University of Warsaw
Faculty of Mathematics, Informatics and Mechanics

mgr Karol Żebrowski

Analysis of the Simple Refreshing in the Noisy Leakage Model

PhD dissertation

Supervisor
prof. dr hab. Stefan Dziembowski
Institute of Informatics
University of Warsaw

June 2024

Author's declaration:
aware of legal responsibility I hereby declare that I have written this dissertation myself and all the contents of the dissertation have been obtained by legal means.

June, 2024 .....

mgr Karol Żebrowski

Supervisor's declaration:
The dissertation is ready to be reviewed.

June, 2024 .....

prof. dr hab. Stefan Dziembowski

Abstract

Masking schemes are a prominent countermeasure against power analysis and work by concealing the values that are produced during the computation through randomness. The randomness is typically injected into the masked algorithm using a so-called *refreshing* scheme, which is placed after each masked operation, and hence is one of the main bottlenecks for designing efficient masking schemes. The dissertation investigates the security of a very simple and efficient refreshing scheme and prove its security in the *noisy leakage model*. Compared to earlier constructions our refreshing is significantly more efficient and uses only $n - 1$ random values and $< 2n$ operations, where n is the security parameter. It is also more practical, as the proof gives meaningful results even for small security parameter n. In addition we show how our refreshing can be used in more complex masked computation in the presence of noisy leakage.

The dissertation presents a new, invented by the author, methodology for analyzing and proving the security of the masked computation with a simple refreshing, that we call a *leakage diagram*. The results of this dissertation were partially presented in the paper *Simple Refreshing in the Noisy Leakage Model* [18] by Stefan Dziembowski, Sebastian Faust and Karol Żebrowski. In comparison to the paper, the dissertation contains also a detailed proof of the security of our construction and presents the technical tools used in the proof.

Key words: leakage-resilient cryptography, side-channel attack, masking scheme

AMS Classification: 94A60, 68P20, 68P30

Streszczenie

Schematy maskujące stanowią znaczący środek zaradczy zapobiegający analizie mocy i działają poprzez ukrywanie wartości generowanych podczas obliczeń używając losowości. Losowość jest zazwyczaj wprowadzana do maskowanego algorytmu przy użyciu tak zwanego schematu *odświeżającego*, który jest umieszczany po każdej maskowanej operacji, dlatego stanowi on jedno z głównych wąskich gardeł w projektowaniu wydajnych schematów maskujących. Rozprawa bada bezpieczeństwo bardzo prostego i wydajnego schematu odświeżania i dowodzi jego bezpieczeństwa w modelu *zaszumionego wycieku*. W porównaniu do wcześniejszych konstrukcji nasze odświeżanie jest znacznie wydajniejsze i wykorzystuje tylko $n-1$ losowych wartości oraz $< 2n$ operacj, gdzie n jest parametrem bezpieczeństwa. Ponadto pokazujemy, jak nasze odświeżanie może zostać wykorzystane w bardziej złożonych maskowanych obliczeniach w obecności zaszumionego wycieku.

Rozprawa doktorska przedstawia nową, wynalezioną przez autora, metodologię analizy i dowodu bezpieczeństwa obliczeń maskowanych z prostym odświeżaniem, nazywną przez nas *diagramem wycieku*. Wyniki z rozprawy były częściowo przedstawione w artykule *Simple Refreshing in the Noisy Leakage Model* [18] autorstwa Stefana Dziembowskiego, Sebastiana Fausta i Karola Żebrowskiego. W porównaniu do artykułu, rozprawa zawiera również szczegóły dowodu bezpieczeństwa oraz przedstawia techniczne narzędzia użyte w dowodzie.

Słowa kluczowe: leakage-resilient cryptography, side-channel attack, masking scheme

Klasyfikacja AMS: 94A60, 68P20, 68P30

Contents

Chapter 1

Introduction

Traditional security proofs in cryptography assume the existence of physical devices that are completely safe against the attacker, and that the parties use such devices for the computation required by the cryptographic protocol. This provides them with a theoretical privacy of local computations. For example, in the chosen-plaintext attack model the adversary has access to an encryption oracle and can ask for the ciphertext of arbitrary plaintext messages. Here, it is assumed that the adversary has only a *black-box* access to the oracle, which means that the computation of the ciphertexts from the plaintexts does not reveal *any information* on the secret key.

Unfortunately, in practice it is difficult to construct a physical device that ensures *no leakage* of information during computation, for example when signing a message with a private key. There are many known practical attacks on cryptographic protocols that exploit this additional information on the internal state of the device.

1.1 Side-channel attacks

The so-called *side-channel attacks* are cryptanalytic techniques which target the cryptographic hardware. They exploit various physical phenomena that occur during computation on a physical device, which allow to extract some additional information about the sensitive data, including the secret key, from a cryptographic implementation. Thereby they violate the fundamental assumption of cryptography. Timing attacks [29] rely on measuring the time taken by the computation. Also electromagnatic radiation during

computation on a smart card can lead to an attack [21]. Some attacks [31, 28] show even how isolated processes can read information from other processes running on the same machine.

Power analysis One of the most powerful side-channel attack is the so-called power analysis, introduced by Kocher et al. [30]. Power analysis exploits that the power consumption of an unprotected physical device is correlated with the internal computation – and in particular with the secret key. There are countless examples that show how an adversary can use these correlations to learn information about the secret key (see, e.g., [36] for an introduction to this topic).

For instance, a very well known attack on a concrete implementation of the RSA algorithm exploits the fact that it proceeds in a loop over the bits of the secret key, and depending the bit executes different operations. If the bit is 1 it performs one squaring operation and one multiplication, and if the bit is 0 just one multiplication is performed. The attacker observing the power consumption trace can easily distinguish between these two cases, and therefore recover the whole secret bit by bit. Essentially, what allows for this attack is the correlation between the side-channel observations and the internal computation of a device, for example a smart card.

These examples clearly show that the real world adversary has to be modeled as having more than just a black box access to the cryptosystem. To ensure security even in the presence of such adversary, the area of *leakage-resilient cryptography* emerged.

1.2 Leakage-resilient cryptography

In recent years cryptographic research has made tremendous progress in developing solid foundations for cryptography in the presence of side-channel leakage (see, e.g., [27] for an overview). The common approach in this area – often referred to as "leakage resilient cryptography" – is to first extend the black-box model to incorporate side-channel leakage, and then to propose countermeasures that are provable secure within this model. The typical leakage model considered in the literature assumes an adversary that obtains some partial knowledge about the internal state of the device. For instance,

the adversary may learn a few bits of the intermediate values that are produced by the device during its computation.

The main challenge in choosing a leakage model is to maintain a balance between practice and theory. On one hand, the model should be broad enough to encapsulate the real-world physical leakage accurately. On the other hand, it should be restrictive enough so that a formal security analysis is still possible. This has sometimes lead to the development of countermeasures that are overly complex to defeat artificial attacks. One prominent example for such a case is the "alternating structure" frequently used to design leakage resilient stream ciphers [19, 33]. Although much simpler constructions are believed to exhibit the same level of practical side-channel resistance [37], an additional complexity – the alternating structure – was introduced into the design to defend against the highly artificial "pre-computation attack" [19].

Roughly speaking, the field of leakage-resilient cryptography can be divided into two strands: first incorporating into the model the leakage from memory, and the second incorporating the leakage from computation. In the first category it is assumed that the adversary is gaining some *partial* information about the secret key, and the goal is to build cryptographic schemes secure in such model. As an example, in the *bounded retrieval model*, introduced by Dziembowski [15] and Di Crescenzo et al. [12], adversary can obtain an arbitrary polynomial-time computable leakage function of the secret key, but the output size of this leakage function is bounded.

On the other hand, leakage from computation, which is the main focus of this dissertation, assumes that the adversary has access to the entire computation rather than just the secret memory. Therefore, to provide some security guaranties, in this category the leakage model typically has to be more restricted.

The effort to provide countermeasures against leakage from computation can be roughly divided into two categories: building secure (in the presence of the side-channel leakage) specific cryptographic primitives, e.g., stream ciphers [19], and building a generic compiler that is able to provide security for any computations, including these required by the cryptographic primitive.

General circuit compilers The goal of a general leakage-resilient compiler is to compile an arbitrary circuit C into a transformed circuit $\widehat{C}$ with the same functionality that is secure in a given leakage model. To model the leakage from a cryptographic device, the circuit C is usually assumed to

be stateful, and the state is assumed to contain some sensitive information, like a secret key. The adversary has input/output access to the circuit $\widehat{C}$ and runs it repeatedly, additionally observing the side-channel leakage. The securtiy definition is simulation-based: the construction is said to be secure if the leakage observed by the adversary can be produced by a simulator that only observes the input/output behavior of the circuit.

In their seminal work Ishai, Sahai and Wagner [25] achieve a general compiler secure in the *t-probing model*. In this model the adversary can choose up to t wires of the circuit and probe them obtaining their values during computation. A long line of the following work is focused on building circuit compilers under various assumptions. Some works consider only a leakage that, as a function of the internal wires of the circuit, is of limited complexity. For example, in [20] it is assumed that the leaking funtion is in the class AC^0, and that its range is bounded. The compilers proposed in the literature often assume also a leak-free components in their constructions [20, 34, 26, 22].

It worth noting that building the compilers for the *stateless* circuits is also very useful. Indeed, the transformation of Ishai, Sahai and Wagner [25] for stateful circuits is based on a compiler for the stateless ones. The state is simply treated as an additional input to the stateless circuit, and it is updated according to its computed output.

One of the most widely used approach to building a leakage-resilient circuit compiler is through the use of so called *masking schemes*.

1.2.1 Masking schemes

One of the most common countermeasures against power analysis attacks are masking schemes (see, e.g., [25, 34, 13, 32, 6, 10, 1, 8, 5, 9]). In order to de-correlate the internal computation of a device from the observable leakage, masking schemes randomize the intermediate values produced during the computation of an algorithm through secret sharing. To this end each sensitive variable x is represented by an encoding $\mathrm{Enc}(x) := (x_1, \ldots, x_n)$ and the corresponding decoding function $\mathrm{Dec}(\cdot)$ recovers $x := \mathrm{Dec}(x_1, \ldots, x_n)$. A simple encoding function uses the additive encoding function, which works by sampling x_i uniformly at random from some finite field $\mathbb{F}$ subject to the constraint that $x := \sum_{i=1}^{n} x_i$. If $\mathbb{F}$ is the binary field, then such a masking scheme is typically called *Boolean masking*.

In addition to an encoding scheme, we need secure algorithms to compute with encoded elements. To this end, the algorithm's computation is typically modeled as an *arithmetic circuit* over a finite field $\mathbb{F}$. In such circuits the wires carry values from $\mathbb{F}$ and the gates perform operations from $\mathbb{F}$. At a high-level the circuit is made out of gates that represent the basic field operations (i.e., addition gate denoted "$\oplus$" and multiplication gate denoted "$\otimes$"). Moreover, it may consist of gates for inversion (i.e., outputting $-x$ on input x), and so-called randomness gates *RND* that take no input and produce an output that is distributed uniformly over $\mathbb{F}$.

Given a circuit built from these gates, a masking scheme then typically works by replacing each of the above operations by a "masked" version of the gate, a so called *gadget*. For instance, in case of the aforementioned additive encoding scheme $(\mathrm{Enc}, \mathrm{Dec})$ the masked version of the $\oplus$ takes as input two encodings $\mathrm{Enc}(x)$ and $\mathrm{Enc}(y)$ and outputs an encoding $\mathrm{Enc}(z)$, where $\sum_i z_i := \sum_i x_i + \sum_i y_i$. Informally, the gadget is said to be secure if the leakage emitted from its internal computation does not reveal any sensitive information. The exact definition depends on the circuit construction and the leakage model considered.

1.2.2 Refreshing schemes

A core ingredient of many secure masking schemes is a *refreshing* algorithm. At a very high level the refreshing algorithm introduces new randomness into the masked computation, thereby preventing an adversary from exploiting correlations between different intermediate values of the computation. Since refreshing schemes are computationally expensive a large body of work has explored how to securely improve their efficiency. One of the most simple and efficient (in terms of computation and randomness) refreshing schemes was proposed by Rivain and Prouff [35]. Unforunately, it was shown in [11] that a simple – though impractical – attack breaks the scheme in the common *threshold probing* leakage model [25].

