ as shown in Figure 6. Let $t > t_1 \geq 1$ and $t_2 = t - t_1$. We will show that

- if G_1 is $(t_1 + t_2)$-SNI and free t_1-SNI, and G_2 is free $(t_2 - 1)$-SNI, then G is free $(t_1 + t_2)$-SNI (*i.e.* result of Proposition 2), and
- if G_1 is $(t_1 + t_2)$-NI and t_1-IOS, and G_2 is free $(t_2 - 1)$-SNI, then G is $(t_1 + t_2)$-IOS (*i.e.* result of Proposition 3).

The proof follows the same path for both cases. We point out the main differences within the proof. In order to prove the result of Proposition 3, we will use the unbalanced free SNI property instead of IOS, which is essentially equivalent to IOS (by Theorem 2) but makes the proof more convenient.

We denote $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$ the input sharings of G, and $\overrightarrow{y}$ its output sharing (which is the output sharing of G_2). We also denote $\overrightarrow{y_1}$ to be the output sharing of G_1, such that $\forall i \in [1:n], y_i = y_{1,i} + z_i$, where $\overrightarrow{z}$ is the output sharing of ZeroEnc internal block of G_2. We consider that G has two input sharings. The proof in the case where G has only one input sharing is similar. Figure 6 shows the construction of the gadget.

Let W be a set of leaking wires such that $|W| \leq t_1 + t_2$. We will split these internal probes into four disjoint sets $W = W_1 \cup W_2 \cup W_3 \cup W_4$ such that:

1. W_1 contains internal probes to gadget G_1
2. W_2 contains probes on the output of G_1, *i.e.* probes on $y_{1,i}$ for $i \in [1:n]$.
3. W_3 contains internal probes on the `ZeroEnc` component of gadget G_2.
4. W_4 contains output probes on the `ZeroEnc` component of gadget G_2, *i.e.* probes on z_i for $i \in [1:n]$.

We consider two cases.

Case 1. Let us first consider the case where $|W_1| \leq t_1$. In this case, we have at most t_1 probes on G_1. Now we differentiate the cases for Proposition 2 and Proposition 3.

– In the case of Proposition 2, G_1 is free t_1-SNI. So we can construct sets I'_1 and I'_2 on inputs $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$ respectively, such that $|I'_1| \leq |W_1|$ and $|I'_2| \leq |W_1|$ and the probes in W_1 as well as the shares $\overrightarrow{y_1}|_{I'_1 \cap I'_2}$ can be perfectly simulated using the input shares $\overrightarrow{x_1}|_{I'_1}$ and $\overrightarrow{x_2}|_{I'_2}$. In addition, for any $O_1 \subsetneq [1:n] \setminus (I'_1 \cap I'_2)$, the shares $\overrightarrow{y_1}|_{O_1}$ are uniformly distributed and mutually independent from $\overrightarrow{y_1}|_{I'_1 \cap I'_2}$ and the probes in W_1.

  Since, we can also have probes on the shares $\overrightarrow{y_1}$ in W_2, for each such probe $y_{1,i}$ for $i \in [1:n]$, we add i to the already constructed sets of the previous step I'_1 and I'_2. Let $K = \{i \mid y_{1,i} \in W_2\}$. We have $I_1 = I'_1 \cup K$, $I_2 = I'_2 \cup K$, $|I_1| \leq |W|$ and $|I_2| \leq |W|$. Thanks to the free $(n-t_1-1)$-SNI property of G_1, we know that the shares $\overrightarrow{y_1}|_{I_1 \cap I_2} = \overrightarrow{y_1}|_{(I'_1 \cap I'_2) \cup K}$ can be perfectly simulated since:
  - the shares $\overrightarrow{y_1}|_{I'_1 \cap I'_2}$ are perfectly simulated from the input shares indexed in I'_1 and I'_2.
  - the shares $\overrightarrow{y_1}|_{K \setminus (I'_1 \cap I'_2)}$ are also perfectly simulated. Indeed, the output shares $\overrightarrow{y_1}|_{O_1}$ are uniformly distributed and mutually independent from $\overrightarrow{y_1}|_{I'_1 \cap I'_2}$ and the rest of the probes, for any $O_1 \subsetneq [1:n] \setminus (I'_1 \cap I'_2)$. In particular, they can be simulated without any input share.

  Notice that we also still have the property that for any $O_1 \subsetneq [1:n] \setminus (I_1 \cap I_2)$, the shares $\overrightarrow{y_1}|_{O_1}$ are uniformly distributed and mutually independent from $\overrightarrow{y_1}|_{I_1 \cap I_2}$ and the probes in W_1, since

$$[1:n] \setminus (I_1 \cap I_2) \subseteq [1:n] \setminus (I'_1 \cap I'_2) \ .$$

  Probes in $W_3 \cup W_4$ can then be trivially simulated without the need for any input shares (recall that G_2 performs a refreshing of a zero encoding), which proves that we can perfectly simulate all probes in $W_1 \cup W_2 \cup W_3 \cup W_4 = W$.

  As for the output shares $\overrightarrow{y}|_{I_1 \cap I_2}$, notice that for $i \in [1:n]$, we have $y_i = y_{1,i} + z_i$. Since we already know that we can perfectly simulate the shares $y_{1,i}$ for $i \in I_1 \cap I_2$ thanks to the above arguments, then we can also perfectly simulate each such y_i by perfectly simulating the corresponding $y_{1,i}$ and z_i.

  Finally, let $O \subsetneq [1:n] \setminus (I_1 \cap I_2)$. Since the shares $\overrightarrow{y_1}|_O$ are independent and uniform conditioned on W_1 and $\overrightarrow{y_1}|_{I_1 \cap I_2}$, then it follows that the output shares $\overrightarrow{y}|_O$ are uniformly distributed and mutually independent from W and $\overrightarrow{y}|_{I_1 \cap I_2}$ (note that the probes in $W_2 \subseteq W$ of the form $y_{1,i}$ are already perfectly simulated since they are indexed in $I_1 \cap I_2$). This shows that G is free $t_1 + t_2$-SNI in the case where $|W_1| \leq t_1$ for Proposition 2.

– In the case of Proposition 3, G_1 is unbalanced free t_1-SNI. So we can construct sets I'_1 and I'_2 and J' on $\overrightarrow{x_1}$, $\overrightarrow{x_2}$, and $\overrightarrow{y}$ respectively, such that $|I'_1| \leq |W_1|$, $|I'_2| \leq |W_1|$, and $|J'| \leq |W_1|$ and the probes in W_1 as well as the shares $\overrightarrow{y_1}|_{J'}$ can be perfectly simulated using the input shares $\overrightarrow{x_1}|_{I'_1}$ and $\overrightarrow{x_2}|_{I'_2}$. In addition, for any $O_1 \subsetneq [1:n] \setminus J'$, the shares $\overrightarrow{y_1}|_{O_1}$ are uniformly distributed and mutually independent from $\overrightarrow{y_1}|_{J'}$ and the probes in W_1.

  Similarly as before, we construct $I_1 = I'_1 \cup K$, $I_2 = I'_2 \cup K$, $J = J' \cup K$ with the set K as defined above. Note that we still have $|I_1| \leq |W|$, $|I_2| \leq |W|$ and $|J| \leq |W|$, and we can prove in the same way as earlier that the shares $\overrightarrow{y_1}|_J = \overrightarrow{y_1}|_{J' \cup K}$ can be perfectly simulated from $\overrightarrow{x_1}|_{I_1}$ and $\overrightarrow{x_2}|_{I_2}$. Finally, all of the probes in W are also simulated as before and for any $O \subsetneq [1:n] \setminus J$, the output shares $\overrightarrow{y}|_O$ are uniformly distributed and mutually independent from W and $\overrightarrow{y}|_J$. This shows that G is unbalanced free (t_1+t_2)-SNI in the case where $|W_1| \leq t_1$ for Proposition 3.

Case 2. Now let us consider the other case $|W_1| \geq t_1 + 1$. Since $|W| \leq t_1 + t_2$, then we have $|W_3| \leq t_2 - 1$. From the (t_2-1)-free property of G_2, there exists a set J', such that $|J'| \leq |W_3|$ and for any $O' \subsetneq [1:n] \setminus J'$, the shares $\overrightarrow{z}|_{O'}$ are uniformly distributed and mutually independent from the internal probes W_3 and $\overrightarrow{z}|_{J'}$. We will use the shares of $\overrightarrow{z}$ in order to prove the free (t_1+t_2)-SNI property of G. For this, we will first start by constructing a new set W'_2 of output probes on gadget G_1 as follows.

We start with $W'_2 = W_2$. Then, for every $i \in J'$, we add $y_{1,i}$ as a probe to W'_2. And for every $z_j \in W_4$ for $j \in [1:n]$, we add $y_{1,j}$ to W'_2. Now, instead of perfectly simulating the probes W_1 and W_2 on G_1, we will simulate W_1 and W'_2, which is stronger since $W_2 \subseteq W'_2$.

For the rest of the proof, we differentiate the cases where G_1 is (t_1+t_2)-SNI for Proposition 2, or (t_1+t_2)-NI for Proposition 3. But first, observe that $|W| \leq t_1 + t_2$, in particular $|W_1| + |W'_2| \leq t_1 + t_2$ by construction of the set W'_2 ($|W'_2| \leq |W_2| + |W_3| + |W_4|$).

– In the case of Proposition 2, G_1 is (t_1+t_2)-SNI, we know that we can perfectly simulate probes in W_1 and W'_2 from sets of input shares I'_1 and I'_2 such that $|I'_1| \leq |W_1|$ and $|I'_2| \leq |W_1|$.

  Then, we consider the following sets of input shares on the gadget G:

$$I_1 = I'_1 \cup J' \cup \{i \mid z_i \in W_4\}$$

  and

$$I_2 = I'_2 \cup J' \cup \{i \mid z_i \in W_4\}$$

  Notice that $|I_1| \leq |I'_1| + |J'| + |W_4| \leq |W_1| + |W_3| + |W_4| \leq |W|$ and $|I_2| \leq |I'_2| + |J'| + |W_4| \leq |W_1| + |W_3| + |W_4| \leq |W|$.

Next, we will prove how to perfectly simulate all probes in W and the output shares $\overrightarrow{y}|_{I_1 \cap I_2}$ from the input shares indexed in I_1 and I_2. We will also prove that for any $O \subsetneq [1:n] \backslash (I_1 \cap I_2)$, $\overrightarrow{y}|_O$ are uniformly distributed and mutually independent from the probes in W and $\overrightarrow{y}|_{I_1 \cap I_2}$.