The *refreshing scheme* takes as input an encoding $\vec{x}^j := (x_1^j, \ldots, x_n^j) = \mathrm{Enc}(x)$ and outputs a new encoding $(x_1^{j+1}, \ldots, x_n^{j+1}) = \vec{x}^{j+1}$ of x. By "new encoding" we mean that this procedure should inject new randomness into the encoding, in such a way that the leakage from the previous encodings should not accumulate. In other words: if we periodically refresh the encodings of x (which leads to a sequence of encodings: $\vec{x}^0 \mapsto \vec{x}^1 \mapsto \vec{x}^2 \mapsto \cdots$) then x should remain secret even if bounded partial information about each

$\vec{x}^j$ leaks to the adversary. The operations of computing $\vec{x}^{j+1}$ from $\vec{x}^j$ is also called a *refreshing round*, and a circuit that consists of some number of such rounds (and not other operations) is called a *multi-round refreshing circuit*.

Simple refreshing. A common approach for securely refreshing additive encodings is to exploit the homomorphism of the underlying encoding with respect to addition. By this we mean that for every x and y we have $\mathrm{Dec}(\mathrm{Enc}(x) + \mathrm{Enc}(y)) = x + y$, where "+" on the left-hand-side denotes the vector addition. One starts by designing an algorithm that samples $(b_1, \ldots, b_n)$ from the distribution $\mathrm{Enc}(0)$, and then, in order to refresh an encoding $(x_1^j, \ldots, x_n^j)$ one adds $(b_1, \ldots, b_n)$ to it. Therefore, the refreshed encoding is equal to $(x_1^j + b_1, \ldots, x_n^j + b_n)$. Observe that after $\mathrm{Enc}(0)$ is generated, the refreshing can be done without any further computation, by just adding b_i to every x_i^j. Of course in this approach the whole technical difficulty is to generate the encodings of 0 in a secure way (without relying on any assumptions on leakage-freeness of the encoding generation).

The most simple and efficient refreshing scheme originally introduced in [35] uses the "encoding of 0 approach" mentioned above and works as follows (see also Fig. 2.1 on page 21). In order to refresh $\vec{x}^j = (x_1^j, \ldots, x_n^j)$, we first sample $b_1^j, \ldots, b_{n-1}^j$ uniformly at random from $\mathbb{F}$ and set $b_n^j := -b_1^j - \ldots - b_{n-1}^j$. Then, we compute the fresh encoding of x as $(x_1^{j+1}, \ldots, x_n^{j+1}) := (x_1^j + b_1^j, \ldots, x_n^j + b_n^j)$. Notice that besides its simplicity the above refreshing enjoys additional beneficial properties including optimal randomness complexity (only $n - 1$ random values are used) and minimal circuit size (only $2n - 1$ field operations are required). Somewhat surprisingly this simple refreshing scheme turns out to be insecure in the security model of *threshold probing attacks* introduced in the seminal work of Ishai, Sahai and Wagner [25].

Insecurity of simple refreshing. The standard model to analyze the security of masking schemes is the *t-probing model* [25]. In the t-probing model the adversary can (adaptively) select up to t wires of the internal masked computation and learn the values carried on these wires during computation. While originally it was believed that the simple refreshing from above guarantees security for $t = n - 1$ [35], Coron et al. [11] showed that when it is combined with certain other masked operations (e.g., in a masked AES) the resulting construction can be broken using only $\leq t := n/2 + 1$

probes.

An even more devastating attack against this natural refreshing can be shown in the following setting. Consider a circuit that consists of a sequence of n refreshings of an encoding $\vec{x}^0$. This may naturally happen in a masked key schedule of the AES algorithm, where the secret key is encoded and after each use for encrypting/decrypting is refreshed. If for each of these refreshings the adversary can learn 2 values, then a simple attack allows to recover the secret (we describe this attack in more detail in Chapter 2). The attack, however, is rather impossible to carry out in practice. In particular, it requires the adversary to learn for the n consecutive executions of the refreshing scheme specific (different) intermediate values.

Other refreshing schemes and their usage. The most often used refreshing scheme proposed by Ishai, Sahai and Wagner [25] is based on "artificial" multiplication by 1: the encoding of a secret value is simply multiplied, using the multiplication gadget, with a *fixed* encoding of 1. Therefore the refreshing complexity, in terms of randomness and computation, is the same as for the multiplication and requires $O(n^2)$ random values and $O(n^2)$ gates.

The randomness consumption is often the bottleneck for an efficient masked implementation. True randomness is hard to generate in practice, and producing securely pseudorandomness is costly as we need to run, e.g., an AES algorithm. Hence, an important goal of research is to minimize the overheads resulting from the use of refreshing. There are two main directions to achieve this. First, we may improve the refreshing algorithm itself. In particular, in [1, 3] it was shown how to build a secure refreshing with circuits size and randomness complexity $O(n)$, where n is the security parameter. While asymptotically optimal from a concrete practical point of view these schemes are very inefficient as they are based on expander graphs and require n to be impractically large. A second direction to improve on the costs for refreshing is to reduce the number of times the refreshing algorithms are used. This approach was taken by several works [6, 10, 8] which develop tools for placing the refreshing algorithm in an efficiency optimizing way without compromising on security.

As the attacks on the simple refreshing in the probing leakage model are rather impractical, it leads to a natural question: *Can we prove the security of the simple refreshing scheme in a weaker model?*

1.2.3 Noisy leakage and p-random probing models

The attack against the simple refreshing illustrates that in some sense the probing model is too strong. An alternative model is the so-called *noisy leakage model* of Prouff and Rivain [34]. In the noisy leakage model the leakage is not quantitatively bounded but instead it is assumed that the adversary obtains a "noisy distribution" of each value carried on a wire (see Sect.3.4). The noisy leakage model is believed to model real-world physical leakage accurately, because such leakages are inherently noisy, and hence it is prominently used in practice to analyze the real-world security of physical devices [14].

In [13] it was shown that the noisy leakage model of [34] can be reduced to the *p-random probing model*. In the p-random probing model we assume that the value carried on each wire is revealed independently with probability p. Since in the p-random probing model the adversary looses control over the choice of wire that he learns, the attack against the simple refreshing ceases to work. This raises the question if the simple refreshing scheme is secure in the p-random probing model, and therefore in the noisy leakage model. The main contribution of this dissertation is to answer this question affirmatively, with security guaranties for constant probability p (independent of the security parameter n).

In contrast, the refreshing scheme of Ishai et al. [25] (mentioned already in Sect. 1.2.2), for security parameter n, results into a circuit of size $O(n^2)$, while only tolerating that wires leak with probability $p \approx 1/n$ (in the p-random probing model). In addition, this refreshing scheme consumes $O(n^2)$ randomness, which in practice will be the main bottleneck for using masking schemes due to high costs for securely generating randomness.

Secure computation in the noisy leakage model. Unfortuantelly both constructions for secure computation in [34, 13] require $p \approx 1/n$, and thus assume the noise to *decrease* with an increase of the security parameter n. Therefore, one important goal of research is to improve the noise parameter p. There has recently been significant progress on this. In [1, 3] it was shown how to securely compute in the random probing model for constant p. Further improvements are made in [2, 23], where the latter achieves security under a quasi-constant noise for a construction with complexity $O(n \log(n))$ avoiding heavy tools such as expander graphs and AG codes. Another line of work investigates relations between different noisy leakage models [17, 24]

and provides tight relations between them. A more practical view on noisy leakage – and in particular a quantitative study of its relation to real-world leakage – was given by Duc et al. [14].

1.3 Results

Here we give a high-level summary of the results presented in this dissertation.

Simple refreshing analysis. Our main contribution is to analyze the security of the simple refreshing scheme from [35] in the noisy leakage model. In particular, we show that refreshing an encoding $(x_1, \ldots, x_n)$ is secure even if each wire in the refreshing circuit is revealed with constant probability p. Our result directly implies that refreshing an encoded secret k times (where k may be much larger than the security parameter n) remains secure under noisy leakages for constant noise parameter. Such consecutive use of refreshings naturally appears in many practical settings such as the key schedule of the AES mentioned above, or in general for refreshing the secret key between multiple runs of any cryptographic primitive. Since the simple refreshing is *optimal* in terms of circuit size and randomness complexity our result significantly improves the practicality of the masking countermeasure.

Concretely, the simple refreshing requires $n - 1$ random values and uses $2n - 1$ addition gates to securely refresh an encoding $(x_1, \ldots, x_n)$ in the random probing model (and hence implying security in the noisy leakage model of [34]). In contrast, the most widely used refreshing scheme from Ishai, Sahai and Wagner [25] requires $(n-1)^2/2$ randoms and $2n^2 + n$ gates and has been proven secure only for $p \approx 1/n$, which is significantly worse than ours. Various works provide asymptotically improved refreshing algorithms. In particular, in [1, 3] it was shown how to build a secure refreshing with circuit size $O(n)$, and randomness complexity $O(n)$ for a constant noise parameter p. While asymptotically these constructions are the same as for the simple refreshing analyzed in our work, from a practical point of view these schemes are very inefficient as they are based on expander graphs.

New techniques for proving security. At the technical level, our main contribution is to introduce a new technique for proving security in the random probing model. Our main observation is that probing security can be

translated into a question of connectivity between nodes in certain graphs. As an example consider the circuit $\widehat{C}$ executing k times the simple refreshing. It can be represented as a grid G with $k+1$ rows and $n+1$ columns, where in each row we have $n+1$ nodes. The edges between the nodes represent intermediate values that are computed during the execution of the circuit. Leakage of a certain subset of wires then corresponds to a subgraph of G, which we call *leakage diagram*.

A crucial property of such representation of a leakage is the following: if its "leftmost side" and its "rightmost side" are *not connected* by a path in the leakage diagram, then it can be shown that the adversary does not learn any information about the encoded secret from the leakage. The above can be extended to arbitrary masked arithmetic circuits, in which the graphs representing the circuit are slightly more involved.

The above allows us to cast security against probing leakage as a question about connectivity of nodes within a graph. To show security in the p-random probing model we then need to bound the probability that the random sub-graph of G representing the leakage contains a path that connects the two sides of G. The main challenge is that although in the p-random probing model each wire leaks independently with probability p, in our graph representation certain edges are more likely to be part of the leakage diagram. Even worse, the events of particular edges of G ending up in the leakage diagram are not independent. This significantly complicates our analysis. We believe that the techniques introduced in this dissertation are of independent interest and provide a novel tool set for analyzing security of masked computation in the random probing model.

Extension to any masked computation. As our last contribution we show how to use the simple refreshing as part of a more complex masked computation. To this end, we study the security of the masking compiler provided by Ishai, Sahai and Wagner [25] when using the simple refreshing described above. Notably, we first show that the simple refreshing can be used to securely compose any affine masked operations. This result is important because it shows for the first time that the most natural and efficient way to carry out affine computation in the masked domain is secure against noisy leakages. Compared to the standard construction of [25] we save a factor of n in circuit and randomness complexity. Moreover, at the concrete level we make huge practical improvements when compared to the recent works

of [1, 3], which use expander graphs and algebraic geometric codes.

Finally, we show that the simple refreshing can also be securely composed with the masked multiplication of [25]. Since the masked multiplication of [25] itself is a composable refreshing [13], this result is maybe not so surprising.

1.4 Organization of the dissertation

Chapter 2 contains an informal description of our techniques and intuitions behind them. In it, we present analysis of the security in case of the multi-round refreshing circuit in the p-random probing model.

Chapter 3 presents the necessary preliminaries and formal security definitions relevant to our work.

Chapter 4 provides detailed description of our circuit compiler and give a general definition of a gadget that can be composed securely with the simple refreshing.

Chapter 5 contains all the auxiliary notions and their properties, necessary for the proof of our main theorem.

In Chapter 6 we state the main theorem of our work and present its proof.

Most parts of this dissertation are covered by the paper [18]. This paper was supported by the Foundation for Polish Science (grant agreement TEAM/2016-1/4) co-financed with the support of the EU Smart Growth Operational Programme (PO IR).

Acknowledgements

First and foremost I would like to thank my advisor, prof. dr hab. Stefan Dziembowski, for his guidance and invaluable help during the process of writing this dissertation and for co-authoring the research paper on which it is based. I would like to thank prof. Sebastian Faust for collaboration during my PhD studies and co-authoring the research paper. I am grateful also to Dr. Vincenzo Iovino for collaboration and support. Finally, I am grateful to all my friends and family for providing help and motivation during my studies.

Chapter 2

Our approach informally

This Chapter is a slightly extended version of the section with the same title in [18]. Here we describe informally our analysis of the security of multi-round refreshing circuit.

As a simple example of circuit to present our approach let us consider a circuit $\widehat{C}$ (in the following the "hat notation" will denote masked/transformed circuits) that is a k-round refreshing circuit. This circuit consist of k consecutive subcircuits that we call *refreshing gadgets* $\widehat{R}$, presented in Fig. 2.1. Note that in addition to the notation from Sect. 1.2.2 we also use terms c_i^j that denote the partial sums: $c_i^j = b_1^j + \cdots + b_i^j$ (for consistency define c_0^j and c_n^j to be always equal to 0). Also, for $i = 0, \ldots, n$ let T_i^j be the partial sum $T_i^j := x_1^j + \cdots + x_i^j$ (assume that always $T_0^j := 0$). The following simple observation, that easily follows from the construction on Fig. 2.1, will be useful in the sequel.

Claim 1. *For every i and j we have that $T_i^{j+1} = T_i^j + c_i^j$.*

Proof. Transform T_i^{j+1} as follows:

$$\begin{aligned} T_i^{j+1} &= x_1^{j+1} + \cdots + x_i^{j+1} \\ &= (x_1^j + b_1^j) + \cdots + (x_i^j + b_i^j) \\ &= \underbrace{x_1^j + \cdots + x_j^j}_{=T_i^j} + \underbrace{b_1^j + \cdots + b_i^j}_{=c_i^j}. \end{aligned}$$

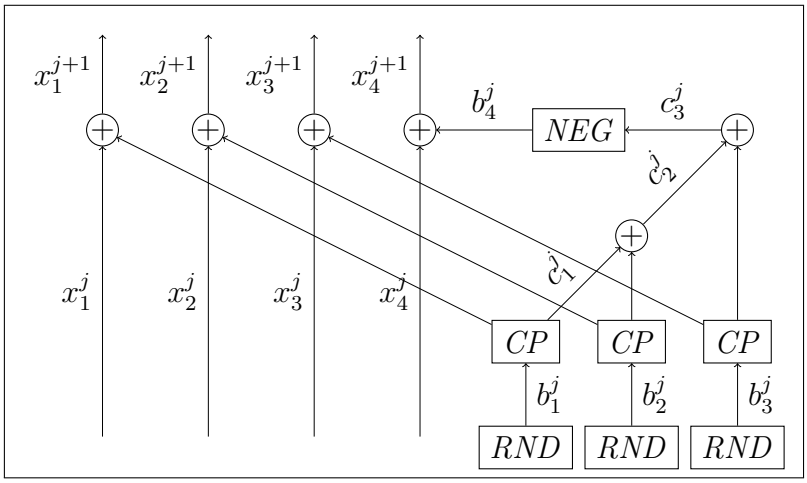
□

We show that the adversary can learn the encoded secret even if just 2 wires from each refreshing gadget leak to her (and no additional leakage

is given). Similar attacks for different refreshing schemes have been shown in [16, 7]. Let $x_1^0, \ldots, x_n^0$ be some initial encoding of x, and consider a k-round

$(b_1^j, \ldots, b_{n-1}^j) \leftarrow \mathbb{F}^{n-1}$
$c_0^j := 0$
for $i = 1, \ldots, n-1$ **do**
  $c_i^j := c_{i-1}^j + b_i^j$
$b_n^j := -c_{n-1}^j$
for $i = 1, \ldots, n$ **do**
  $x_i^{j+1} := x_i^j + b_i^j$

(a) Pseudocode of the simple refreshing gadget $\widehat{R}$.



(b) Corresponding circuit (for $n = 4$).

Figure 2.1: The refreshing gadget. The "j" superscript is added for the future reference (e.g. on Fig. (2.3a)).

refreshing circuit $\widehat{C}$. In the jth round (for $j = 0, \ldots, n-1$), the adversary chooses to learn x_{j+1}^j, and the partial sum c_{j+1}^j[1]. We now show that the following invariant holds.

[1]The reader may notice that strictly speaking leaking the last c_{j+1}^j (i.e.: c_n^{n-1}) is not needed, since anyway, this value is always equal to 0.

Claim 2. *After n rounds the adversary can compute the sum T_n^n.*

After showing this we will be done, since clearly $x = T_n^n$.

Proof. By applying recursively n times Claim 1 alternately with the fact that $T_i^j = T_{i-1}^j + x_i^j$ we get that

$$\begin{aligned} T_n^n &= T_n^{n-1} + c_n^{n-1} \\ &= T_{n-1}^{n-1} + x_n^{n-1} + c_n^{n-1} \\ &\vdots \\ &= T_0^0 + (x_1^0 + c_1^0) + \cdots + (x_n^{n-1} + c_n^{n-1}) \end{aligned} \tag{2.1}$$

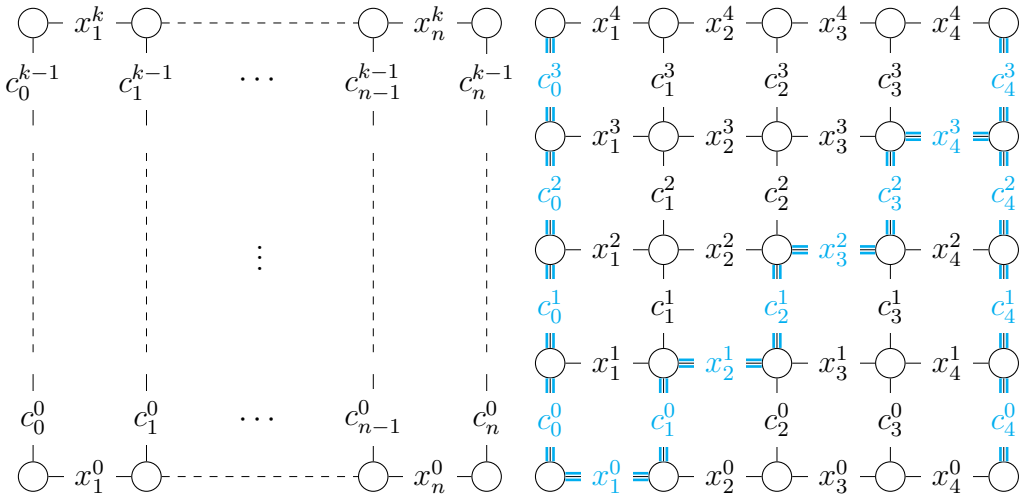
This finishes the proof, since $T_0^0 = 0$ and all the other variables in Eq. (2.1) are known to the adversary. □

We now show how the proof of Claim 2 can be represented "graphically". First draw an $(n+1) \times (n+1)$ grid G with edges labeled with the edges x_i^j and c_i^j as on Fig. (2.2a). Now, assume $n = 4$ and mark every edge that leaked to the adversary with double colored lines (with the exception of c_n^{n-1} which, by the observation from footnote 1 is anyway not relevant to the attack). This is done on Fig. (2.2b). Additionally, observe that the adversary knows every c_0^j and c_n^j "for free" since they are always equal to 0. Therefore we also draw double lines over every c_0^j and c_n^j.

It is easy to see that the proof of Claim 2 went through because the leftmost side of the graph on Fig. (2.2b) (i.e. the path labeled with $c_0^0, c_0^1, \ldots$) is connected by a double line "diagonal" path (labeled with $x_1^0, c_1^0, x_2^1, c_2^1, \ldots, c_3^2, x_4^3$) with the rightmost path (labeled with $c_n^0, c_n^1, \ldots$). Indeed, what our inductive proof essentially shows (for $n = 4$) is that

$$\begin{aligned} & x_1^0 + c_1^0 + x_2^1 + c_2^1 + \cdots + c_3^2 + x_4^3 \\ = \; & x_1^4 + \cdots + x_4^4 \; (= x). \end{aligned}$$

Relaxing the leakage model. It is easy to see that the attack described above strongly relies on the fact that the adversary can *choose* which wires he learns, since only then she is able to gradually learn the values of $T_1^1, T_2^2, \ldots$. As discussed in the introduction, in the weaker p-random probing model it is very unlikely that the adversary will be lucky enough to learn x_{j+1}^j and c_{j+1}^j in each round (unless p is close to 1). Of course, the fact that one



(a) Graph G corresponding to the k-round refreshing circuit. It has $k+1$ rows. In each jth row (for $j = 0, \ldots, k$) it has $n+1$ vertices connected with edges (there is an edge labeled with "x_i" between the ith and $(i+1)$st vertex). It also has an edge between every pair of ith vertices (for $i = 0, \ldots, n$) in the jth and $j+1$st row. This edge is labeled with "c_i^j".

(b) The nodes and the lines represent the diagram that corresponds to the proof of Claim 2 (see remarks after the proof of this claim). The colored double lines correspond to the wires that leaked to the adversary, or are known to him because they are always equal to 0 (i.e.: the c_0^j's and the c_4^j's).

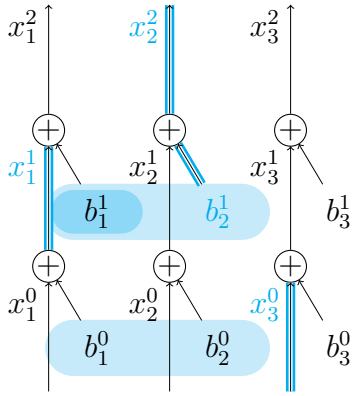
Figure 2.2: Graph G and a leakage diagram.

particular attack does not work, does not immediately imply that the scheme is secure. As already mentioned in Sect. 1.3 our first main contribution is a formal proof that indeed this simple refreshing procedure is secure in the p-random probing model. Our starting point is the natural question: *can we characterize the leakages which allow the adversary to compute the secret?* We answer this question affirmatively by introducing the notion of *leakage diagrams*, which we explain below (for formal definitions see Sect. 5.2).

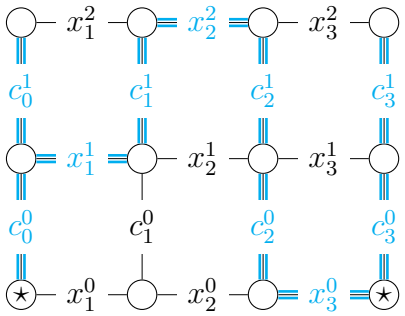
Leakage diagrams. Essentially, the leakage diagrams are graphs that can be viewed as abstract representations of the leakage that occurred during the evaluation of a circuit. For a moment let us focus only on leakage diagrams

that correspond to k-round refreshing circuits $\widehat{C}$. Let $x_1^0, \ldots, x_n^0$ be some initial encoding of the secret x. In this case the leakage diagram will be a subgraph of a $(n+1) \times (k+1)$ grid G with edges labeled x_i^j and c_i^j as on Fig. (2.2a).

To illustrate how the leakage diagrams are constructed take as an example a 2-round refreshing circuit (with $n = 3$) that is depicted on Fig. (2.3a). Note that this picture omits the part of circuit that is responsible for generating the b_i^j's, and in particular the wires carrying the c_i^j values are missing on it. This is done in order to save space on the picture. Let L be the wires that



(a) A circuit with leaking wires marked with colored double lines. Additionally wires $c_2^0 (= b_1^0 + b_2^0)$, $c_1^1 (= b_1^1)$, and $c_2^1 (= b_1^1 + b_2^1)$ leak, which is indicated by colored shaded areas around "$b_1^0\ b_2^0$", "b_1^1", "$b_1^1\ b_2^1$".



(b) The corresponding leakage diagram. We show how the adversary can compute the sum of edges x_1^0, x_2^0, and x_3^0. The leftmost and the rightmost vertices of the row containing these edges are marked with "$\circledast$".

Figure 2.3: A leaking circuit and its corresponding leakage diagram.

leaked in the refreshing procedure. Suppose the leaking wires are x_3^0, x_1^1, x_2^3, and b_2^1, which is indicated by double color lines over the corresponding edges on Fig. (2.3a). We also have to remember about the c_i^j's that were omitted on the figure and can also leak. Recall that every c_i^j is equal to a sum $b_1^j + \cdots + b_i^j$. Hence, the leakage from c_i^j is indicated by a shaded colored region around $b_1^j, \ldots, b_i^j$. Let us assume that c_2^0, c_1^1, and c_2^1 are leaking, and therefore the shaded regions on Fig. (2.3a) are placed over b_1^1, and the pairs $(b_1^0, b_2^0), (b_1^1, b_2^1)$.