First, probes in $W_3 \cup W_4$ can be simulated without the need for any input share. Also, probes in $W_1 \cup W_2$ can be perfectly simulated from input shares indexed in I_1 and I_2 since they are perfectly simulated using the input shares indexed in I'_1 and I'_2 from the SNI property of G_1 and the fact that $I'_1 \subseteq I_1$ and $I'_2 \subseteq I_2$. Next, let us consider the output shares $\overrightarrow{y}|_{I_1 \cap I_2}$. Let $i \in I_1 \cap I_2$ and $y_i = y_{1,i} + z_i$. Then, we have two possibilities:

1. $i \in J' \cup \{i \mid z_i \in W_4\}$. In this case, we can perfectly simulate $y_{1,i}$ using input shares in I_1 and I_2, by construction of the set of output probes W'_2 on G_1. Hence, we can perfectly simulate the output share y_i (we do not require any input share to simulate the second term z_i).
2. $i \notin J' \cup \{i \mid z_i \in W_4\}$. In this case, we know that the share z_i is independent and uniform conditioned on the internal probes and the other output shares by the property of gadget G_2 (as long as at least another z_j does not appear in the probes, which is the case by assumption). Hence, the output share $y_i = y_{1,i} + z_i$ is also independent and uniform and can be simulated without the need for any input share.

Thus we proved that all probed intermediate variables on gadget G and the output shares $y_{|I_1 \cap I_2}$ can be perfectly simulated from the sets of input shares I_1 and I_2.

Finally, let $O \subsetneq [1:n] \backslash (I_1 \cap I_2)$. Let $i \in O$. Since in particular $i \notin J' \cup \{i \mid z_i \in W_4\}$, then we can again use the property of gadget G_2 in order to prove that $\overrightarrow{y}|_O$ are uniformly distributed and mutually independent from the internal probes and $\overrightarrow{y}|_{I_1 \cap I_2}$, which comes from the fact that $\overrightarrow{z}|_O$ are independent and uniform and that $\overrightarrow{y} = \overrightarrow{y_1} + \overrightarrow{z}$.

This concludes the proof that G is free $(t_1 + t_2)$-SNI in the case where $|W_1| \geq t_1 + 1$.

– In the case of Proposition 3, G_1 is $(t_1 + t_2)$-NI, we know that we can perfectly simulate probes in W_1 and W'_2 from sets of input shares I'_1 and I'_2 such that $|I'_1| \leq |W_1| + |W'_2|$ and $|I'_2| \leq |W_1| + |W'_2|$.

  Then, we consider the following sets of input shares on the gadget G:

$$I_1 = I'_1, \qquad I_2 = I'_2$$

  and the following set of output shares

$$J = J' \cup \{i \mid z_i \in W_4\}$$

  As earlier, we have $|I_1| \leq |W|$ and $|I_2| \leq |W|$. In addition, $|J| \leq |J'| + |W_4| \leq |W_3| + |W_4| \leq |W|$.

  Next, can prove in a similar way as earlier that we can perfectly simulate all probes in W and the output shares $\overrightarrow{y}|_J$ from the input shares indexed in I_1

and I_2, and that for any $O \subsetneq [1:n] \setminus J$, $\overrightarrow{y}|_O$ are uniformly distributed and mutually independent from the probes in W and $\overrightarrow{y}|_J$. This concludes the proof that G is unbalanced free (t_1+t_2)-SNI in the case where $|W_1| \geq t_1+1$.

This finally concludes the proofs of Proposition 2 and Proposition 3. $\square$

B.2 Proof of Proposition 4

We will prove Proposition 4. Namely, let G_{ZE} be an n-share ZE-refresh gadget. Let $G_{\mathsf{cp}} = (G_{\mathrm{ZE}}, G_{\mathrm{ZE}})$. We will prove that if G_{ZE} is free t-SNI, then G_{cp} is balanced t-IOS.

Let us first formally define what it means to be (balanced) IOS for a copy gadget, or more generally for a 1-to-2 gadget. This definition is a straightforward generalization of the 1-to-1 and 2-to-1 notions.

Definition 19 (Two-Output IOS). *Let $G : \overrightarrow{x} \mapsto (\overrightarrow{y_1}, \overrightarrow{y_2})$ be an (n-share, 1-to-2) gadget. G is said* t-input-output separative (t-IOS)*, if it is uniform and if there exists a (two-stage) simulator $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ such that for every admissible triple $(\overrightarrow{x}, \overrightarrow{y_1}, \overrightarrow{y_2})$ for G and for every set of wires W of G with $|W| \leq t$, we have*

1. *$\mathsf{Sim}_1(W) = (I, J_1, J_2)$ where $I, J_1, J_2 \subseteq [1:n]$, with $|I| \leq |W|$, $|J_1| \leq |W|$ and $|J_2| \leq |W|$, and*
2. *$\mathsf{Sim}_2(W, \overrightarrow{x}|_I, \overrightarrow{y_1}|_{J_1}, \overrightarrow{y_2}|_{J_2}) \overset{\mathrm{id}}{=} \mathsf{AssignWires}(G, W, \overrightarrow{x}, (\overrightarrow{y_1}, \overrightarrow{y_2}))$.*

Moreover, G is balanced t-IOS *if for every set of wires W, the sets returned by $\mathsf{Sim}_1(W)$ satisfy $I = J_1 = J_2$.*

We are now ready to prove Proposition 4. We recall that the considered copy gadget is defined as

$$G_{\mathsf{cp}} : x \mapsto (\overrightarrow{y_1}, \overrightarrow{y_2}) := \big(\overrightarrow{x} + \mathsf{ZeroEnc}(), \overrightarrow{x} + \mathsf{ZeroEnc}()\big)$$

where $\mathsf{ZeroEnc}$ is a t-free encoding of zero (or equivalently G_{ZE} is free t-SNI). Let W be a set of probes on the gadget G_{cp} such that $|W| \leq t$. We can split W into disjoint sets W_1, W_2 and W_3, where W_1 contains probes (internal and output) on the "left" $\mathsf{ZeroEnc}$, W_1 contains probes (internal and output) on the "right" $\mathsf{ZeroEnc}$, and W_3 contains probes on the input shares x_i, in other words $W_3 \subseteq \{x_1, \ldots, x_n\}$. Let us denote $\overrightarrow{z_1}$ and $\overrightarrow{z_2}$ the outputs of the two encodings of zero such that $(\overrightarrow{y_1}, \overrightarrow{y_2}) = (\overrightarrow{x} + \overrightarrow{z_1}, \overrightarrow{x} + \overrightarrow{z_2})$.

Since $\mathsf{ZeroEnc}$ is t-free, there exists a set J_1' such that $|J_1'| \leq |W_1|$ and for any $O_1 \subsetneq [1:n] \setminus J_1'$, the shares $\overrightarrow{z_1}|_{O_1}$ are uniformly distributed and mutually independent from the probes in W_1 and $\overrightarrow{z_1}|_{J_1'}$. Similarly, there exists a set J_2' such that $|J_2'| \leq |W_2|$ and for any $O_2 \subsetneq [1:n] \setminus J_2'$, the shares $\overrightarrow{z_2}|_{O_2}$ are uniformly distributed and mutually independent from the probes in W_2 and $\overrightarrow{z_2}|_{J_2'}$. Let us further denote I' the set of indices for which the input shares $\overrightarrow{x}|_{I'}$ are in W_3. We defined the first IOS simulator as

$$\mathsf{Sim}_1(W) = (I, I, I) \quad \text{with} \quad I = I' \cup J_1' \cup J_2' \ .$$

Then the simulator Sim_2 works as follows: upon receiving shares $\overrightarrow{x}|_I, \overrightarrow{y_1}|_I, \overrightarrow{y_2}|_I$, it first computes $\overrightarrow{z_1}|_I = \overrightarrow{y_1}|_I - \overrightarrow{x}|_I$ and $\overrightarrow{z_2}|_I = \overrightarrow{y_2}|_I - \overrightarrow{x}|_I$. Then it performs a simulation of the wires in W_1 which is consistent with $\overrightarrow{z_1}|_I$ and a simulation of the wires in W_2 which is consistent with $\overrightarrow{z_2}|_I$. By the free property of $\mathsf{ZeroEnc}$ these two simulations are consistent with any value of $\overrightarrow{z_1}|_{[1:n]\setminus I}$ and $\overrightarrow{z_2}|_{[1:n]\setminus I}$ (satisfying the constraint $\sum_i z_{1,i} = \sum_i z_{2,i} = 0$), namely they are consistent with any admissible value of the triple $(\overrightarrow{x}, \overrightarrow{y_1}, \overrightarrow{y_2})$ matching the input of the simulator $(\overrightarrow{x}|_I, \overrightarrow{y_1}|_I, \overrightarrow{y_2}|_I)$. The wires in W_3 are directly simulated from $\overrightarrow{x}|_I$. We thus obtain a perfect simulation of all the wires in W which is consistent with the triple $(\overrightarrow{x}, \overrightarrow{y_1}, \overrightarrow{y_2})$ which implies that G_{cp} is t-IOS. Moreover, since the sets returned by Sim_1 are all equal, the gadget is further balanced t-IOS.This concludes the proof of Proposition 4. □

B.3 Proof of Proposition 5

We prove that the ISW multiplication n-share gadget described in Algorithm 1 is free $(n-2)$-SNI.

Let W be a set of internal leaking wires such that $|W| \leq n-2$. First, observe that the leaking wires in W are of the following forms:

1. input shares a_i, b_i, product of shares $a_i \cdot b_i$.
2. partial sum $c_{i,j} = \begin{cases} a_i \cdot b_i - r_{i,1} - \cdots - r_{i,j} & \text{if } j < i \\ a_i \cdot b_i - r_{i,1} - \cdots - r_{i,i-1} + r_{i,i+1} + \cdots + r_{i,j} & \text{otherwise} \end{cases}$ such that $j \in [1:n] \setminus \{i\}$.
3. product of shares $a_i \cdot b_j$ with $i \neq j$.
4. random variable r_{ij} for $i < j$, variable $r_{ji} = a_i \cdot b_j - r_{ij} + a_j \cdot b_i$ for $j > i$.
5. variable $a_i \cdot b_j - r_{ij}$ with $i \neq j$.

We build sets I_1 and I_2 on input shares from empty sets as follows. For every wire in W of the first or second form, we add index i to I_1 and I_2. For every wire in W of the third form, we add index i to I_1 and j to I_2. For every wire in W of the fourth or fifth form, if $i \in I_1$, we add j to I_1, otherwise we add i to I_1, and if $i \in I_2$, we add j to I_2, otherwise we add i to I_2. We can see that $|I_1| \leq |W|$ and $|I_2| \leq |W|$. Let $O \subsetneq [1:n] \setminus (I_1 \cap I_2)$, and without loss of generality, we let O of maximal size, *i.e.* $|O| = n - |I_1 \cap I_2| - 1$. Let $o \in [1:n]$ such that $o \notin O \cup (I_1 \cap I_2)$. Since $|W| \leq n-2$, then we are sure that $|O| \geq 1$ and that the index o exists.

We will first start by showing that the output shares indexed in O are uniform and independent conditioned on the probes in W and the output shares indexed in $I_1 \cap I_2$ and the other output shares in O. We will show this by demonstrating that each such output share indexed in O can be masked by a random value which does not involve any of the probes, the shares $c_{|I_1 \cap I_2}$, or the other output shares in O.