The corresponding leakage diagram is a subgraph of the graph G from

Fig. (2.2a) with $k := 2$ and $n := 3$. The leakage diagram $S(L)$ has the same vertices as G, but it has only a subset of its edges. Informally, the labels on the edges of $S(L)$ are variables that suffice to fully reconstruct the leakage from the circuit. More precisely: given these values one can compute the same leakage information that the adversary received. Going back to our example: the leakage diagram corresponding to the leakage presented on Fig. (2.3a) is depicted on Fig. (2.3b), on which the members of $S(L)$ are marked with double colored lines. The set $S(L)$ is created according to the following rules. First, we add to $S(L)$ all the edges labeled x_i^j and c_i^j if the corresponding wires are in L. For this reason $S(L)$ on Fig. (2.3b) contains $x_3^0, x_1^1, x_2^2, c_2^0, c_1^1$, and c_2^1. Handling leaking b_i^j's is slightly less natural, since graph G does not contain edges labeled with the b_i^j's. To deal with this, we make use of the fact that every b_i^j can be computed from c_i^j and c_{i-1}^j (as $b_i^j = c_i^j - c_{i-1}^j$). Hence, for every b_i^j from L we simply add c_i^j and c_{i-1}^j to $S(L)$. For this reason we add c_1^1 and c_2^1 to L (as b_2^1 is in L). This approach works, since, as mentioned above, the edges in $S(L)$ should suffice to fully reconstruct L. Note that in some sense we are "giving out too much" in the leakage diagram (as c_i^j and c_{i-1}^j cannot be uniquely determined from b_i^j). Fortunately, this "looseness" does not cost us much in terms of parameters, while at the same time it greatly simplifies our proofs. Finally, we add to $S(L)$ all the edges labeled with c_0^j and c_n^j (i.e.: the leftmost and the rightmost columns in G). We can do it since these edges are always equal to 0 and hence the adversary knows them "for free".

What the adversary can learn from a leakage diagram. The ultimate goal of the adversary is to gain some information about the encoded secret. To achieve this it is enough that she learns the sum of all the x_i^j's from some row of the diagram. We now show how in case of leakage from Fig. 2.3 the adversary can compute $x_1^0 + x_2^0 + x_3^0$ from the values that belong to the $S(L)$ (i.e. those that are marked with double colored lines on Fig. (2.3b)). Using the facts that $x_i^{j+1} = x_i^j + b_i^j$ and $c_{i+1}^j = c_i^j + b_{i+1}^j$ several times we have:

$$x_1^0 + x_2^0 + x_3^0 = (x_1^1 - b_1^0) + (x_2^1 - b_2^0) + x_3^0 = x_1^1 + x_2^1 - (b_1^0 + b_2^0) + x_3^0 = \\ x_1^1 + (x_2^2 - b_2^1) - c_2^0 + x_3^0 = x_1^1 + x_2^2 + c_1^1 - c_2^1 - c_2^0 + x_3^0$$

where all the variables on the right hand side belong to $S(L)$. It is easy to see that the reason why the adversary is able to compute $x_1^0 + x_2^0 + x_3^0$ is that the leftmost and the rightmost nodes in the row containing edges labeled with

variables were connected. These nodes are indicated with the "$\circledast$" symbol on Fig. (2.3b).

Since the leftmost and the rightmost columns always belong to the leakage diagram, thus in general a similar computation is possible when these two columns are connected. Our first key observation is that if these columns are *not* connected, then the secret x remains secure. We state this fact below in the form of a following informal lemma.

Informal Lemma 1. *Consider a multi-round refreshing circuit. Let L be the set of leaking wires. Let E denote the event that the leftmost and the rightmost columns of $S(L)$ are connected. If E did not occur then the adversary gains no information about the secret.*

This informal lemma is formalized as Claim 6 in the full proof of our central Theorem 1, where it is also stated in a more general form, covering the case of more complicated circuits (i.e. those that perform some operations in addition to refreshing).

The rest of this chapter is organized as follows. In Sect. 2.1 we outline the main ideas behind the proof on Informal Lemma 1, in Sect. 2.2 we sketch the proof of the upper bound on the probability of E. This, together with the Informal Lemma 1 shows the security of our multi-round refreshing construction. Then in Sect. 2.3 we describe how these ideas can be generalized to arbitrary circuits. Besides of presenting the intuitions behind our formal proof, the goal of this part is also to introduce some more terminology that is useful later (e.g.: the "modification vectors"). In the sequel we use the following convention: if G is a labeled graph such that the labels on its edges are unique, then we sometimes say "edge λ" as a shortcut for "edge *labeled* with λ". The same convention applies to circuits and wires.

2.1 Proof sketch of Informal Lemma 1

Here we present the main ideas behind the proof of Informal Lemma 1. Consider a k-round refreshing circuit $\widehat{C}$ that takes as input a secret shared over n wires. For two arbitrary field elements $x^0, x^1 \in \mathbb{F}$ consider experiments of applying $\widehat{C}$ to their random encodings. In the proof we consider a fixed set L of leaking wires in $\widehat{C}$. Assume that event E did not occur, i.e., the leftmost and the rightmost columns of the leakage diagram are disconnected.

To prove Informal Lemma 1, it is enough to show that for the distributions of the values of wires in L are identical in both experiments (following the standard approach in cryptography this formally captures the fact that the adversary "gains no information about the secret"). We do it using a hybrid argument. Namely, we consider a sequence of experiments denoted $\mathrm{Exp}_A^0, \mathrm{Exp}_B^0, \mathrm{Exp}_C, \mathrm{Exp}_B^1$, and Exp_A^1 (see below), such that: (a) Exp_A^ℓ (for $\ell = 0, 1$) is equal to the original experiment in which x^ℓ is refreshed k times, and (b) the view of the adversary is identical for each pair of consecutive experiments on this list (and hence it is identical for all of them).

Extending the notation from the pseudocode given in Fig. (2.1a), we will add for future reference to the procedure that refreshes a secret x^ℓ (with $\ell \in \{0,1\}$) a superscript "ℓ" to all the labels, i.e., denote $\vec{x}^{j,\ell} := (x_1^{j,\ell}, \ldots, x_n^{j,\ell}), \vec{b}^{j,\ell} := (b_1^{j,\ell}, \ldots, b_n^{j,\ell})$ and $\vec{c}^{j,\ell} := (c_1^{j,\ell}, \ldots, c_n^{j,\ell})$. Note that all the operations in the refreshing circuit are linear, and in terms of linear algebra this experiment (repeated k times) can be described as:

$\mathbf{Exp}_A^\ell$:

Sample $\vec{x}^{0,\ell} \leftarrow \mathrm{Enc}(x^\ell)$.
For $j = 0$ to $k-1$ do:
1. sample $\vec{b}^{j,\ell} \leftarrow \mathrm{Enc}(0)$,
2. let $\vec{c}^{j,\ell} := f(\vec{b}^{j,\ell})$,
3. let $\vec{x}^{j+1,\ell} := \vec{x}^{j,\ell} + \vec{b}^{j,\ell}$,

where f is a linear function defined as

$$f(\vec{b}^{j,\ell}) = \begin{array}{l} (b_1^{j,\ell}, \\ b_1^{j,\ell} + b_2^{j,\ell}, \\ \quad \vdots \\ b_1^{j,\ell} + \cdots + b_n^{j,\ell}). \end{array} \tag{2.2}$$

It is easy to see that the following experiment (where the $\vec{x}^{j,\ell}$'s are chosen *first*, and then $\vec{b}^{j,\ell}$ is computed as their difference) has the same distribution of the variables

$\mathbf{Exp}_B^\ell$:

Sample $\vec{x}^{0,\ell} \leftarrow \mathrm{Enc}(x^\ell)$.
For $j = 0$ to $k-1$ do:
1. sample $\vec{x}^{j+1,\ell} \leftarrow \mathrm{Enc}(x^\ell)$,
2. let $\vec{b}^{j,\ell} := \vec{x}^{j+1,\ell} - \vec{x}^{j,\ell}$,
3. let $\vec{c}^{j,\ell} := f(\vec{b}^{j,\ell})$.

Hence, we can think of the k-round refreshing of secret x^ℓ (for $\ell = 1, 2$) as an experiment of choosing $k+1$ random encodings $\vec{x}^{0,\ell}, \ldots, \vec{x}^{k,\ell}$ of x^ℓ and then computing the $\vec{b}^{j,\ell}$'s and $\vec{c}^{j,\ell}$'s (according to the rules from steps 2 and 3 of Exp_B^ℓ). Let $\vec{x}^{0,\ell}, \ldots, \vec{x}^{k,\ell}$ be the random encodings of x^ℓ. Recall that our goal is to show that the encodings of x^0 are indistinguishable from encodings of x^1 given the leakage from wires in the set L. Our approach to this is as follows. Based on the leakage diagram $S(L)$ (and *independently* from the choice of the $x_i^{j,\ell}$'s) we construct carefully crafted vectors $\vec{m}^0, \ldots, \vec{m}^k \in \{-1, 0, 1\}^n$ that we call *basic modification vectors* such that for every j we have that

$$m_1^j + \cdots + m_n^j = 1 \tag{2.3}$$

(where $(m_1^j, \ldots, m_n^j) = \vec{m}^j$). These vectors have to satisfy also some other conditions (that we define in a moment, see Eq. (2.8) and (2.9)). See Fig. 2.4 for an example. We then consider the following modification of Exp_B^ℓ:

Exp$_C$:

Sample $\vec{x}^{0,1} \leftarrow \mathrm{Enc}(x^0) + (x^1 - x^0) \cdot \vec{m}^0$.
For $j = 0$ to $k-1$ do:

1. sample $\vec{x}^{j+1,1} \leftarrow \mathrm{Enc}(x^0) + (x^1 - x^0) \cdot \vec{m}^{j+1}$,
2. let $\vec{b}^{j,1} := \vec{x}^{j+1,1} - \vec{x}^{j,1}$,
3. let $\vec{c}^{j,1} := f(\vec{b}^{j,1})$.

We now have the following simple observation.

Claim 3. *The joint distribution of the variables $\vec{x}^{j,1}, \vec{b}^{j,1}$, and $\vec{c}^{j,1}$ in Exp_C is the same as in Exp_B^1.*

Proof. Since the $\vec{b}^{j,1}$'s and $\vec{c}^{j,1}$'s are functions of the $\vec{x}^{j,1}$'s thus it is enough to consider only the $\vec{x}^{j,1}$'s. First observe that $\vec{x}^{j,1}$ in Exp_C indeed encode x^1. Indeed, we have

$$\begin{aligned}
\mathrm{Dec}(\vec{x}^{j,1}) &= \mathrm{Dec}(\mathrm{Enc}(x^0) + (x^1 - x^0) \cdot \vec{m}^j) \\
&= \mathrm{Dec}(\mathrm{Enc}(x^0)) + (x^1 - x^0) \cdot \mathrm{Dec}(\vec{m}^j) && (2.4) \\
&= x^0 + (x^1 - x^0) \cdot 1 && (2.5) \\
&= x^1,
\end{aligned}$$

where in (2.4) we used the linearity of the encoding scheme, and in (2.5) we used the property of the basic modification vectors from Eq. (2.3). To

see why every $\vec{x}^{j,1}$ is distributed uniformly over the set of all encodings of x^1 it is enough to observe that $\vec{x}^{j,1}$ is a result of adding a constant vector $(x^1 - x^0) \cdot \vec{m}^j$ to a vector that is chosen uniformly from the set of all encodings of x^0. □

Therefore what remains is to show the following.

Claim 4. *The view of the adversary (i.e. the values that leak) are distributed identically in* Exp_C *and in* Exp^0_B.

(Clearly, after showing this we will be done as Claims 3 and 4 will together imply that the leakages in Exp^0_B and Exp^1_B are distributed identically.)

We do not show a formal proof of this claim (for a formal proof in case of arbitrary circuits see the proof of Thm. 1 in Chapter 6), but only provide some intuitions behind it. The validity of Claim 4 depends, of course, on the details of the construction of the basic modification vectors (whose description we postponed until now). Informally, we need to construct them in such a way that the encodings

$$\vec{x}^{j,1} := \mathrm{Enc}(x^0) \tag{2.6}$$

and

$$\vec{x}^{j,1} := \mathrm{Enc}(x^0) + (x^1 - x^0) \cdot \vec{m}^j \tag{2.7}$$

are "indistinguishable" from the point of view of the adversary, i.e., the leaking edges $S(L)$ have the same values no matter if $\vec{x}^{j,1}$ was computed according to (2.6) or (2.7). To deal with leaking $x_i^{j,1}$'s we introduce the following requirement:

> if an edge $x_i^{j,1}$ belongs to $S(L)$, then the ith coordinate of $\vec{m}^j$ equals to 0. (2.8)

The reason for having this requirement is that without it (2.6) and (2.7) would obviously differ on the ith coordinate. To deal with leaking $c_i^{j,1}$'s we have the following:

> if the edge $c_i^{j,1}$ belongs to $S(L)$, then the sum of first i coordinates is the same for vectors $\vec{m}^j$ and $\vec{m}^{j+1}$. (2.9)

The reason for having this is similar to the one for (2.8). To see it let X^j and X^{j+1} be two independent and random encodings of x^0, let $\vec{c}^{j,1}_{(2.6)}$ be computed from (2.6) as in Exp^0_B, i.e.:

$$\vec{c}^{j,1}_{(2.6)} = f(X^{j+1} - X^j),$$

and let $\vec{c}^{j,1}_{(2.7)}$ denote the "$\vec{c}^{j,1}$ vector computed from (2.7) as in Exp_C", i.e.

$$\begin{aligned}\vec{c}^{j,1}_{(2.7)} &= f\left((X^{j+1} + (x^1 - x^0)\cdot\vec{m}^{j+1}) - (X^j + (x^1 - x^0)\cdot\vec{m}^j)\right) \\ &= f(X^{j+1} - X^j) + (x^1 - x^0)\cdot f(\vec{m}^{j+1} - \vec{m}^j), \qquad (2.10)\end{aligned}$$

where (2.10) follows from the linearity of f. Obviously $\vec{c}^{j,1}_{(2.6)}$ and $\vec{c}^{j,1}_{(2.7)}$ are not necessarily identical since vector

$$f(\vec{m}^{j+1} - \vec{m}^j) \qquad (2.11)$$

does not need to be equal to $(0, \ldots, 0)$. Fortunately what suffices for us is that $\vec{c}^{j,1}_{(2.6)}$ and $\vec{c}^{j,1}_{(2.7)}$ are equal to each other "from the point of view of the adversary". This can be expressed as: for every $\vec{c}^j_i$ that belongs to $S(L)$, the value of the ith coordinate of vector (2.11) has to be equal to 0. From the definition of f (see Eq. (2.2)) we know that the ith coordinate of $f(\vec{m}^{j+1} - \vec{m}^j)$ is equal the sum of first i coordinates of the vector $(\vec{m}^j - \vec{m}^{j+1})$, which has to be equal to 0 by the requirement from Eq. (2.9). Hence $\vec{c}^{j,1}_{(2.6)}$ and $\vec{c}^{j,1}_{(2.7)}$ are indeed indistinguishable from the point of view of the adversary. Note that by dealing with the c^j_i's we automatically dealt with leaking b^j_i's, because of the way $S(L)$ was constructed: for each leaking b^j_i's two c's were added.

What remains is to show how the basic modification vectors are constructed. Let LS be the connected component of $S(L)$ that contains its leftmost column. By assumption, E did not occur so LS does *not* contain the rightmost column of $S(L)$. This makes it possible to construct the basic modification vectors with desired properties. For each j construct $\vec{m}^j = (m^j_1, \ldots, m^j_n)$ according to the following rules: (i) if the left node of the edge "x^j_i" *does* belong to LS and its right node *does not* belong to LS, then let m^j_i be equal to $+1$, (ii) if the left node of the edge "x^j_i" *does not* belong to LS and its right node *does belong* to LS, then let m^j_i be equal to -1, and (iii) let all the other m^j_i's be equal to 0. An example of how the basic modification vectors are constructed is presented on Fig. 2.4 (these vectors and their coordinates are marked there with numbers in boxes). As it turns out (see Lemma 8 in the security proof for a generalization of this statement) these rules guarantee that the requirements from Eq. (2.3), (2.8), and (2.9) are satisfied.

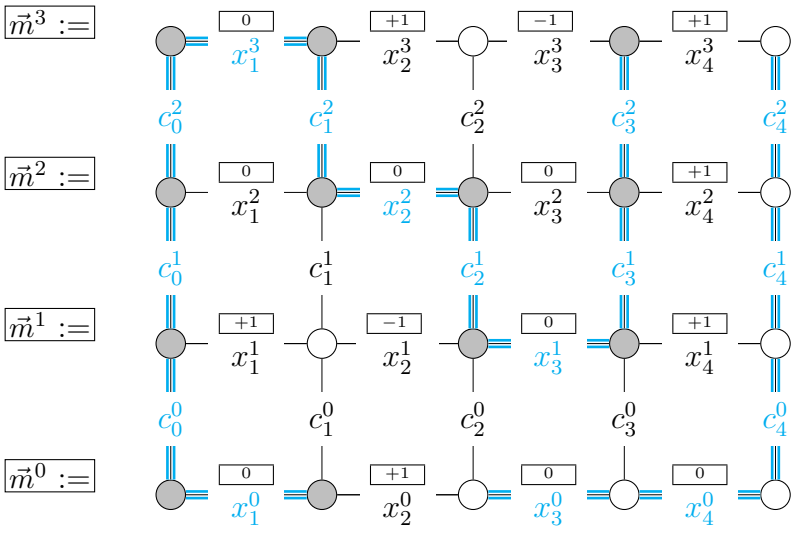


Figure 2.4: The example of the leakage diagram with leakage indicated with double colored lines. The nodes of the connected component LS (containing the leftmost column) are indicated with gray color. The modification vectors $\vec{m}^j$ and their coordinates are placed in boxes (e.g.: $\vec{m}^0 := (0, +1, 0, 0)$).

2.2 Bounding the probability of E

To show how we derive a bound on the probability of E we take a closer look at how, from the probabilistic point of view, the leakage diagram is constructed (see p. 25). By definition, it is a subgraph of a graph G from Fig. (2.2a). Recall that in our experiment every wire of the circuit $\widehat{C}$ leaks independently at random with probability p. The leakage diagram $S(L)$ corresponding to leakage L is a random subgraph of G.

Let us now analyze the distribution of $S(L)$. It is easy to see that every edge "x_i^j" is added to $S(L)$ independently with probability p. Unfortunately, the situation is slightly more complicated when it comes to the c_i^j's. Recall that c_i^j's can be added to $S(L)$ for three reasons. The first (trivial) reason is that $i = 0$ or $i = n$. The second reason is that the wire "c_i^j" leaks in $\widehat{C}$ (i.e.: it belongs to L). The third reason is that the wire b_i^j or b_{i+1}^j leaks in $\widehat{C}$. Because of this, the events $\{$"c_i^j belongs to $S(L)$"$\}_{i,j}$ are *not* independent, and the probability of each of them may *not* equal to p.[2]

[2]For example: it is easy to see that if we know that $c_i^j \in S(L)$ then the event "c_{i+1}^j

Let us look at the "non-trivial" edges in $S(L)$, i.e., the x_i^j's and the c_i^j's such that $i \in \{1, \ldots, n-1\}$. Let $\mathcal{U}$ be the variable equal to the set of non-trivial edges in $S(L)$. To make the analysis of the leakage diagram simpler it will be very useful to eliminate the dependencies between the "$c_i^j \in \mathcal{U}$" events. We do it by defining another random variable $\mathcal{Q}$ (that takes the same values as $\mathcal{U}$), and that has the following properties.

1. It is "more generous to the adversary", i.e., for every set $\mathcal{C}$ of the edges we have that
$$\Pr[\mathcal{C} \subset \mathcal{Q}] \geq \Pr[\mathcal{C} \subset \mathcal{U}] \tag{2.12}$$
(we will also say that the distribution of Q *covers the distribution of* $\mathcal{U}$, see Def. 1 on p. 37), and

2. The events $\{v \in \mathcal{Q}\}$ (where v is a non-trivial edge) are independent and have equal probability. Denote this probability q, and say that $\mathcal{Q}$ has a *standard distribution* (see Def. 2 on p. 37).

Now, consider an experiment $\mathrm{Exp}_{\mathcal{Q}}$ of constructing a leakage diagram when the "$c_{i,j}$" and "$x_{i,j}$" edges are chosen according to $\mathcal{Q}$. More precisely: let the edges in the leakage diagram be sampled independently according to the following rules: the $\{c_0^j\}$'s and $\{c_n^j\}$'s are chosen with probability 1, and the remaining $\{c_i^j\}$'s are chosen with probability q. It is easy to see that, thanks to Eq. (2.12), the probability of E in $\mathrm{Exp}_{\mathcal{Q}}$ is at least as high as in the probability in the original experiment. Hence, to give a bound on the probability of E it suffices to bound the probability of this event in $\mathrm{Exp}_{\mathcal{Q}}$. Thanks to the independence of the events $\{c_i^j \in \mathcal{Q}\}_{i,j} \cup \{x_i^j \in \mathcal{Q}\}_{i,j}$ bounding the probability of E in $\mathrm{Exp}_{\mathcal{Q}}$ becomes a straightforward probability-theoretic exercise. For the details on how it is done see the proof of the Thm. 1 in Chapter 6.

2.3 Generalizations to arbitrary circuits

As mentioned in Sect. 1.3, our final main contribution is a circuit compiler that uses the simple refreshing together with gadgets that perform the field operations. We follow the standard method of constructing compilers in a "gate-by-gate" fashion (see, e.g., [25], and the follow up work). A compiler

belongs to $S(L)$" becomes more likely (because leakage of b_{i+1}^j is more likely).

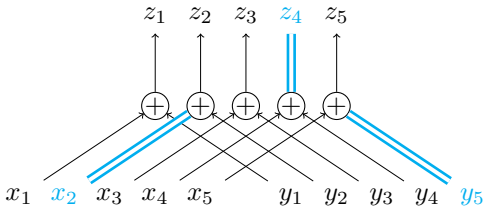
takes as input a circuit C (for simplicity assume it has no randomness gates) and produces as output a transformed circuit $\widehat{C}$ (that contains randomness gates *RND*). More concretely a wire carrying x in C gets transformed into a *bundle* of n wires carrying a random encoding of x. Every gate Γ in C is transformed into a "masked gate" $\widehat{\Gamma}$. For example, an addition gadget will have $2n$ inputs for n-share encodings of two values a and b, and n output wires that will carry some encoding of $a+b$. The masked input gates simply encode the secret (they have one input and n outputs). The masked output gates decode the secret (they have n inputs and one outputs). These two gadgets are assumed to be leak-free. They are also called: *input encoder* $\widehat{I}$ and *output decoder* $\widehat{O}$, respectively. For technical reasons, in our construction we insert the refreshing gadgets between the connected gadgets.

The main challenge in extending our ideas to such general circuits is that we need to take into account the leakage from wires of the individual gadgets, and represent them in the leakage diagram. We do it in such a way that unless an event E occurs, we are guaranteed that the adversary gained no information about the secret input. By the "event E" we mean a generalization of the event E (from the previous sections) to more complicated leakage diagrams. More concretely (see Sect. 5.2 for details) our approach is to represent each gadget $\widehat{\Gamma}$ in the graph G with a path $N_0^{\widehat{\Gamma}} - \cdots - N_n^{\widehat{\Gamma}}$ of length n and to "project" the leaking wires of the given gadget onto the edges of the path. Technically, this is done be defining, for every gadget $\widehat{\Gamma}$, a leakage *projection function* (see Sect. 4.2) that describes how a leakage from an internal wire is mapped on the path.

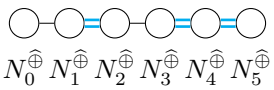
A projection function P, by definition, takes as argument a leaking wire w in a gadget $\widehat{\Gamma}$, and returns a subset of $[n]$ (usually of size 1 except for some wires in the multiplication gadget). We can refer to a projection of a set of wires in $\widehat{\Gamma}$ defined in a natural way as $P(\{w_1, \ldots, w_l\}) := P(w_1) \cup \cdots \cup P(w_l)$. One of the requirements that we impose on the function P is the following: every set of probes $\{w_1, \ldots, w_l\}$ (regardless of its size) from $\widehat{\Gamma}$ can be simulated knowing only input shares of indices in the projection $P(\{w_1, \ldots, w_l\})$ within each input bundle. Notice that it makes our definition of the gadget security similar in spirit to the existing definitions for the t-probing leakage model, like d-non-interference. One of the differences is that we care not only about the number of input shares that suffice to simulate the leakage, but also take into account their indices in a particular input bundle. Having a leakage projection function P defined for a gadget $\widehat{\Gamma}$, we

will represent a leakage from that gadget in the leakage diagram as a subset of the edges from the path in G: $N_0^{\widehat{\Gamma}} - \cdots - N_n^{\widehat{\Gamma}}$. The positions (with the edge $N_0^{\widehat{\Gamma}} - N_1^{\widehat{\Gamma}}$ being the 1st one) of these edges in the path are taken from the set $P(\{w_1, \ldots, w_l\})$, when the wires $w_1, \ldots, w_l$ are leaking. This way we can "project" any given leakage from a gadget onto the path of length n in the leakage diagram.