Let $i \in O$. Since, $i \notin I_1 \cap I_2$, we know that there is no probe of the form $c_{i,j}$ for any $j \in [1:n]$ (*i.e.* partial sum of c_i) by construction of the sets I_1 and I_2. In addition, each c_i indexed in O is composed of $n-1$ random values, and at most one of them can enter in the expression of each other output wire c_j for $j \in [1:n] \setminus \{i\}$ or a probe in W. In particular, for the index $o \notin O \cup (I_1 \cap I_2)$ chosen above, we can have two cases:

- if $o < i$, we know that the random variable $r_{o,i}$ does not appear in any probe in W by construction of the sets I_1 and I_2 (neither $r_{o,i}$ nor a partial sum of $r_{i,o} = a_o \cdot b_i + r_{o,i} + a_i \cdot b_o$, or of the output shares c_i and c_o are observed). Hence, the random value $r_{o,i}$ only appears in the expression of c_i, so it can be used to mask its expression. Hence, c_i can be generated as a fresh random value, conditioned on the probes in W, on the output shares indexed in $I_1 \cap I_2$, and also on the other output shares in O.
- if $o > i$, we know that the random variable $r_{i,o}$ does not appear in any probe in W by construction of the sets I_1 and I_2 (neither $r_{i,o}$ nor a partial sum of $r_{o,i} = a_i \cdot b_o + r_{i,o} + a_o \cdot b_i$, or of the output shares c_i and c_o are observed). Hence, the random value $r_{i,o}$ only appears in the expression of c_i, so it can be used to mask its expression. Hence, c_i can be generated as a fresh random value, conditioned on the probes in W, on the output shares indexed in $I_1 \cap I_2$, and also on the other output shares in O.

Hence, we proved that each output share indexed in O is generated as a fresh random value independently of the probes, the output shares indexed in $I_1 \cap I_2$, and the other output shares in O. Next, we will prove how to jointly simulate the probes in W and the output shares $c_{|I_1 \cap I_2}$.

It is easy to see that probes in W of the first or third form (*i.e.* product of shares) can be perfectly simulated using the corresponding input shares in I_1 and I_2. Let us now consider probes of the fourth and fifth form. If the probe is a random value $r_{i,j}$ for $i < j$, then it can be perfectly simulated without the need for any input share. Otherwise, if the probe is of the form $a_i \cdot b_j + r_{ij}$ or $r_{ji} = a_i \cdot b_j + r_{ij} + a_j \cdot b_i$ for $j > i$ (we will denote such as probe as p), then we can have two cases:

- if $i, j \in I_1 \cap I_2$, then the probe can be perfectly simulated from the input shares in I_1 and I_2 and by generating the random value $r_{i,j}$.
- otherwise, we are sure that $i \in I_1 \cap I_2$ and $j \notin I_1 \cap I_2$, by construction of the sets I_1 and I_2 and since at least the expression p is probed. This means only one expression of the form p is probed (*i.e.* there are no probes of the form $r_{i,j}$ either). The $r_{i,j}$ random value may also appear in the output share c_j. Meanwhile, since $j \notin I_1 \cap I_2$, then either $j = o$ and $o \notin O \cup (I_1 \cap I_2)$, or $j \in O$ and the output share c_j is masked by the random value $r_{j,o}$ if $j < o$ or $r_{o,j}$ if $o < j$, which is different than $r_{i,j}$ since $i \in I_1 \cap I_2$. We can conclude that the random value $r_{i,j}$ only appears in the expression of p and can thus be used to mask it. Hence, p can be generated as a fresh random value.

It remains to show that we can perfectly simulate the probes in W of the second form and the output shares $c_{|I_1 \cap I_2}$. We will prove their simulation at the same time. Let $i \in I_1 \cap I_2$.

– if there are no probes in W of the form $c_{i,j}$ for any $j \in [1:n]$ (*i.e.* partial sum of c_i), then c_i is composed of $n-1$ random values, and at most one of them can enter in the expression of each other output wire c_j for $j \in [1:n] \setminus \{i\}$. Since we have at most $n-2$ probes ($|W| \leq n-2$), then there is at least one index $j \notin I_1 \cap I_2$ such that the random value $r_{i,j}$ (or $r_{j,i}$ if $j < i$) can be used to mask the expression of c_i (either $j = o$ and $o \notin O \cup (I_1 \cap I_2)$, or $j \in O$ and the output share c_j is masked by the random value $r_{j,o}$ if $j < o$ or $r_{o,j}$ if $o < j$, which is different than $r_{i,j}$ since $i \in I_1 \cap I_2$). We can conclude that c_i can be perfectly simulated without the need for any input share.

– if there is at least partial sum of c_i observed through W, then let us consider the biggest such partial sum *i.e.* $\max_{j'} c_{i,j'}$ such that $c_{i,j'} \in W$. We will prove how to perfectly simulate each term in $c_{i,j'}$ and then each term in $c_i + c_{i,j'}$ independently. This will allow us to perfectly simulate the partial sums of c_i observed through W as well as the output share c_i.
Observe first that $a_i \cdot b_i$ is perfectly simulated using the input share $i \in I_1 \cap I_2$. Then, for each $r_{i,j}$ such that $i < j$ in $c_{i,j'}$ (and similarly in $c_{i,j'} + c_i$), we can simulate as a random value without the need for any input share. Next, for $i > j$ we can simulate the term $r_{i,j} = a_j \cdot b_i + r_{j,i} + a_i \cdot b_j$ as follows:
  - if $i, j \in I_1 \cap I_2$, then the probe can be perfectly simulated from the input shares in I_1 and I_2 and by generating the random value $r_{j,i}$.
  - otherwise, we are sure that $i \in I_1 \cap I_2$ and $j \notin I_1 \cap I_2$, by construction of the sets I_1 and I_2. Meanwhile, since $j \notin I_1 \cap I_2$, we are sure that there are no probes in W on $r_{i,j}$ nor a partial sum of it nor $r_{j,i}$. In addition, since $j \notin I_1 \cap I_2$, then either $j = o$ or $j \in O$ and all other output shares of indices in O are masked by random values different than $r_{j,i}$. Then, we can conclude that the random value $r_{j,i}$ only appears in the expression of $r_{i,j}$ and can thus be used to mask it, similarly as to the above case.

  Hence, we proved how to simulate each term in the expressions of $c_{i,j'}$ and $c_i + c_{i,j'}$ independently, which allows us to simulate the output share c_i and the probed variables of the form $c_{i,j'}$ for the index i.

This concludes the proof that the n-share ISW multiplication gadget is free $(n-2)$-SNI.

B.4 Proof of Proposition 6

We will prove that the n-share ISW multiplication gadget as described in Algorithm 1 is not free $(n-2)$-SNI. To demonstrate this result, we exhibit a counterexample as a set of probes which represents a failure for the property. Let $W = \{a_1 \cdot b_2 - r_{1,2}, \ldots, a_1 \cdot b_n - r_{1,n}\}$ be a set of $n-1$ probes on an n-share ISW multiplication gadget. For the gadget to be free $(n-1)$-SNI, there must exist sets of input shares indices I_1 on $\overrightarrow{a}$ and I_2 on $\overrightarrow{b}$ such that $|I_1| \leq n-1$, $|I_2| \leq n-1$, and all output shares $\overrightarrow{c}_{|I_1 \cap I_2}$ and probes in W can be perfectly simulated using input shares from I_1 and I_2. In addition, for any $O \subsetneq [1:n] \setminus (I_1 \cap I_2)$, the output shares $\overrightarrow{c}_{|O}$ must be uniform and independent conditioned on probes in W and $\overrightarrow{c}_{|I_1 \cap I_2}$.

However, this condition is not satisfied for the previously defined set of probes W. In fact, observe that the output share

$$c_1 = a_1 \cdot b_1 + r_{1,2} + \ldots + r_{1,n}$$

is not independent and uniform conditioned on the probes in W. Indeed, we have

$$c_1 + \sum_{w \in W} w = a_1 \cdot b.$$

Hence, we cannot have the index 1 in O, and we must add the index 1 to both I_1 and I_2. Moreover, to perfectly simulate the probes W and the output share c_1 (we need to simulate it because $1 \in I_1 \cap I_2$), we need all input shares of $\overrightarrow{b}$, and the input share a_1. This means that we have $\{1\} \in I_1$ and $I_2 = \{1, \ldots, n\}$. Since $|I_2| = n$, then we cannot satisfy the necessary conditions for free $(n-1)$-SNI. $\square$

B.5 Proof of Lemma 1

We will prove Lemma 1, *i.e.* that the n-share ISW refresh gadget is free $(n-1)$-SNI. Let $\overrightarrow{x}$ be the input sharing of the ISW refresh gadget, and $\overrightarrow{y}$ be its output sharing. Let W be a set of internal probes on the gadget such that $|W| \leq n-1$. Notice that compared to the ISW multiplication gadget, the probes have simpler forms. Namely, each probe is of one of the following forms:

1. input shares x_i.
2. partial sum $y_{i,j} = \begin{cases} x_i - r_{1,i} - \cdots - r_{j,i} & \text{if } j < i \\ x_i - r_{1,i} - \cdots - r_{i-1,i} + r_{i,i+1} + \cdots + r_{i,j} & \text{otherwise} \end{cases}$ such that $j \in [1:n] \setminus \{i\}$.
3. random variable r_{ij} for $i < j$.

In this case, we can construct the set of input shares indices I on $\overrightarrow{x}$ similarly to the way we construct the set I_1 in the case of the ISW multiplication gadget. In this case, for each probe of the first or second form (*i.e.* input share or partial sum of output share), we add i to I. Then, for each probe of the third form, if $i \in I$, we add j to I, otherwise we add i to I. We can easily see that all of the probes in W can be perfectly simulated from the input shares indexed in I. In addition, all of the output shares $\overrightarrow{y}|_I$ can also be trivially perfectly simulated from the input shares indexed in I and by generating random values and summing them. Notice that at this point in the case of the ISW multiplication gadget, we already have a failure for free $(n-1)$-SNI as shown in Proposition 6. This comes from the fact that each output share involves several shares of the second input which leads to a failure. In the case of the ISW refresh gadget, each output share involves exactly one input share.

We hence only need to prove that for any set $O \subsetneq [1:n] \setminus I$, the output shares $\overrightarrow{y}|_O$ are uniformly distributed and mutually independent from the probes in W and $\overrightarrow{y}|_I$. We can prove this in the same way we do it for the case of the ISW multiplication gadget. Namely, let $O \subsetneq [1:n] \setminus I$ and let $o \notin O \cup I$ (we are sure that o exists since $O \subsetneq [1:n] \setminus I$), and we observe that each y_i indexed in O is composed of $n-1$ random values, and at most one of them can enter in the expression of each other output wire y_j for $j \in [1:n] \setminus \{i\}$ or a probe in W. Then for each $i \in O$:

- if $o < i$, we know that the random variable $r_{o,i}$ does not appear in any probe in W by construction of the set I since $i \notin I$ and $o \notin I$, and so $r_{o,i}$ does not appear in any output share indexed in I. It only appears in the expression of y_i, so it can be used to mask its expression. Hence, y_i can be generated as a fresh random value.
- if $o > i$, we know that the random variable $r_{i,o}$ does not appear in any probe in W by construction of the set I since $i \notin I$ and $o \notin I$, and so $r_{i,o}$ does not appear in any output share indexed in I. It only appears in the expression of y_i, so it can be used to mask its expression. Hence, y_i can be generated as a fresh random value.