As an example consider the addition gadget "$\widehat{\oplus}$" that computes an encoding $\vec{z}$ of $z = x + y$ as $\vec{z} := \vec{x} + \vec{y}$ (where $\vec{x}$ and $\vec{y}$ are encodings of x and y, respectively). The leakage projection function $P_{\widehat{\oplus}}$ for this gadget is defined as follows. Each input wire that is on ith position in the input bundle is projected onto the set $\{i\}$, i.e., $P_{\widehat{\oplus}}(x_i) = \{i\}$ and $P_{\widehat{\oplus}}(y_i) = \{i\}$. Moreover, projection of the output wires is defined similarly, namely $P_{\widehat{\oplus}}(z_i) = \{i\}$. It is easy to see that with such projection function the above mentioned simulation requirement is satisfied. For example, the leakage illustrated on Fig. (2.5a) can be simulated knowing 3 input shares from each input bundle, namely x_2, x_4, x_5 and y_2, y_4, y_5. On the leakage diagram we represent this particular leakage from the addition gadget with 3 edges, as illustrated on Fig. (2.5b). Note that the addition gate is simple, and hence the projection function for it is rather straightforward. The projection function for a multiplication gadget is more involved (see Sect. 4.3).



(a) An example of leakage from the addition gadget "$\widehat{\oplus}$" (marked with double colored lines).



(b) The corresponding "projected" leakage in the leakage diagram (marked with double colored lines).

Figure 2.5: Leakage from an addition gadget and the corresponding "projected" leakage. This is a valid projection, since it is enough to know x_2, x_4, x_5 and y_2, y_4, y_5 to simulate the leakage.

Having the projections of leakages for individual gadgets defined, we can generalize the idea of a leakage diagram $S(L)$ presented in previous sections

from simple sequential k-round refreshing circuits to arbitrary private circuits built according to our construction. Recall that we insert a refreshing gadget between each pair of connected gadgets. The leakage from each individual gadget is projected onto a respective path in the leakage diagram, and the leakage from the remaining wires, i.e., wires used to generate encodings Enc(0) between two gadgets is "projected" onto the edges connecting the respective paths (analogue of the edges c_i^j's from previous sections). See Sect. 5.2 for the details. Overall, we obtain a graph that is similar to the leakage diagrams from the previous sections, but it is more general. In case of an example depicted on Fig. 2.6 the leakage from the gadget $\widehat{\Gamma}_1$ induces a projection set $\{3\}$. This fact is represented by including the edge $N_2^{\widehat{\Gamma}_1} - N_3^{\widehat{\Gamma}_1}$ into the leakage diagram.

A crucial property of such leakage diagrams is that the generalization of the Informal Lemma 1 still holds: the notion of the leftmost and the rightmost column are generalized to the leftmost and the rightmost *sides* (respectively). On Fig. 2.6 the leftmost side is a graph consisting of nodes $N_0^{\widehat{\Gamma}_1}, N_0^{\widehat{\Gamma}_2}, N_0^{\widehat{\Gamma}_3}, N_0^{\widehat{\Gamma}_4}$, and $N_0^{\widehat{\Gamma}_5}$, while the rightmost one consists of nodes $N_3^{\widehat{\Gamma}_1}, N_3^{\widehat{\Gamma}_2}, N_3^{\widehat{\Gamma}_3}, N_3^{\widehat{\Gamma}_4}$, and $N_3^{\widehat{\Gamma}_5}$. We now define the event E as: "the leftmost and the rightmost sides are connected". For example E does *not* hold for the diagram on Fig. 2.6. To make it easier to verify this fact, we indicate (with gray color) the nodes connected with the leftmost side.

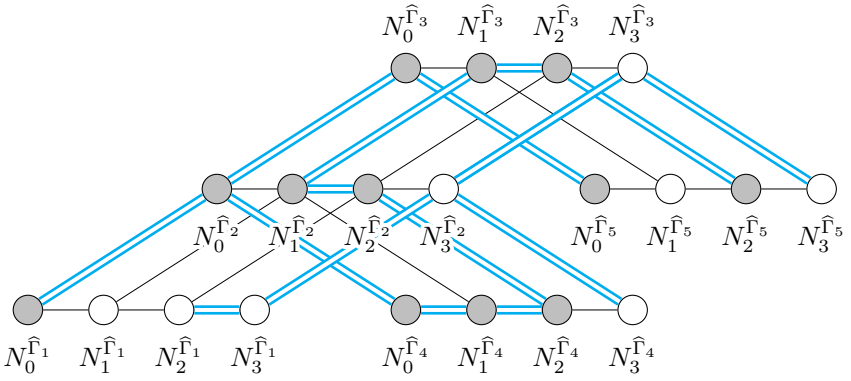


Figure 2.6: An example of a leakage diagram for a transformed circuit $\widehat{C}$ with 5 gadgets. The nodes connected with the leftmost side are marked in gray.

When using the leakage projection functions we encounter the following

problem that is similar to the "lack of independency problem" described in Sect. 2.2. Namely, it may happen that the events different edges become part of the projected set are *not* independent (this is, e.g., the case for the multiplication gadget in Sect. 4.3). We handle this problem in a similar way as before (see points 1 and 2 on page 32). That is: we define a "more generous" *leakage projection distribution* that (1) "covers" the original distribution, and (2) is "standard" (see the aforementioned points for the definition). Let q be the parameter denoting the probability in the standard distribution. This parameter, of course, depends on the probability p with which a wire leaks. A function that describes this dependence is called *projection probability function*. Every gadget in our construction comes with such a function. See Sect. 4.2 for a formalization of these notions.

Our construction is modular and works for different implementations of the addition and multiplication gadgets, assuming that they come with the leakage projection that satisfies certain conditions (see Thm. 1 on p. 71). We show (see Sect. 4.3) that the standard gadgets from the literature (including the ISW multiplication gadget [25]) satisfy this condition. Note that the construction and reasoning regarding the refreshing circuit presented in previous sections are special case of the construction and the security proof for the general arithmetic circuit. Indeed, we can treat each bundle between refreshing gadgets as an "identity gadget" (see Sect. 4.3).

Chapter 3

Preliminaries

In this chapter we present some basic notions and definitions used in the sequel chapters. Moreover, we recall the security reduction of [13] from the noisy leakage model to the random probing model.

Let us start with introducing some standard notation. In the sequel $[n]$ denotes the set $\{1, 2, \ldots, n\}$. We write $x \leftarrow \mathcal{X}$ when the element x is chosen uniformly at random from the finite set $\mathcal{X}$. For a random variable X we denote with $\mathcal{D}(X)$ its distribution.

3.1 Partial order of the distributions over the subsets

In this section we provide a formal definition of what it means that one probability distribution "covers" another one. The motivation and the intuition behind this concept were described in Sect. 2.2.

Definition 1. *Consider a fixed finite set A and its power set $P(A)$. Let D_1 and D_2 be some probability distributions over $P(A)$. We will say that distribution D_2* covers *distribution D_1 if there exist a finite sequence $D^0, D^1, \ldots, D^m$ of distributions over $P(A)$ such that $D^0 = D_1$, $D^m = D_2$ and for each $i = 1, 2, \ldots, m$ the distribution D^i is obtained by modifying distribution D^{i-1} in the following way:*

1. *Pick two subsets* S_1, S_2 *satisfying* $S_1 \subset S_2 \subset A$.

2. *Pick a real value* d *satisfying* $0 \leq d \leq D^{i-1}(S_1)$.

3. *Define the distribution* D^i *as follows:*

   - $D^i(S) := D^{i-1}(S)$ *for each subset* $S \subset A$ *except for* S_1 *and* S_2
   - $D^i(S_1) := D^{i-1}(S_1) - d$
   - $D^i(S_2) := D^{i-1}(S_2) + d$

Remark 1. *It is clear from the definition above that the relation of covering is a partial order of the probability distributions. We will write* $D_1 \geq D_2$ *to denote the coverage relationship of the distributions* D_1 *and* D_2 *(when it is clear from the context over which power set these distributions are).*

As already mentioned in Sect. 2.2, one specific distribution over a power set $P(A)$ that we will consider is *a standard distribution* $D_p(A)$ where $0 < p < 1$.

Definition 2. *Let* A *be a finite set and let* $0 < p < 1$. *We define a random subset* S *of* A *as follows: any element of* A *belongs to* S *with probability* p*, independently. We call the distribution over the power set* $P(A)$ *determined by the random subset* S a standard distribution $D_p(A)$.

For a random variable X with a domain $P(A)$ we denote by $\mathcal{D}(X)$ the probability distribution over $P(A)$ generated by that variable. For two random variables X, Y with the same domain $P(A)$ we will say that Y covers X if $\mathcal{D}(Y)$ covers $\mathcal{D}(X)$.

3.2 Assumptions about the circuit

Throughout the rest of the dissertation we assume that the original circuit C that is a subject to our transformation is a standard arithmetic circuit over field $\mathbb{F}$, i.e., its gates are

- addition gate $\oplus$,
- multiplication gate $\otimes$,

- negation gate *NEG* implementing the additive inverse operation in $\mathbb{F}$, i.e., for input x it outputs $-x$
- copy gate *CP* copying its input, which for input x it has two outputs (x, x),
- constant gate $Const_\alpha$ that has no input and outputs constant α.

Notice that we assume that the original circuit C is deterministic, i.e. it has no random gates. This can be done without loss of generality, as the randomness can be provided to C as an additional input. The concrete gadgets replacing the gates in our construction are listed in Sect. 4.3. Moreover, we assume that the gadgets in the transformed circuit $\widehat{C}$ may use the random gate *RND*, which outputs a field element x chosen uniformly at random, i.e., $x \leftarrow \mathbb{F}$.

In the sequel we always assume a fixed security parameter n, i.e. every wire in the original circuit C will be replaced by a bundle of n wires in the transformed circuit $\widehat{C}$.

We introduce also a special input encoding gate I and a special output decoding gate O. They simply implement the identity function, but are replaced during our transformation with special non-leaking input encoder $\widehat{I}$ and output decoder $\widehat{O}$, introduced already in Sect. 2.3. The input encoder $\widehat{I}$ simply takes one input $x \in \mathbb{F}$, and outputs an encoding of x chosen uniformly at random, i.e., outputs n shares $(x_1, \ldots, x_n) \in \mathbb{F}^n$ chosen uniformly at random subject to a condition $\Sigma_{i=1}^n x_i = x$. Similarly, the output decoder takes as input n shares $(x_1, \ldots, x_n) \in \mathbb{F}^n$ and simply outputs $x = \Sigma_{i=1}^n x_i$.

We say that a circuit C is *affine* if it has only affine gates, i.e., does not use multiplication gates. We denote with $|C|$ the size of circuit C, i.e., the number of gates in C.

3.3 Security definitions

In this section we present the formal definitions of soundness and privacy of a circuit transformation. First, let us recall a standard definition of a *statistical distance*.

Definition 3. *A* statistical distance *between two random variables X_0 and X_1 (distributed over some set $\mathcal{X}$) is defined as*

$$\Delta(X_0; X_1) := 1/2 \cdot \sum_{x \in \mathcal{X}} |\Pr[X_0 = x] - \Pr[X_1 = x]|.$$

If $\Delta(X_0; X_1) \leq \epsilon$ *then we say that* X_0 and X_1 are ϵ-close.

Soundness of the circuit transformation is defined as follows.

Definition 4. *We say that transformation* $\widehat{C}$ *of* k*-input circuit* C *is* sound *if it preserves the functionality of* C*, that is*

$$\widehat{C}(\vec{x}) = C(\vec{x})$$

for every input $\vec{x}$ *of length* k*.*

Our privacy definition of a transformed circuit is essentially the same as the definition of statistical privacy given by Ishai, Sahai and Wagner in [25]. To reason about privacy of the circuit we consider the following experiment.

Definition 5. *For a fixed circuit* C *with* k *input wires, its input* $\vec{x} = (x_1, \ldots, x_k)$ *and probability* p *we define an experiment* $Leak(C, \vec{x}, p)$ *that outputs an adversarial view as follows:*

1. *Transformed circuit* $\widehat{C}$ *is fed with* $(x_1, \ldots, x_k)$ *resulting with some assignment of the wires in* $\widehat{C}$*.*
2. *Each wire in* $\widehat{C}$ *leaks independently with probability* p*.*

   Note that the input and output wires do not leak, as they are part of non-leaking $\widehat{I}$ *and* $\widehat{O}$ *gadgets.*
3. **Output:** *(*LW*: set of leaking wires in* $\widehat{C}$*,* A*: values assigned to the leaking wires in* LW *during the circuit evaluation).*

We are now ready to define the privacy of a circuit transformation.

Definition 6. *We say that transformation* $\widehat{C}$ *of circuit* C *is* (p, ϵ)*-private if leakage in experiment* $Leak(C, \vec{x}, p)$ *can be simulated up to* ϵ *statistical distance, for any input* $\vec{x}$*. More precisely, there exist a simulation algorithm that, not knowing input* $\vec{x}$*, outputs a random variable that is* ϵ*-close to the actual output of* $Leak(C, \vec{x}, p)$*.*

3.4 p-random probing model to noisy leakage model

As mentioned in Sect. 1.2.3, the noisy leakage model can be reduced to the p-random probing model. In this section we recall the definition of the noisy leakage model, originally proposed by Prouff and Rivain in [34], and its reduction to the p-random probing model proved by Duc et al. in [13]. The definitions given here are adapted to our needs, as we do not consider stateful circuits.

First, we give the definition of a noisy function.

Definition 7. *Let $Noise : \mathcal{X} \to \mathcal{Y}$ be a randomized function. We say that $Noise$ is δ-noisy if*

$$\delta = \Delta(X; (X|Noise(X)))$$

for X being chosen from $\mathcal{X}$ uniformly at random, i.e. $X \leftarrow \mathcal{X}$.

In the noisy leakage model the adversary specifies a noisy function for each of the wire in the circuit. The adversarial view consists of noisy leakage from all the wires, assuming the noises are mutually independent. More precisely, the adversary $\mathcal{A}$ plays and receives the output of the following game:

Definition 8. *For a fixed circuit C with k input wires, its input $\vec{x} = (x_1, \ldots, x_k)$ and noise δ we define a game $NoisyLeak(C, \vec{x}, \delta)$ that outputs an adversarial view as follows:*

1. *Adversary $\mathcal{A}$ specifies for each wire w_i in the transformed circuit $\widehat{C}$ a δ_i-noisy function $Noise_i$, where $\delta_i \leq \delta$.*
2. *$\widehat{C}$ is fed with $(x_1, \ldots, x_k)$ resulting with some assignment of the wires in $\widehat{C}$.*
3. *For each wire w_i in $\widehat{C}$ that is assigned with value v_i, the noisy value $Noise_i(v_i)$ is added to the adversarial view, where the noises is mutually independent. Note that the input and output wires do not leak, as they are part of non-leaking $\widehat{I}$ and $\widehat{O}$ gadgets.*
4. **Output:** *All the noisy values $Noise_i(v_i)$.*

Thanks to the Lemma 3 in [13], we can reduce the security in the noisy leakage model to the security in the random probing model. Here we state this lemma in a slightly adapted version:

Lemma 1. *Let $\mathcal{A}$ be an adversary in the game NoisyLeak$(C, \vec{x}, \delta)$. His view can be simulated perfectly, given the output of the experiment Leak$(C, \vec{x}, \delta \cdot |\mathbb{F}|)$.*

Therefore, the adversarial view in the noisy leakage model can be simulated perfectly, given the leakage in the p-random probing model. Thus in the rest of the dissertation we focus solely on proving the privacy of our construction in the random probing model.

Chapter 4

Details of the circuit transformation

In this chapter we present the technical details of our construction of the transformed circuit $\widehat{C}$, including a general description of a gadget and concrete gadgets we use.

4.1 Our construction of the transformed circuit $\widehat{C}$

We assume that the arithmetic circuit C that is subject to our transformation is as described in Sect. 3.2. For syntactic purposes we introduce also a special single-input single-output refreshing gate R that simply implements an identity function, similarly to I and O gates (see Sect. 3.2), but can be placed anywhere in the circuit C.

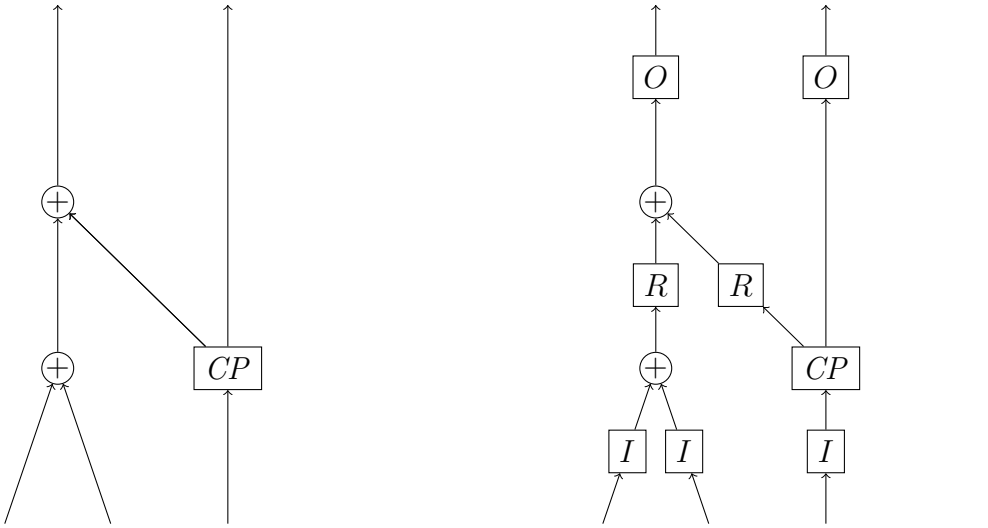
The transformation of the original circuit C consists of two phases:

1. *Preprocessing phase.* In this phase we add I gate to every input wire and O to every output wire in C. Moreover, we add refreshing gate R on *every* wire of C that connects any two gates, except for I and O (see Fig. 4.1). We call the resulting circuit C'.

2. *Actual transformation phase.* During this phase each wire in C', except its input and output wires, is replaced with a bundle of n wires (carrying an encoding). Each gate Γ in C' is replaced with a respective

gadget subcircuit $\widehat{\Gamma}$ that operates on the encodings. In particular, each refreshing gate R is replaced with a refreshing gadget $\widehat{R}$.

We give a detailed description of the gadget subcircuits in Sect. 4.3.



(a) An original circuit C

(b) Circuit C' after the preprocessing phase

Figure 4.1: Example of the preprocessing phase of the transformation.

We say that two gadgets, other than refreshing gadgets, $\widehat{\Gamma}_1$ and $\widehat{\Gamma}_2$ in $\widehat{C}$ are *connected* if there is a refreshing gadget between them. More precisely, if there is a refreshing gadget $\widehat{R}$ that takes as input the output bundle of $\widehat{\Gamma}_1$, and outputs the input bundle to $\widehat{\Gamma}_2$.

4.2 General gadget description

In this section we give a general definition of a gadget that can be used in our construction of a transformed circuit, and its required properties. Every gadget used in the construction, except for the special refreshing gadget $\widehat{R}$, satisfies the given definition (see Sect. 4.3). To distinguish them from the refreshing gadget, we call them *regular gadgets*.

Input and output wires of the gadget. Let us consider a gate Γ in the original circuit C with $0 \leq i \leq 2$ inputs and $1 \leq o \leq 2$ outputs, excluding the case $(i, o) = (2, 2)$. For example Γ might be a sum gate $\oplus$ or a multiplication gate $\otimes$. A respective gadget $\widehat{\Gamma}$ will have i input wire bundles and o output wire bundles, each bundle consisting of n wires, that is $i \cdot n$ inputs and $o \cdot n$ outputs in total.

We will denote with $IN_k^b(\widehat{\Gamma})$ the k-th wire of its b-th input bundle and with $OUT_k^b(\widehat{\Gamma})$ the k wire of its b-th output bundle. We denote with $IN_k(\widehat{\Gamma})$ all the input wires of index k in its input bundle. More precisely,

$$IN_k(\widehat{\Gamma}) := \{IN_k^b(\widehat{\Gamma}) | 1 \leq b \leq i\}.$$

Similarly, we define $OUT_k(\widehat{\Gamma})$ as

$$OUT_k(\widehat{\Gamma}) := \{OUT_k^b(\widehat{\Gamma}) | 1 \leq b \leq o\}.$$

Moreover, we use $IN^b(\widehat{\Gamma})$ to denote the b-th input bundle of $\widehat{\Gamma}$ and $OUT^b(\widehat{\Gamma})$ to denote its b-th output bundle. That is,

$$IN^b(\widehat{\Gamma}) := \{IN_k^b(\widehat{\Gamma}) | 1 \leq k \leq n\},$$

and

$$OUT^b(\widehat{\Gamma}) = \{OUT_k^b(\widehat{\Gamma}) | 1 \leq k \leq n\}.$$

Gadget correctness. Let $g : \mathbb{F}^i \to \mathbb{F}^o$ be the function computed by the gate Γ. The gadget $\widehat{\Gamma}$ should *implement* the same functionality as Γ. More precisely, if $g(x_1, \ldots, x_i) = (y_1, \ldots, y_o)$ then for *any* encoding $(\vec{x_1}, \ldots, \vec{x_i})$ of $(x_1, \ldots, x_i)$ fed to circuit $\widehat{\Gamma}$ as input, it outputs *some* encoding $(\vec{y_1}, \ldots, \vec{y_o})$ of $(y_1, \ldots, y_o)$.

Leakage projections. We now define the "leakage projections" already informally discussed in Sect. 2.3. Every gadget $\widehat{\Gamma}$ comes with *a leakage projection function* P that takes as argument a leaking wire w in $\widehat{\Gamma}$ and returns an associated subset $P(w)$ of $[n]$ (usually an one-element subset, except for the ISW multiplication gadget). We extend the domain of the function P, in a natural way, to the subsets of wires in $\widehat{\Gamma}$. For a subset of wires W we define it as

$$P(W) := \bigcup_{w \in W} P(w).$$

We require two following properties of the projection P to be satisfied:

- For any subset LG of leaking wires in $\widehat{\Gamma}$, it is enough to know the values carried by wires of the indices in the set $P(LG)$ from every input bundle, i.e., the wires in $\{IN_k^b(\widehat{\Gamma}) | 1 \leq b \leq i, k \in P(LG)\}$, to simulate the leakage from $\widehat{\Gamma}$ perfectly (without knowing the values of the other input wires).

- For every output wire w in $\widehat{\Gamma}$ that is k-th wire in any output bundle, i.e. $w \in OUT_k(\widehat{\Gamma})$, we have $P(w) = \{k\}$.

Leakage projection distribution. As described in Sect. 2.3, the leakage from the wires in a gadget used in our construction is "projected" onto a path in the diagram corresponding to the gadget (see Sect. 5.2 for details). In order to reason about the properties of the resulting leakage diagram, treated as a random variable, we introduce the notion of *leakage projection distribution*:

Definition 9. *Consider an experiment where each wire in the gadget $\widehat{\Gamma}$ leaks independently with probability p. Let LG denote the random set of the leaking wires. Induced projection of the leakage $P(LG)$, treated as a random variable, defines a probability distribution over the subsets of $[n]$. We call it* a leakage projection distribution *and denote it with $D_p(\widehat{\Gamma})$.*

Projection probability function. As described in Sect. 2.3, when using a leakage diagram representing the leakage in our construction of the private circuit, we encounter a "lack of independency problem": for a particular gadget $\widehat{\Gamma}$ (e.g., see the case of multiplication gadget in Sect. 4.3.1) the events of the edges representing the leakage in $\widehat{\Gamma}$ being included into the diagram, are *not* independent. To tackle this problem, we introduce a notion of *projection probability function* describing a particular gadget. Essentially, it expresses with what probability do we need to include each of the numbers from the set $[n]$ *independently*, to cover (as in Def. 1) the projection of the leakage from $\widehat{\Gamma}$.

Definition 10. *We say that a function $f : [0, 1] \to \mathbb{R}$ is* a projection probability function *for a gadget $\widehat{\Gamma}$ if the leakage projection distribution $D_p(\widehat{\Gamma})$ is covered (as in Def. 1) by the standard distribution $D_{f(p)}([n])$ (see Def. 2).*

Note that the function f may depend on the security parameter n, like in the case of the ISW multiplication gadget (see Sect. 4.3.1).

4.3 The gadgets used in our construction

In this section we present all the gadgets used in our construction.

4.3.1 ISW multiplication gadget

As the multiplication gadget $\widehat{\otimes}$ in our construction we use the gadget proposed in [25]. Here we recall their scheme and prove that it satisfies the general gadget definition.