Hence, we proved that each output share indexed in O is generated as a fresh random value independently of the probes, the output shares indexed in I, and the other output shares in O. This concludes the proof that the ISW refresh gadget is free $(n-1)$-SNI, which concludes the proof of Lemma 1. □

B.6 Proof of Proposition 7

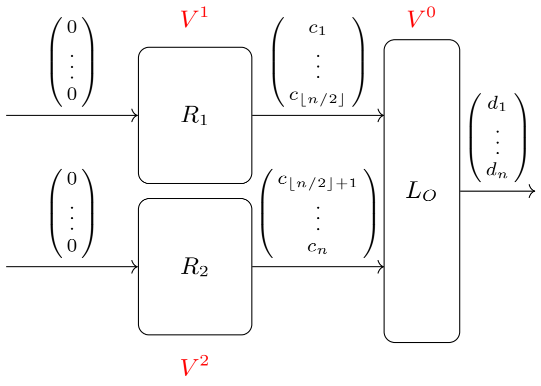
V^1
V^0
V^2
R_1
R_2
L_O
$(0, \dots, 0)$
$(0, \dots, 0)$
$(c_1, \dots, c_{\lfloor n/2 \rfloor})$
$(c_{\lfloor n/2 \rfloor+1}, \dots, c_n)$
$(d_1, \dots, d_n)$

Fig. 7. Optimized $\mathcal{O}(n \log n)$ zero-encoding gadget from Algorithm 2

We will prove Proposition 7 ,*i.e.* that Algorithm 2 is $(n-1)$-free. We will prove the result by recurrence on the number of shares $n \geq 2$.

The gadget in the base case $(n = 2)$ gives the following output sharing:

$$d_1 \leftarrow r$$
$$d_2 \leftarrow -r$$

The proof in this case is easy. Mainly, we can have $|W| = 0$ or $|W| = 1$.

- if $W = \emptyset$, then J has to be $\emptyset$. Then the sets O to consider are $O = \{1\}$ or $O = \{2\}$, and it is clear that in both cases, $\overrightarrow{d}|_O$ is independent and uniformly distributed thanks to the random value r which doesn't appear in any other probed value.
- If $|W| = 1$, then we certainly have $W = \{r\}$ or $W = \{-r\}$ and in this case we can fix $J = \{1\}$. We can clearly see that $[1 : 2] \setminus J = \{2\}$, and so $O \subsetneq [1 : 2] \setminus J = \emptyset$. The proof holds in this case since $\overrightarrow{d}|_O = \emptyset$.

This conclues the proof for the base case.

Next we suppose that for any number of shares $n' < n$, the gadget G satisfies Proposition 7, and we prove the property for n shares. To prove this, we split the gadget into three subgadgets as in Figure 7, where R_1 and R_2 gadgets correspond to the two recursive calls respectively, and gadget L_O corresponds to the loop which adds $\lfloor n/2 \rfloor$ random values to the intermediate sharing. Any set of probes W on the gadget can be split it into disjoint sets $W = V^0 \cup V^1 \cup V^2$ as shown in Figure 7. Namely, the set V^1 (resp. V^2) contains internal and output probes on sub-gadget R_1 (resp. R_2), while the set V^0 contains internal and output probes on the layer L_O, *i.e.* probes of the form d_i for $i \in [1 : n]$ or r_j for $j \in [1 : \lfloor n/2 \rfloor]$. Hence, probes on the input of L_O are considered in V^1 and V^2. Observe that the output sharing d_i can be expressed as:

- if n is odd, then for $i \in [1 : \lfloor n/2 \rfloor]$,

$$\begin{aligned} d_i &= c_i + r_i \\ d_{i+\lfloor n/2 \rfloor} &= c_{i+\lfloor n/2 \rfloor} - r_i \\ d_n &= c_n, \end{aligned}$$

- if n is even, then for $i \in [1 : n/2]$,

$$\begin{aligned} d_i &= c_i + r_i \\ d_{i+n/2} &= c_{i+n/2} - r_i. \end{aligned}$$

In order to use the induction hypothesis, we need the following condition to hold for the gadget R_1:

$$|V^1| \leq \lfloor n/2 \rfloor - 1 \tag{2}$$

and the following for the gadget R_2:

$$|V^2| \leq \lceil n/2 \rceil - 1. \tag{3}$$

We consider first two easy cases for the proof:

1. $|\mathbf{V^1}| \geq \lfloor \mathbf{n/2} \rfloor$. Then we must have $|V^2| \leq \lceil n/2 \rceil - 1$, because we have that $|W| \leq n - 1$.
   Since (3) holds, from the induction hypothesis on R_2, there exists a subset J_2 of output indices with $|J_2| \leq |V^2|$, such that for any $O \subsetneq [\lfloor n/2 \rfloor + 1 : n] \setminus J_2$,

the shares in $\overrightarrow{c}|_O$ are uniformly and independently distributed, conditioned on the probed variables V^2 and $\overrightarrow{c}|_{J_2}$.
Since $|V^1| \geq \lfloor n/2 \rfloor$, we can set $J_1 = [1 : \lfloor n/2 \rfloor]$. We finally define a set of output shares for the overall gadget as $J = J_1 \cup J_2 \cup \{i \mid d_i \in V^0\}$.
We have

$$|J| \leq |J_1| + |J_2| + |V^0| \leq |W| \ .$$

Next, let

$$O \subsetneq [1 : n] \setminus J \ .$$

First observe that

$$([1 : n] \setminus J) \subseteq ([1 : n] \setminus (J_1 \cup J_2)) = ([\lfloor n/2 \rfloor + 1 : n] \setminus J_2)$$

since $J_1 = [1 : \lfloor n/2 \rfloor]$. This means that $O \subsetneq [\lfloor n/2 \rfloor + 1 : n] \setminus J_2$. From the induction hypothesis on R_2, we know that the shares $\overrightarrow{c}|_O$ are uniformly and independently distributed conditioned on the internal probes V^2 and the shares $\overrightarrow{c}|_{J_2}$. Since the gadgets R_1 and R_2 are independent and use different random values, then $\overrightarrow{c}|_O$ are uniformly and independently distributed conditioned on V^2, V^1 and $c|_{J_1 \cup J_2}$.
Let us now consider $\overrightarrow{d}|_O$. Recall that

$$\overrightarrow{d} = (c_1 + r_1, \ldots, c_{n/2} + r_{n/2}, c_{1+n/2} + r_1, \ldots, c_n + r_{n/2})$$

if n is even, or

$$\overrightarrow{d} = (c_1 + r_1, \ldots, c_{\lfloor n/2 \rfloor} + r_{\lfloor n/2 \rfloor}, c_{1+\lfloor n/2 \rfloor} + r_1, \ldots, c_{n-1} + r_{\lfloor n/2 \rfloor}, c_n)$$

if n is odd. Also, for $i \in [1 : n] \setminus J$, we are sure that $d_i \notin V^0$ by construction of the set J. Hence, we can also conclude that the output shares $\overrightarrow{d}|_O$ are uniformly and independently distributed, conditioned on the probed variables W and $d|_{J_1 \cup J_2}$. Indeed, this comes from the fact that $\overrightarrow{c}|_O$ are independent and uniform conditioned on V^2, V^1 and $\overrightarrow{c}|_{J_1 \cup J_2}$, and consequently also independent of the probed random values in V^0, *i.e.* each d_i for $i \in O$ can be simulated as a fresh random value thanks to the uniformity and independence of c_i without using the random r_i. This concludes the proof of the Lemma in the case where $|V^1| \geq \lfloor n/2 \rfloor$.

2. $|\mathbf{V^2}| \geq \lceil \mathbf{n/2} \rceil$. This case can be treated exactly like the above case.

For the rest of the proof, we suppose that $|V^1| \leq \lfloor n/2 \rfloor - 1$, and $|V^2| \leq \lceil n/2 \rceil - 1$.

We first consider that n is even. Without loss of generality, let $|V^1| \leq |V^2|$ (the case where $|V^2| \leq |V^1|$ is symmetric since n is even). Since (2) holds, from the induction hypothesis on R_1, there exists a subset J_1 of output indices with $|J_1| \leq |V^1|$, such that for any $O_1 \subsetneq [1 : n/2] \setminus J_1$, the shares in $\overrightarrow{c}|_{O_1}$ are uniformly and independently distributed, conditioned on the probed variables V^1 and $c|_{J_1}$. Without loss of generality, let $J_1 = \{1, \ldots, \ell\}$ with $\ell \leq n/2 - 1$. We construct the set J_2 as follows. We start with $J_2 = \{1 + n/2, \ldots, \ell + n/2\}$.

Then, for each $r_i \in V^0$ such that $i \notin J_1$, we add $i + n/2$ to J_2. We will consider the probes $d_i \in V^0$ later in the proof. Finally, we let $J = J_1 \cup J_2$. We can check that

$$|J| \leq 2 \cdot |V^1| + |V^0| \leq |V^1| + |V^2| + |V^0| \leq |W| \ .$$

We will now prove that for any $O \subsetneq [1 : n] \setminus J$, the output shares $\overrightarrow{d}|_O$ are independent and uniform conditioned on the probes W and the shares $d|_J$. Let us consider

$$O' = [1 : n] \setminus J = ([1 : n/2] \setminus J_1) \cup ([n/2 + 1 : n] \setminus J_2) \ .$$

Now, denote $[n/2 + 1 : n] \setminus J_2 = \{n/2 + i_1, \ldots, n/2 + i_k\}$ where $\{i_1, \ldots, i_k\} \subseteq \{\ell + 1, \ldots, n/2\} \subseteq [1 : n/2] \setminus J_1$. Observe that

$$\overrightarrow{d}|_{O'} = (c_{\ell+1} + r_{\ell+1}, \ldots, c_{n/2} + r_{n/2}, \qquad c_{n/2+i_1} + r_{i_1}, \ldots, c_{n/2+i_k} + r_{i_k}) \ .$$

From the construction of the set $J = J_1 \cup J_2$, we know that $\{r_{i_1}, \ldots, r_{i_k}\} \cap V^0 = \emptyset$. Now let $O \subsetneq [1 : n] \setminus J = O'$, *i.e.* there is at least one i such that $i \in O' \setminus O$.

1. If $i \in \{\ell + 1, \ldots, n/2\}$, then we can use the induction hypothesis of gadget R_1 and the fact that the shares $\{c_{\ell+1}, \ldots, c_{n/2}\} \setminus \{c_i\}$ are independent and uniform conditioned on the probes V^1 and the shares $\overrightarrow{c}|_{I_1}$. Hence, the output shares $\{c_{\ell+1} + r_{\ell+1}, \ldots, c_{n/2} + r_{n/2}\} \setminus \{c_i + r_i\}$ are independent and uniform conditioned on the probes in $W \setminus \{d_i \in V^0\}$, and the other output shares, by independence of the random values used in the gadgets R_1, R_2 and L_O. In addition, since the random values $\{r_{i_1}, \ldots, r_{i_k}\}$ are not probed, then the output shares $\{c_{n/2+i_1} + r_{i_1}, \ldots, c_{n/2+i_k} + r_{i_k}\}$ are also independent and uniform, conditioned on the other probes and the output shares. Hence, all output shares $\overrightarrow{d}|_O$ are independent and uniform conditioned on the probes $W \setminus \{d_i \in V^0\}$ and the output shares $\overrightarrow{d}|_J$.
2. If $i \in \{n/2+i_1, \ldots, n/2+i_k\}$, then in particular, the random value $r_{i-n/2}$ can be used to mask the expression of the output share $d_{i-n/2} = c_{i-n/2} + r_{i-n/2}$ for $i-n/2 \in \{\ell+1, \ldots, n/2\}$. Then we can again use the induction hypothesis of R_1 and the same arguments as before to prove that all output shares $d|_O$ are independent and uniform conditioned on the probes $W \setminus \{d_i \in V^0\}$ and the output shares $d|_I$.

Finally, in order to consider probes $d_i \in V^0$, for each such probe, we add i to J. This amounts to removing one coordinate from the considered vector $d|_{O'}$. Clearly, this does not change the correctness of the proof, *i.e.* thanks to the above arguments, we sill have that for any $O \subsetneq [1 : n] \setminus (J \cup \{i \mid d_i \in V^0\}) \subseteq O'$, the output shares $\overrightarrow{d}|_O$ are independent and uniform conditioned on the probes W and the output shares $\overrightarrow{d}|_J$. Additionally, $|J|$ is still less than $|W|$ since we only considered the remaining probes in V^0.

This concludes the proof in the case where n is even.

Next, let us suppose that n is odd in order to conclude the proof of Proposition 7. We consider two cases:

1. $|V^2| \leq |V^1|$. Since (3) holds, from the induction hypothesis on R_2, there exists a subset J_2 of indices with $|J_2| \leq |V^2|$, such that for any $O_2 \subsetneq [\lceil n/2 \rceil : n] \setminus J_2$, the shares in $\overrightarrow{c}|_{O_2}$ are uniformly and independently distributed, conditioned on the probed variables V^2 and $c|_{J_2}$. Without loss of generality, we define $J_2 = \{i_1, \ldots, i_\ell\}$ with $\ell = |J_2|$. We construct the set J_1 as follows. We start with

   $$J_1 = \{j_1 - \lfloor n/2 \rfloor, \ldots, j_{\ell'} - \lfloor n/2 \rfloor\}, \quad \text{for} \ \ \{j_1, \ldots, j_{\ell'}\} = J_2 \setminus \{n\} \ .$$

   Then, for each $r_i \in V^0$ such that $i + \lfloor n/2 \rfloor \notin J_2$, we add i to J_1. We can consider probes $d_i \in V^0$ later similarly as above in the proof. Finally, we let $J = J_1 \cup J_2$. We can check that

   $$|J| \leq 2 \cdot |J_2| + |V^0| \leq 2 \cdot |V^2| + |V^0| \leq |V^1| + |V^2| + |V^0| \leq |W| \ .$$

   The rest of the proof is similar to the case where n is even. Indeed, we can show that for $O \subsetneq [1 : n] \setminus J$, part of the output shares $\overrightarrow{d}|_O$ are independent and uniform thanks to the induction hypothesis on R_2 (the output share $d_n = c_n$ is either independent and uniform by the hypothesis on R_2, or $n \in J_2$ and it does not need to be independent and uniform), while the other part of the shares are independent and uniform thanks to the fact that the random values $r_{i-\lfloor n/2 \rfloor}$ for $i \in [\lceil n/2 \rceil : n-1] \setminus J_2$ are not observed through V^0.

2. $|V^1| < |V^2|$. Since (2) holds, from the induction hypothesis on R_1, there exists a subset J_1 of input indices with $|J_1| \leq |V^1|$, such that for any $O_1 \subsetneq [1 : \lfloor n/2 \rfloor] \setminus J_1$, the shares in $\overrightarrow{c}|_{O_1}$ are uniformly and independently distributed, conditioned on the probed variables V^1 and $\overrightarrow{c}|_{J_1}$. Without loss of generality, let $J_1 = \{1, \ldots, \ell\}$ with $\ell \leq \lfloor n/2 \rfloor - 1$. We construct the set J_2 as follows. We start with $J_2 = \{1 + \lfloor n/2 \rfloor, \ldots, \ell + \lfloor n/2 \rfloor\} \cup \{n\}$. Then, for each $r_i \in V^0$ such that $i \notin J_1$, we add $i + \lfloor n/2 \rfloor$ to J_2. We can consider probes $d_i \in V^0$ later similarly as above in the proof. Finally, we let $J = J_1 \cup J_2$. We can check that

   $$|J| \leq 2 \cdot |J_1| \ \underline{+1} + |V^0| \leq 2 \cdot |V^1| + 1 + |V^0| \leq |V^1| + |V^2| - 1 \ \underline{+1} + |V^0| \leq |W|$$

   where the underlined term in the equation is due to the fact that we add the share n to J_2 anyway. Since $n \in J$, then the share $d_n = c_n$ does not need to be independent and uniform, in other words for any $O \subsetneq [1 : n] \setminus J$, we have $n \notin O$. Hence, the rest of the proof in this case is similar to the proof in the other cases. We can proof in the same way that the output shares $\overrightarrow{d}|_O$ are independent and uniform conditioned on the shares $\overrightarrow{d}|_I$ and the probes W.

This concludes the proof in the case where n is odd, which concludes the proof of Proposition 7. □

C Proofs of Section 5

C.1 Proof of Proposition 8

We will prove Proposition 8, *i.e.* that Algorithm 3 is correct. Recall that the algorithm verifies, for an n-share gadget G of output sharing $\overrightarrow{y}$, if for any $O \subsetneq [1:n]$, the output shares $\overrightarrow{y}_{|O}$ are independent and uniform. In our context, this means that each of the output shares in $\overrightarrow{y}_{|O}$ can be generated as a fresh random value conditioned on the other output shares in $\overrightarrow{y}_{|O}$.
Consider the matrix $S = (\overrightarrow{s_{y_1}} | \ldots | \overrightarrow{s_{y_n}})^T$. In fact, Algorithm 3 performs a gaussian elimination on this matrix thanks to the Gaussian procedure (first line). Then, the algorithm checks if the rank of S is exactly $n-1$. This is equivalent to checking that $v = n-1$ in the output of the Gaussian procedure (second and third line). We hence need to prove that $rank(S) = n-1$ is equivalent to the fact that for any $O \subsetneq [1:n]$, the output shares $\overrightarrow{y}_{|O}$ are independent and uniform.

Direction 1: $rank(S) = n-1 \implies$ for any $O \subsetneq [1:n]$, the output shares $\overrightarrow{y}_{|O}$ are independent and uniform.

Suppose that $\exists O \subsetneq [1:n]$ and $\exists j \in O$ such that y_j cannot be generated as a fresh independent random value conditioned on the other output shares in $O \setminus \{j\}$. This means that we can write a linear combination

$$\overrightarrow{s_{y_j}} + \overrightarrow{s_{y_{i_1}}} + \ldots + \overrightarrow{s_{y_{i_k}}} = \overrightarrow{0} \ . \qquad (4)$$

for $\{i_1, \ldots, i_k, j\} \subseteq O$.
Now consider the matrix $S = (\overrightarrow{s_{y_1}} | \ldots | \overrightarrow{s_{y_n}})^T$. By correctness of the output of the gadget G, we know that there is a linear combination of size n such that

$$\overrightarrow{s_{y_1}} + \ldots + \overrightarrow{s_{y_n}} = \overrightarrow{0} \ . \qquad (5)$$

In addition, from (4) and (5), there is another linear combination

$$\overrightarrow{s_{j_1}} + \ldots + \overrightarrow{s_{y_{j_k}}} = \overrightarrow{0} \ . \qquad (6)$$

such that $\{j_1, \ldots, j_k\} = [1:n] \setminus \{i_1, \ldots, i_k, j\}$. Thus, $rank(S) \leq n-2$. This proves that the rank of the matrix S is not equal to $n-1$.

Direction 2: for any $O \subsetneq [1:n]$, the output shares $\overrightarrow{y}_{|O}$ are independent and uniform $\implies rank(S) = n-1$.

Suppose that $rank(S) < n-1$. Then in a similar way, since we know by correctness of the gadget that the linear combination of size n in equation (5) holds, then there must exist at least another linear combination of a form similar to that in (4) such that $|\{j, i_1, \ldots, i_k\}| < n-1$. Then, we can choose $O = \{j, i_1, \ldots, i_k\}$, for which the output shares $\overrightarrow{y_{|O}}$ are clearly not independent and uniform.

By this, we conclude the proof of Proposition 8.

C.2 Proof of Lemma 3

We will prove Lemma 3, *i.e.* that Algorithms 4 and 5 are correct when checking free t-SNI property. As described in Section 6 , Algorithm 4 starts by determining the sets of input shares I_1 and I_2 necessary to perfectly simulate the probes given as input. The correctness of this process (lines 1 to 7 of the algorithm) directly follows from the results in [13]. Then, the test on line 7 determines if the set of probes already constitutes a failure for the SNI property, which implies a failure for free SNI. After that, Algorithm 4 performs Gaussian elimination and determines the index v as described in Lemma 2, in order to call Algorithm 5, which is a direct application of the lemma. Namely, it looks for sets I'_1 and I'_2 of size at most t starting from I_1 and I_2, respectively, such that $I'_1 \cap I'_2$ satisfies the free SNI property. The construction of the sets follows the partition process described in Lemma 2. This can be seen through the loop on line 2 where for each possible subset C' of C we add O_{f_i} to I'_1 and I'_2 for $i \in C'$, and $\overline{O_{f_i}}$ to I'_1 and I'_2 for $i \notin C'$. Note that since we also need to perfectly simulate the expressions f_i for $i \in C$, we add I_{1,f_i} and I_{2,f_i} to I'_1 and I'_2 respectively for $i \in C'$, and $\overline{I_{1,f_i}}$ and $\overline{I_{2,f_i}}$ to I'_1 and I'_2 respectively for $i \notin C'$. In addition, one can check that adding these sets of input shares does not change the correctness of Lemma 2. Finally, thanks to the result of the lemma, if we cannot find such sets I'_1 and I'_2 of size less than t, then the set of probes constitutes a failure for free SNI, which is why the algorithm returns false on line 14. This concludes the proof of Lemma 3.

C.3 Proof of Lemma 4

The proof of Lemma 4 is very similar to that of Lemma 3. The only difference between Algorithm 6 and Algorithm 5 is in the construction of the sets I'_1, I'_2 and O'. In the case of the free SNI property, the set O' is imposed to be $I'_1 \cap I'_2$, while in the case of the IOS property, the set O' must only be of size at most t. Algorithm 6 looks for sets I'_1, I'_2, and O', which satisfy the unbalanced free SNI property, equivalent to the IOS property. The construction of these sets also follows the result of Lemma 2.

C.4 Proof of Lemma 2

We will prove Lemma 2. Namely, let G be a uniform n-share LR-gadget for the base field $\mathbb{K} = \mathbb{F}_2$. Let $\overrightarrow{W} = (w_1, \ldots, w_k)$ be a tuple of internal probes on G and let

$$(f_1, \ldots, f_{n+k-1}) \leftarrow \mathsf{Gaussian}(w_1, \ldots, w_k, y_1, \ldots, y_{n-1}) \ .$$

Let $v \in [0 : n+k-1]$ such that $\overrightarrow{s_{f_i}} \neq \overrightarrow{0}$ for all $i \leq v$ and $\overrightarrow{s_{f_i}} = \overrightarrow{0}$ for all $i > v$ and denote $P = \{f_{v+1}, \ldots, f_{n+k-1}\}$. We will consider partitions $P_1 \cup P_2 = P$ (with $P_1 \cap P_2 = \emptyset$).

Let us start with the case $P_1 = P$ and $P_2 = \emptyset$ and recall the definition of the set $I_{(P_1,P_2)}$ as

$$I_{(P_1,P_2)} = \Big(\bigcup_{f_i \in P_1} O_{f_i} \Big) \cup \Big(\bigcup_{f_i \in P_2} \overline{O_{f_i}} \Big) ,$$

with $\overline{O_{f_i}} = [1 : n] \setminus O_{f_i}$. Let us consider some given values of the probed wires $\overrightarrow{W}$ and some given values of the input shares $\overrightarrow{x_1}$, $\overrightarrow{x_2}$. These values imply some constraints on the output shares $\overrightarrow{y}$ and the randomness $\overrightarrow{r}$ which are summed up with the following system:

$$N \cdot (w_1, \dots, w_k, y_1, \dots, y_{n-1})^T = S' \cdot \overrightarrow{r} + \overrightarrow{F}(\overrightarrow{x_1}, \overrightarrow{x_2}) \tag{7}$$

with the additional equation:

$$y_1 + \cdots + y_n = g(\overrightarrow{x_1}, \overrightarrow{x_2}) \tag{8}$$

where N is the $(k + n - 1) \times (k + n - 1)$ square matrix obtained by applying Gaussian, $\overrightarrow{F}$ is the vector of functions $N \cdot (f_{w_1}, \dots, f_{w_k}, f_{y_1}, \dots, f_{y_{n-1}})$, and g is the function computed by the gadget (such as defined at the beginning of Section 5.1). Recall that S' is the row reduced version of S, which (without loss of generality, up to a permutation of the w_i's and the r_i's) is of the following form:

$$S' = \left(\begin{array}{c|c} I_v & * \\ \hline 0 & 0 \end{array} \right) .$$

In the following, we shall call the v first equations of the system in (7) (corresponding to the v first rows of N), the "probabilistic equations" while the next $k + n - 1 - v$ shall be called the "deterministic equations" and (8) the "output sum" equation. By definition of I, the deterministic equations only involve output shares from $\overrightarrow{y}|_I$ while the probabilistic equations further involve output shares from $\overrightarrow{y}|_{[1:n-1]}$. The output sum equation involves all the output shares.

Given some values of $\overrightarrow{W}$, $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$, the deterministic equations yield linear constraints in the $\overrightarrow{y}|_I$, which means that the $\overrightarrow{y}|_I$ are jointly dependent on $\overrightarrow{W}$, $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$. Then, for any assignment of $\overrightarrow{y}|_I$ satisfying these constraints, we further have that for every set $O \subsetneq [1 : n] \setminus I$ of cardinality $n - 1 - |I|$ and any value of $\overrightarrow{y}|_O$, the system has exactly $|\mathbb{F}_2|^{\rho - v} = 2^{\rho - v}$ solutions (where $\rho = |\overrightarrow{r}|$ the number of random gates in G). Indeed, given $\overrightarrow{y}|_I$ and $\overrightarrow{y}|_O$, we have exactly one assignment of $\overrightarrow{y}$ satisfying the output sum equation (8). On the other hand, we have $|\mathbb{F}_2|^{\rho - v}$ assignments of $\overrightarrow{r}$ satisfying the system. Indeed, remark that thanks to the form of S', we can take any value for the $\rho - v$ last coordinates of $\overrightarrow{r}$ and deduce the v first accordingly (and from the values of $\overrightarrow{W}$, $\overrightarrow{x_1}$, $\overrightarrow{x_2}$ and $\overrightarrow{y}$).

We have thus shown that for any choice of $\overrightarrow{y}|_O$ the system has exactly the same number of solutions for $\overrightarrow{r}$ which implies that $\overrightarrow{y}|_O$ is mutually independent of $\overrightarrow{W}$, $\overrightarrow{x_1}$, $\overrightarrow{x_2}$ and $\overrightarrow{y}|_I$. Note that since the property holds for all $O \subsetneq [1:n] \setminus I$ of cardinality $n-1-|I|$, it trivially holds for any $O \subsetneq [1:n] \setminus I$ (possibly of smaller cardinality). The free property of I for the probes $\overrightarrow{W}$ directly follows.

Let us now consider the case $P_2 \neq \emptyset$. For each f_i in P_2, subtract the equation (8) to the ith equation of (7). It is not hard to see that the new system is equivalent to the former one. Then we can apply the above argument in the exact same way. Indeed, for equation i such that $f_i \in P_2$ (which is a deterministic equation by definition), the involved output shares are those in $\overline{O_{f_i}}$, hence the output shares $\overrightarrow{y}|_I$ are dependent on $\overrightarrow{W}$, $\overrightarrow{x_1}$, $\overrightarrow{x_2}$ (with I defined in (1) w.r.t. P_1, P_2) while the $\overrightarrow{y}|_O$ for any set $O \subsetneq [1:n] \setminus I$ are uniform conditioned to $\overrightarrow{W}$, $\overrightarrow{x_1}$, $\overrightarrow{x_2}$ and $\overrightarrow{y}|_I$. In other words, any set I of the form of Equation (1) satisfies the free property for $\overrightarrow{W}$.

Let us now consider a set I' which is not a super set of the set $I_{(P_1,P_2)}$ defined by (1) whatever the partition $P_1 \cup P_2 = P$. This implies that, whatever the equivalent version of the system of constraints, there always exists a set $J \subsetneq [1:n] \setminus I'$ such that all the shares $\overrightarrow{y}|_J$ are involved in the deterministic equations of the system, which means that I' cannot satisfy the free property for $\overrightarrow{W}$. To see the existence of such a set J, note that since I' is not a super set of the set $I_{(P_1,P_2)}$, then we can define $J = I_{(P_1,P_2)} \setminus I' \neq \emptyset$. Moreover, we can exclude the case $J = [1:n] \setminus I'$ since this would imply

$$I_{(P_2,P_1)} = [1:n] \setminus I_{(P_1,P_2)} \subseteq I'$$

and hence would lead to a contradiction (since I' would be a super set for the swapped partition (P_2, P_1)).

This concludes the proof of Lemma 2.

D Proof of generalized threshold-probing composition

Let us first give a formal definition for completeness (which is needed for Definition 13).

Definition 20 (Two-Input Probing Completeness). *Let G be an (n-share, 2-to-1) gadget. G is said* t-probing complete*, if there exists at least one wire p and two distinct indices $i_1, i_2 \in [1:n]$, such that for every (two-stage) simulator* $\mathsf{Sim} = \big(\mathsf{Sim}_1, \mathsf{Sim}_2\big)$ *and for every input* $(\overrightarrow{x_1}, \overrightarrow{x_2}) \in \mathbb{K}^n \times \mathbb{K}^n$,

1. $\mathsf{Sim}_1(p) = (I_1, I_2)$ *where* $I_1, I_2 \subseteq [1:n]$,
2. $\mathsf{Sim}_2(p, \overrightarrow{x_1}|_{I_1}, \overrightarrow{x_2}|_{I_2}) \overset{\text{id}}{=} \mathsf{AssignWires}(G, p, (\overrightarrow{x_1}, \overrightarrow{x_2}))$,

we have $i_1 \in I_1$ and $i_2 \in I_2$.

Next, our proof is based on the games Game 0, Game 1', Game 2' and Game 3 defined in Figure 4 and Figure 5.

D.1 Game definitions

We now formally define the associated constraints on the set of probes $\mathcal{P}_0$, $\mathcal{P}_{1'}$, $\mathcal{P}_{2'}$ and $\mathcal{P}_3$. For the sake of completeness, we also define with the same formalism the sets $\mathcal{P}_1$ and $\mathcal{P}_2$ that correspond to Game 1 and Game 2 of [12].

In the games, $\mathcal{P}_0$ is a t-probe set in C, that is, an arbitrary set of $t = n-1$ probes in the original circuit C. We can attribute each probe to one of the three kinds of gadgets: affine (a), single-input RNL (I) and two-input RNL (II), giving the partition

$$\mathcal{P}_0 = \mathcal{P}_a \cup \mathcal{P}_I \cup \mathcal{P}_{II}. \tag{9}$$

Next, for Game 1, $\mathcal{P}_1$ is a t-I-only probe set. That is, it is a union $\mathcal{P}_1 = \mathcal{P}'_a \cup \mathcal{P}'_I \cup \mathcal{P}'_{II}$, where $\mathcal{P}'_a$, $\mathcal{P}'_I$ and $\mathcal{P}'_{II}$ are derived from a partition $\mathcal{P}_0 = \mathcal{P}_a \cup \mathcal{P}_I \cup \mathcal{P}_{II}$ of a t-probe set, and contain only the probes specified next. For each affine gadget g and each probe $p \in \mathcal{P}_a$ that belongs to g, the set $\mathcal{P}'_a$ contains all the inputs of g with the same share index as p. For each single-input (respectively two-input) RNL gadget g in C, the set $\mathcal{P}'_I$ (resp. $\mathcal{P}'_{II}$) contains the input shares needed to simulate the probes in g. If there are at most $n-2$ probes in g, this set is determined using the free SNI simulator, otherwise it is determined with the NI simulator. For Game 1', the set $\mathcal{P}'_1$ is a t-IO-only probe set. That is, it is a union $\mathcal{P}'_1 = \mathcal{P}'_a \cup \mathcal{P}'_I \cup \mathcal{P}'_{II} \cup \mathcal{P}^o_I \cup \mathcal{P}^o_{II}$, where $\mathcal{P}'_a$, $\mathcal{P}'_I$ and $\mathcal{P}'_{II}$ are derived from a t-probe set in the same way as for a t-I-only probe set. Moreover, for each gadget g, $\mathcal{P}^o_I$ (resp. $\mathcal{P}^o_{II}$) contains the output shares of g for which the input share (resp. both input shares) with the same index belong to $\mathcal{P}'_I$ (resp. $\mathcal{P}'_{II}$).

Let us now discuss Game 2 and Game 2'. For the former, the set is the same as before the flattening: $\mathcal{P}_2$ must be a t-I-only probe set. For the latter, $\mathcal{P}'_2$ is a t-f-IO-only probe set. That is, it is a union $\mathcal{P}'_2 = \mathcal{P}'_a \cup \mathcal{P}'_I \cup \mathcal{P}'_{II} \cup \mathcal{P}^f_I \cup \mathcal{P}^f_{II}$, where $\mathcal{P}'_a$, $\mathcal{P}'_I$ and $\mathcal{P}'_{II}$ are derived from a partition $\mathcal{P}_0 = \mathcal{P}_a \cup \mathcal{P}_I \cup \mathcal{P}_{II}$ of a t-IO-probe set $\mathcal{P}'_1 = \mathcal{P}'_a \cup \mathcal{P}'_I \cup \mathcal{P}'_{II} \cup \mathcal{P}^o_I \cup \mathcal{P}^o_{II}$. Moreover, for each gadget g, $\mathcal{P}^f_I$ (resp. $\mathcal{P}^f_{II}$)

contains the shares of a new input sharing that correspond to output shares of g that belong to index belong to $\mathcal{P}_I^o$ (resp. $\mathcal{P}_{II}^o$).

Finally, for Game 3, $\mathcal{P}_3$ is t-dI-only: for each two-input RNL gadget g in C, it contains $w(g)$ shares of each input sharing of g, for some weighing w of the gadgets such that the weights belong to $\{0, \ldots, t\}$ and the sum of the weights is t.

D.2 Reductions

We begin with the new reductions.

Proposition 10 (Game 1' to Game 0). *Let C be a n-GTPC. If, for any adversary $\mathcal{A}_{1'}$, there exists a simulator $\mathcal{S}_{1'}$ that wins Game 1', then, for any adversary $\mathcal{A}_0$, there exists a simulator $\mathcal{S}_0$ that wins Game 0.*

Proof. Let $\mathcal{P}_0$ be the set of probes chosen by $\mathcal{A}_0$. If all the probes belong to a single SA gadget, then $\mathcal{A}_0$ behaves as a $\mathcal{A}_{1'}$ adversary and $\mathcal{S}_0$ can behave like $\mathcal{S}_{1'}$. Similarly, if all the probes belong to a single RNL gadget, then using the $n-1$-NI simulator for that gadget, we get a set of probes $\mathcal{P}_{1'}$ on its inputs, and these probes can be simulated by $\mathcal{S}_{1'}$ (we do not use the output probe values generated by $\mathcal{S}_{1'}$).

Next, if the probes are spread on multiple gadgets, there is at most $n-2$ probes in a single gadget. Therefore, using the balanced $(n-2)$-IOS (obtained thanks to Theorem 1) simulators for the RNL gadgets, we get a set of probes $\mathcal{P}_{1'}$ that is similar to $\mathcal{P}_0$, but contains input and output probes instead of internal probes for RNL gadgets. We can simulate the probes $\mathcal{P}_{1'}$ using $\mathcal{S}_{1'}$ and then use the balanced IOS simulators to simulate the internal probes. □

Proposition 11 (Game 2' to Game 1'). *Let C be a n-GTPC. If, for any adversary $\mathcal{A}_{2'}$, there exists a simulator $\mathcal{S}_{2'}$ that wins Game 2', then, for any adversary $\mathcal{A}_{1'}$, there exists a simulator $\mathcal{S}_{1'}$ that wins Game 1'.*

Proof. For each adversary $\mathcal{A}_{1'}$ returning $(x_1, \ldots, x_m)$ and $\mathcal{P}_{1'}$, we define $\mathcal{A}_{2'}$ as the adversary that returns the set of probes $\mathcal{P}_{2'}$ (which is the t-f-IO probe set derived from the t-IO-probe set $\mathcal{P}_{1'}$) and the extended input $(x_1, \ldots, x_M)$ such that the m first elements match the choice of $\mathcal{A}_{1'}$ and the $M-m$ last match the unmasked outputs of the corresponding RNL gadgets. Then, by the free SNI definition, the output of the RNL gadgets are independent of their input sharings. Therefore, the real-world distributions for Game 1' and Game 2' are identical. We can then define $\mathcal{S}_{1'}$ to output the same distribution as the simulator $\mathcal{S}_{2'}$ that wins against $\mathcal{A}_{2'}$. □

Lemma 5. *Let $[v_1], \ldots [v_n]$ be vectors of probes (grouped by share index) representing a $(n-1)$-f-IO-only probe set in a flattened GTPC C'. Let $[x_i]$ be the vector of the input shares of C' with share index i for $i = 1, \ldots, n$, and let M_i and $[a_i]$ be constants such that $[v_i] = M_i[x_i] + [a_i]$ for $i = 1, \ldots, n$. The set of probes can be perfectly simulated iff*

$$Im(M_1) \cap \cdots Im(M_n) = \emptyset.$$

Proof. Since the constants $[a_i]$ are public, it is equivalent to simulate $[v_i]$ or $[v_i] - [a_i]$, therefore this lemma is a consequence of [12, Lemma 2].

Proposition 12 (Game 3' to Game 2'). *Let C be a n-GTPC. If, for any adversary $\mathcal{A}_3$, there exists a simulator $\mathcal{S}_3$ that wins Game 3, then, for any adversary $\mathcal{A}_{2'}$, there exists a simulator $\mathcal{S}_{2'}$ that wins Game 2'.*

Proof. Let us prove the contrapositive statement, *i.e.* if there is a $\mathcal{A}_{2'}$ such that all $\mathcal{S}_{2'}$ fail Game 2', then there is a $\mathcal{A}_3$ such that all $\mathcal{S}_3$ fail Game 3. We denote by $(x_1, \dots, x_M)$ and by $\mathcal{P}_{2'}$ the outputs of $\mathcal{A}_{2'}$. Further, let $M_1, \dots, M_n$ the matrices induced by Lemma 5. Using this lemma, the assumption of the contrapositive statement implies

$$\mathrm{Im}(M_1) \cap \cdots \cap \mathrm{Im}(M_n) \neq \emptyset.$$

Let us now use the partition $\mathcal{P}_{2'} = \mathcal{P}'_a \cup \mathcal{P}'_I \cup \mathcal{P}'_{II} \cup \mathcal{P}^f_I \cup \mathcal{P}^f_{II}$. Let S be the set of share indexes of the probes in $\mathcal{P}'_a \cup \mathcal{P}'_I \cup \mathcal{P}^f_I$. Moreover, let S' be the set of share indexes that appear at least twice in the probes $\mathcal{P}'_{II}$, and let $\mathcal{P}''_{II}$ be the probes in $\mathcal{P}'_{II}$ whose index does not belong to S'. Therefore, all the probes in $\mathcal{P}_{2'} \setminus \mathcal{P}''_{II}$ have their share index in $S \cup S'$, and by construction, knowing that $\mathcal{P}_{2'}$ is a $(n-1)$-f-IO-only probe set, we have $|S \cup S'| + |\mathcal{P}''_{II}|/2 \leq n - 1$.

All the probes in $\mathcal{P}_{2'} \setminus \mathcal{P}''_{II}$ therefore contribute lines in at most $|S \cup S'|$ distinct matrices M_i. Since every matrix contains at least one row (otherwise the intersection would be empty), the remaining probes contribute to at least one row in each of the remaining matrices. There are at least $n - |S \cup S'|$ such matrices and the remaining probes are $\mathcal{P}''_{II}$, whith $|\mathcal{P}''_{II}| \leq 2(n - |S \cup S'|) - 2$. Therefore, at least one matrix M_{i^*} has a single row (let us next denote it w), and that row corresponds to a probe on an input sharing of a two-input RNL gadget (next denoted g).

Let $\mathcal{P}''$ be the set of probes containing the probes $\mathcal{P}''_2$, and where each probe in $\mathcal{P}'_a$, $\mathcal{P}'_1$ and $\mathcal{P}'_2 \setminus \mathcal{P}''_2$ is transformed into a pair of probes on the input sharings of g (both new probes have the same share index as the original probe), which does not change the set of probed inputs $\mathcal{P}'_i$. Let this new set of probes be the output of $\mathcal{A}_3$: the corresponding intersection of matrix images still contains w, hence the probes cannot be perfectly simulated.

Finally, we can prove Theorem 4.

Proof (of Theorem 4). The previous propositions show that if C is secure for Game 3, then it is secure for Game 0. Next, for a standard shared circuit, security for Game 0 implies successively security for Games 1, 2 and 3 [12]. Let us remark that these reductions can be trivially generalized to GTPC. Indeed, their proofs rely on the uniformity of the output sharing of ISW gadgets and on their probing completeness, and these properties are also satisfied for the RNL gadgets in GTPC. For the addition gadget, no particular property is used, hence generalizing them to SA gadgets is not an issue. □

E Proofs for Region Probing Composition

We first give the proof of Theorem 5.

Proof. Let C be an n-RTPC. We shall denote by G_1, G_2, $\ldots$, G_m the different regions of C. For every i, G_i is either a t-IOS gadget, with $t = \lfloor (n-1)/3 \rfloor$ or a full rank SA region as defined in Section 7. For the sake of simplicity, we prove the result assuming that the RCNL gadgets are unbalanced t-IOS (this reduce the number of sets to be considered).

We have to show that for any sets of wires $W_1 \subseteq W_{G_1}, \ldots, W_m \subseteq W_{G_m}$ such that $|W_1| \leq \lceil r|G_1| \rceil \leq t$, $\ldots$, $|W_m| \leq \lceil r|G_m| \rceil \leq t$, there exists a simulator Sim which satisfies

$$\mathsf{Sim}(C, W) = \mathsf{AssignWires}(C, W, \mathsf{AddEnc}(x_1, \ldots, x_\ell))$$

where $W = W_1 \cup \cdots \cup W_m$.

Without loss of generality, let us assume that $|W_1| = |W_2| = \cdots = |W_m| = t$. The simulation goes as follows. For each G_j which is an ℓ_j-to-m_j IOS gadget, we first call $\mathsf{Sim}_1^{\text{ios-}G_j}$ the first IOS simulator of G_j on the set of probes W_j. The simulator then returns sets $I_1, \ldots, I_{\ell_j} \subseteq [1:n]$ and $J_1, \ldots, J_{m_j} \subseteq [1:n]$ all of cardinality at most t such that the probes in W_j can be perfectly simulated from the input shares indexed by $I_1, \ldots, I_{\ell_j}$ (each set for one input sharing) and the output shares indexed by $J_1, \ldots, J_{m_j}$ (each set for one output sharing). Without loss of generality, we assume $|I_1| = \cdots = |I_{\ell_j}| = t$ and $|J_1| = \cdots = |J_{m_j}| = t$. All the input/output shares which are necessary to the simulations of IOS gadgets are input/output shares of SA regions. Moreover, for each input sharing and output sharing of each SA region, we need exactly t shares. For a SA region G_j we denote W'_j the set of wires corresponding to those input/output shares.

The region probing simulator then performs a simulation of $W_j \cup W'_j$ for each SA region G_j, which we explain hereafter. Note that all the input sharings of the SA regions are output of IOS gadgets and are hence uniform sharing mutually independent of each other. This implies that all these SA region simulation can be done independently. Then, the set $W' = W_1 \cup \ldots \cup W_m$ contains all the input-output shares which are necessary to the simulation of the IOS gadgets. By calling the second IOS simulators $\mathsf{Sim}_2^{\text{ios-}G_j}$ for each IOS gadget G_j with relevant input shares from W', we get a perfect simulation of the wires W_j of IOS gadgets G_j. We thus have a full simulation of W.

It remains to show how to perform a perfect simulation of the wires $W_j \cup W'_j$ for an SA region G_j. This can be shown with a reasoning close to the linear algebra characterization of $\mathsf{tightPROVE}$ [12]. Let us denote $\overrightarrow{x_1}, \ldots, \overrightarrow{x_{\ell_j}}$ the input sharings of G_j and let define

$$\overrightarrow{X_1} := (x_{1,1}, x_{2,1}, \ldots, x_{\ell_j,1})$$
$$\vdots$$
$$\overrightarrow{X_n} := (x_{1,n}, x_{2,n}, \ldots, x_{\ell_j,n})$$

namely $\overrightarrow{X_i}$ is the vector with coordinates the ith share of each input sharing. Because G_i is sharewise affine, each wire w of G_i can be expressed as $w = \phi_w(\overrightarrow{X_i})$ for some $i \in [1:n]$ and for ϕ_w an affine function. Now, let us define the set

$$I_{W_j} := \{i \ \mid \ \exists w \in W_i \text{ s.t. } w = \phi_w(\overrightarrow{X_i})\} \ .$$

Namely, a share index $i \in [1:n]$ is included in I_{W_j} iff one probed wire in W_j is of share index i (*i.e.* it is an affine function of the ith shares of the input sharings). By definition of the set I_{W_j}, we can simulate all the probed wires in W_j from the shares $(\overrightarrow{X_i})_{i \in I_{W_j}}$. To complete the proof, we will show that we can perform a perfect joint simulation of the shares $(\overrightarrow{X_i})_{i \in I_{W_j}}$ and all the input-output shares in W'_j (from which we can then derive a perfect simulation of the wires in $W_j \cup W'_j$).

Let us denote by $\overrightarrow{g_k} \in \mathbb{K}^{\ell_j}$ the vector such that the kth output y_k of the SA region is defined as

$$y_k = \langle \overrightarrow{g_k}, (x_1, \ldots, x_{\ell_j}) \rangle$$

where $x_1, \ldots, x_{\ell_j}$ are the ℓ_j input of the SA region and where, for the sake of simplicity, we assume that the constant of the affine function is zero (this does not change the following reasoning). Each output share, $y_{k,i}$ of the SA region can then be expressed as $y_{k,i} = \langle \overrightarrow{g_k}, \overrightarrow{X_i}) \rangle$. From those notations, we have that all the values in $(\overrightarrow{X_i})_{i \in I_{W_j}}$ and W'_j (which we aim to simulate) can be expressed in terms of the following matrix-vector products:

$$M_1 \cdot \overrightarrow{X_1} \ , \quad M_2 \cdot \overrightarrow{X_2} \ , \quad \ldots \ , \quad M_n \cdot \overrightarrow{X_n} \ ,$$

where

- M_i is the identity matrix if $i \in I_{W_j}$,
- the row vector
$$\overrightarrow{e_k} = (\underbrace{0, \ldots, 0}_{k-1 \text{ times}}, 1, \underbrace{0, \ldots, 0}_{n-k \text{ times}})$$
is in M_i if $x_{i,k}$ (the ith share of the kth input sharing) belongs to W'_j (the probed input-output shares),
- the row vector $\overrightarrow{g_k}$ is in M_i if $y_{i,k}$ (the ith share of the kth output sharing) belongs to W'_j (the probed input-output shares).

Now, from [12, Lemma 2] we know that $(M_1 \cdot \overrightarrow{X_1},\ M_2 \cdot \overrightarrow{X_2},\ \ldots,\ M_n \cdot \overrightarrow{X_n})$ can be perfectly simulated whenever

$$\langle M_1 \rangle \cap \langle M_2 \rangle \cap \ldots \langle M_n \rangle = \emptyset$$

where $\langle M_i \rangle$ denotes the linear span of the rows of M_i. We refer to [12] for a proof of the fact. The intuition is that if those spans do not intersect, then all the variables in $(M_1 \cdot \overrightarrow{X_1},\ M_2 \cdot \overrightarrow{X_2},\ \ldots,\ M_n \cdot \overrightarrow{X_n})$ are uniform and independent (for full rank M_i's which is wlog).

Since M_i is the identity matrix for all $i \in I_{W_j}$, we have

$$\bigcap_{i\in[1:n]} \langle M_i \rangle = \bigcap_{i\in[1:n]\setminus I_{W_j}} \langle M_i \rangle \ .$$

Now observe that we have more than $2t$ terms in the above intersection (since $[1:n] \setminus I_{W_j} = n - t > 2t$). Moreover

- all the $\overrightarrow{e_k}$'s are linearly independent vectors and each of them goes in at most t matrices M_i (this is because we have t probes on $\overrightarrow{x_k}$ in W'_j),
- all the $\overrightarrow{g_k}$'s are linearly independent vectors and each of them goes in at most t matrices M_i (this is because we have t probes on $\overrightarrow{y_k}$ in W'_j).

We deduce that $\bigcap_{i\in[1:n]\setminus I_{W_j}} \langle M_i \rangle = \emptyset$. Therefore, thanks to [12, Lemma 2], we can perfectly simulate $(M_1 \cdot \overrightarrow{X_1},\ M_2 \cdot \overrightarrow{X_2},\ \ldots,\ M_n \cdot \overrightarrow{X_n})$ which is equivalent to perfectly simulate $(\overrightarrow{X_i})_{i\in I_{W_j}}$ and W'_j. $\square$

We now prove Proposition 9.

Proof. Recall that $G : (x_1, x_2) \mapsto y$ is t-probing secure gadget and G_{ref} is a t-IOS refresh gadget. We show that

$$G' : (x_1, x_2) \mapsto G_{\mathsf{ref}}\big(G\big(G_{\mathsf{ref}}(\overrightarrow{x_1}), G_{\mathsf{ref}}(\overrightarrow{x_2})\big)\big)$$

is t-IOS. Let us further define a few notations:

$$\begin{aligned} \overrightarrow{x_1}' &= G_{\mathsf{ref}}(\overrightarrow{x_1}) \\ \overrightarrow{x_2}' &= G_{\mathsf{ref}}(\overrightarrow{x_2}) \\ \overrightarrow{y} &= G\big(\overrightarrow{x_1}', \overrightarrow{x_2}'\big) \\ \overrightarrow{y}' &= G_{\mathsf{ref}}(\overrightarrow{y}) \end{aligned}$$

Consider a set of probes $W = W_1 \cup W_2 \cup W_3 \cup W_4$, where W_1 denotes the probes on the refreshing of $(\overrightarrow{x_1})$, W_2 denotes the probes on the refreshing of $(\overrightarrow{x_2})$, W_3 denotes the probes on the refreshing of $(\overrightarrow{y})$, and where W_4 denotes the probes on G. We will further denote $t_i = |W_i|$.

We show that for any such set of probes W and any admissible triple $(\overrightarrow{x_1}, \overrightarrow{x_2}, \overrightarrow{y}')$, we can perfectly simulate the wires in W constrained to the values of $(\overrightarrow{x_1}, \overrightarrow{x_2}, \overrightarrow{y}')$. Using the first IOS simulator of G_{ref}, we get

- two sets I_1 and J_1, with $|I_1|, |J_1| \leq t_1$, such that the probes in W_1 can be perfectly simulated from $\overrightarrow{x_1}|_{I_1}$ and $\overrightarrow{x_1}'|_{J_1}$,
- two sets I_2 and J_2, with $|I_2|, |J_2| \leq t_2$, such that the probes in W_2 can be perfectly simulated from $\overrightarrow{x_2}|_{I_2}$ and $\overrightarrow{x_2}'|_{J_2}$,
- two sets I_3 and J_3, with $|I_3|, |J_3| \leq t_3$, such that the probes in W_3 can be perfectly simulated from $\overrightarrow{y}|_{I_3}$ and $\overrightarrow{y}'|_{J_3}$.

We now define

$$I = I_1 \cup I_2 \cup J_3 \ .$$

The first-step (balanced) IOS simulator for G' returns the above set I. The second-step (balanced) IOS simulator receives $\overrightarrow{x_1}|_I$, $\overrightarrow{x_2}|_I$, and $\overrightarrow{y}'|_I$ and must output a perfect simulation of W which is consistent with any admissible triple $(\overrightarrow{x_1}, \overrightarrow{x_2}, \overrightarrow{y}')$. It goes as follows. It first call the probing secure simulator of G to get a perfect simulations of

$$W_4 \cup \{\overrightarrow{x_1}'|_{J_1}\} \cup \{\overrightarrow{x_2}'|_{J_2}\} \cup \{\overrightarrow{y}|_{I_3}\}$$

We note that the above set contains $t_1 + t_2 + t_3 + t_4 = |W| \leq t$ wires hence the probing secure simulator succeeds. then

- from $\{\overrightarrow{x_1}|_{I_1}\} \subseteq \{\overrightarrow{x_1}|_I\}$ (input of the IOS simulator of G') and $\overrightarrow{x_1}'|_{J_1}$ (output of the probing secure simulator of G) we can perfectly simulate the probed wires in W_1 thanks to the IOS simulator of G_{ref},
- from $\{\overrightarrow{x_2}|_{I_2}\} \subseteq \{\overrightarrow{x_2}|_I\}$ (input of the IOS simulator of G') and $\overrightarrow{x_2}'|_{J_2}$ (output of the probing secure simulator of G) we can perfectly simulate the probed wires in W_2 thanks to the IOS simulator of G_{ref},
- from $\{\overrightarrow{y}'|_{J_3}\} \subseteq \{\overrightarrow{y}'|_I\}$ (input of the IOS simulator of G') and $\overrightarrow{y}|_{I_3}$ (output of the probing secure simulator of G) we can perfectly simulate the probed wires in W_3 thanks to the IOS simulator of G_{ref}.

We thus get a perfect simulation of all the probed wires in $W = W_1 \cup W_2 \cup W_3 \cup W_4$ from $\overrightarrow{x_1}|_I$, $\overrightarrow{x_2}|_I$, and $\overrightarrow{y}'|_I$ for any admissible triple $(\overrightarrow{x_1}, \overrightarrow{x_2}, \overrightarrow{y}')$. Namely, G' is balanced t-IOS. $\square$