1. **Input:** 2 bundles $\vec{x} = (x_1, \dots x_n)$ and $\vec{y} = (y_1, \dots y_n)$
2. For $1 \leq i < j \leq n$ sample $z_{i,j} \leftarrow \mathbb{F}$
3. For $1 \leq i < j \leq n$ compute $z_{j,i} = (z_{i,j} \oplus x_i \otimes y_j) \oplus x_j \otimes y_i$
4. Compute the output encoding $(t_1, \dots t_n)$ as $t_i = x_i \otimes y_i \oplus \bigoplus_{j \neq i} z_{i,j}$
5. **Output:** a bundle $\vec{t} = (t_1, \dots, t_n)$

We define the projection function P for this gadget as follows: For every wire w of the form x_i, y_i, $x_i \otimes y_i$, $z_{i,j}$ (for any $j \neq i$) or a sum of values of the above form (with t_i as a special case), $P(w) = \{i\}$. For the remaining wires w, which are of the form $x_i \otimes y_j$ or $z_{i,j} \oplus x_i \otimes y_j$, we define $P(w) = \{i, j\}$.

The following two lemmas allow the use of the ISW multiplication gadget in our construction.

Lemma 2. *The ISW multiplication gadget with its projection function satisfies a general gadget description (given in Sect. 4.2) for the multiplication function $g(x, y) = x \cdot y$.*

Proof. For the correctness, first notice that for $i \neq j$ we have $z_{j,i} \oplus z_{i,j} = x_i \otimes y_j \oplus x_j \otimes y_i$. It is clear that the gadget output bundle encodes the value

$$\sum_{i=1}^{n} t_i = \sum_{i=1}^{n} (x_i \otimes y_i \oplus \bigoplus_{j \neq i} z_{i,j}) = \left(\sum_{i=1}^{n} x_i \otimes y_i \right) \oplus \bigoplus_{1 \leq i < j \leq n} z_{i,j} \oplus z_{j,i}$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i \otimes y_j) = \left(\sum_{i=1}^{n} x_i \right) \otimes \left(\sum_{i=1}^{n} y_i \right),$$

so indeed the gadget implements a multiplication function.

For the leakage projection properties of the gadget, it is clear that for the i-th wire w of the output bundle we have $P(w) = \{i\}$. Now let us assume that the gadget is fed with some input $(\vec{x}, \vec{y})$ and LG is a set of leaking wires. Let $I = P(LG)$. We will show how to simulate the joint leakage from wires in LG given the values $\{x_i, y_i | i \in I\}$. The procedure is as follows:

1. Assign values to the $z_{i,j}$ as follows:
   - If $i \notin I$ (regardless of j) then $z_{i,j}$ does not enter into the computation for any $w \in LG$ and can be left unassigned.
   - If $i \in I$, but $j \notin I$ then $z_{i,j} \leftarrow \mathbb{F}$ is assigned with a random independent value. Note that if $i < j$ this is what would have happened during an honest evaluation of the gadget. However, if $i > j$ then we are making use of the fact that, by construction of the gadget and the definition of the projection function, $z_{j,i}$ will never be used in the computation of values asigned to any wires $w \in LG$. Hence, we can treat $z_{i,j}$ as an uniformly random and independent value.
   - If both $i \in I$ and $j \in I$ then we have access to x_i, x_j, y_i, y_j. Thus, we compute $z_{i,j}$ and $z_{j,i}$ honestly, as it would happen during the evaluation of the gadget: assuming $i < j$ we assign $z_{i,j}$ a random independent value and compute $z_{j,i} = (z_{i,j} \oplus x_i \otimes y_j) \oplus x_j \otimes y_i$.
2. For every wire $w \in LG$ of the form $x_i, y_i, x_i \otimes y_i, z_{i,j}$ or a sum of values of the above form, we have $i \in I$ and hence we have access to x_i and y_i. Also, all the needed values of $z_{i,j}$ have already been assigned in a perfect simulation. Thus the value carried by the wire w also can be computed in a perfect simulation.
3. The remaining unassigned wires $w \in LG$ are of the form $x_i \otimes y_j$ or $z_{i,j} \oplus x_i \otimes y_j$. In this case both $i, j \in I$ and $z_{i,j}$ has been already assigned. Thus, the value of w can be simulated perfectly.

This perfect simulation shows that the ISW multiplication gadget indeed

satisfies the general gadget description. □

Lemma 3. *The function $f(p) = n(8p + \sqrt{3p})$ is a projection probability function for the ISW multiplication gadget.*

Proof. Notice that by construction of the gadget, for any $i \in [n]$ there is no more that $8n$ wires that are projected onto the set $\{i\}$. Moreover, for any pair $i, j \in [n]$ such that $i \neq j$ there is no more than 3 wires that are projected onto the set $\{i, j\}$. These observations allow us to estimate the projection probability function for the gadget.

Fix probability $p \in [0, 1]$. Because of the observations above, it is clear that the leakage projection distribution $D_p(\widehat{\otimes})$ is covered (as in Def. 1) by the distribution $\mathcal{D}(P_1)$ of a random subset P_1 of the set $[n]$, produced in the following experiment:

1. Start with an empty set P_1.
2. For each $i \in [n]$, add i to P_1 with probability $8np$.
3. For each pair $i \neq j \in [n]$, add both i, j to P_1 with probability $3p$.

This distribution in turn is covered by the distribution $\mathcal{D}(P_2)$ of a random variable P_2 produced in the following experiment:

1. Start with an empty set P_2.
2. For each $i \in [n]$, add i to P_1 with probability $8np$.
3. For each pair $i \neq j \in [n]$ do the following:
   - add i to P_2 with probability $\sqrt{3p}$
   - add j to P_2 with probability $\sqrt{3p}$

It is clear from the description of P_2 that its distribution is covered by a standard distribution (see Def. 2) $D_{n(8p+\sqrt{3p})}([n])$. Hence, we have the coverage $\mathcal{D}(P_2) \leq D_{n(8p+\sqrt{3p})}([n])$, and thus we obtain the coverage relation (see Remark 1):

$$D_p(\widehat{\otimes}) \leq \mathcal{D}(P_1) \leq \mathcal{D}(P_2) \leq D_{n(8p+\sqrt{3p})}([n]),$$

which shows that $f(p) = n(8p + \sqrt{3p})$ is indeed a projection probability function for the gadget. □

4.3.2 Other gadgets

We already described the addition gadget $\widehat{\oplus}$ that implements the function $g(x, y) = x + y$ in Sect. 2.3. In this section we give a formal definition of this gadget and the rest of the gadgets used in our construction.

Addition gadget $\widehat{\oplus}$ implementing the function $g(x, y) = x + y$:

1. **Input:** 2 bundles $\vec{x} = (x_1, \ldots x_n)$ and $\vec{y} = (y_1, \ldots y_n)$
2. Compute the output encoding $(z_1, \ldots z_n)$ as $z_i = x_i \oplus y_i$
3. **Output:** a bundle $(z_1, \ldots, z_n)$

Copy gadget $\widehat{CP}$ implementing the function $g(x) = (x, x)$:

1. **Input:** 1 bundle $\vec{x} = (x_1, \ldots x_n)$
2. Apply copy gate CP to $x_1, \ldots x_n$ to obtain $(y_i, z_i) = CP(x_i)$
3. **Output:** 2 bundles $\vec{y} = (y_1, \ldots, y_n)$ and $\vec{z} = (z_1, \ldots, z_n)$

Negation gadget $\widehat{NEG}$ implementing the function $g(x) = -x$:

1. **Input:** 1 bundle $\vec{x} = (x_1, \ldots x_n)$
2. Apply negation gate NEG to $x_1, \ldots x_n$ to obtain $y_i = \widehat{NEG}(x_i)$
3. **Output:** 1 bundles $\vec{y} = (y_1, \ldots, y_n)$

Constant gadget $\widehat{Const_\alpha}$ implementing the function $g() = \alpha$:

1. **Input:** takes no input bundles
2. **Output:** 1 bundle $(\alpha, 0, \ldots, 0)$

A special *identity gadget* $\widehat{ID}$ implements the identity function $g(x) = x$.

Identity gadget $\widehat{ID}$ implementing the function $g(x) = x$:

1. **Input:** 1 bundle $\vec{x} = (x_1, \dots x_n)$
2. **Output:** 1 bundle $\vec{x} = (x_1, \dots x_n)$

The projection function P. For each of these gadgets we define the projection function P as follows: for every wire w of the form x_i, y_i, z_i, we define $P(w) = \{i\}$.

Properties of gadgets other than ISW multiplication gadget. It is clear that all the gadgets described above correctly implement the desired functions.

Moreover, for each of these gadgets the function $f(p) = 3p$ is a projection probability function, because for each number $i \in [n]$ there are at most 3 wires that are projected onto i, and these events are *independent*. For the gadgets $\widehat{Const_\alpha}$ and $\widehat{ID}$ even smaller function $f(p) = p$ is a projection probability function, because in cases of these gadgets there is exactly 1 wire projected onto number i, for each $i \in [n]$.

Chapter 5

Technical tools

In this section we present the technical tools used in the proof of our central theorem given in Chapter 6.

5.1 Refreshing gadget properties

In this section we describe properties of the refreshing gadget $\widehat{R}$ (see Fig. (2.1a) on p. 21) that are crucial to the security of the construction, and are used in the privacy proof. We will denote by *refreshing bundle* $B_{\widehat{R}}$ the wires that are used to generate the fresh encoding Enc(0) in the refreshing gadget $\widehat{R}$, i.e., wires carrying $b_1^j, \ldots, b_n^j$ and $c_1^j, \ldots, c_{n-1}^j$ on Fig. 2.1.

In order to represent the leakage from the refreshing bundles in the transformed circuit on the leakage diagram we need to define the projection of the leakage.

Refreshing bundle leakage projection. Consider a refreshing bundle $B_{\widehat{R}}$. Suppose that LR is a set of leaking wires in $B_{\widehat{R}}$. We define a subset $P_R(LR)$ of the set $\{0, \ldots, n\}$ representing the leakage LR as follows:

- Start with the set $P_R = \{0, n\}$.
- For every wire of the form $c_k^j = b_1^j \oplus b_2^j \oplus \cdots \oplus b_k^j$ in LR, where $1 \leq k < n$, add k to P_R.
- For every wire of the form b_k^j in LR , where $1 < k \leq n$, add k and $k-1$ to P_R.

Remark 2. *One may think of the function $P_R(\cdot)$ as an analogue of the leakage projection function (introduced in Section 4.2) in case of a refreshing gadget. The difference is, however, that $P_R(\cdot)$ codomain size is $n+1$ instead of n, and that two elements, namely 0 and n, belong to $P_R(LR)$ "by default".*

Leakage projection coverage. For the purposes of the privacy proof we construct a random subset of the set $\{0,\ldots,n\}$ that covers (as in Def. 1) the projection of the refreshing bundle leakage.

Definition 11. *Let us define a random subset R_q of $\{0,\ldots,n\}$ as follows:*

- *R_q contains 0 and n with probability 1.*
- *For any other number $i \in \{0,\ldots,n\}$, R_q contains i with probability q, independently.*

Lemma 4. *Let LR be a random subset of the leaking wires of a refreshing bundle $B_{\widehat{R}}$ when each wire leaks independently with probability p. Then the distribution of the random subset $P_R(LR) \subset \{0,\ldots,n\}$ is covered (as in Def. 1) by the distribution of $R_{p+2\sqrt{3p}}$ (see Def. 11), i.e. $\mathcal{D}(P_R(LR)) \leq \mathcal{D}(R_{p+2\sqrt{3p}})$.*

Proof. By construction of the refreshing gadget and by definition of $P_R(LR)$, for any $i \in \{1,\ldots,n-1\}$ there is no more than 1 wire that is projected onto the set $\{i\}$. Moreover, for any $i \in \{0,\ldots,n-1\}$ there is no more than 3 wires that are projected onto the set $\{i, i+1\}$. These observations allow us to prove the Lemma.

By construction, the distribution $\mathcal{D}(P_R(LR))$ is covered by the distribution $\mathcal{D}(P_1)$ of a random subset P_1 produced in the following experiment:

1. Start with a set $P_1 = \{0, n\}$.
2. For each $i \in \{1,\ldots,n-1\}$, add i to P_1 with probability p.
3. For each $i \in \{0,\ldots,n-1\}$, add both $i, i+1$ to P_1 with probability $3p$.

This distribution in turn is covered by the distribution $\mathcal{D}(P_2)$ of a random subset P_2 produced in the following experiment:

1. Start with a set $P_2 = \{0, n\}$.
2. For each $i \in \{1, \ldots, n-1\}$, add i to P_2 with probability p.
3. For each $i \in \{0, \ldots, n-1\}$ do the following:
   - add i to P_2 with probability $\sqrt{3p}$
   - add $i+1$ to P_2 with probability $\sqrt{3p}$

Let A_i be the event of $i \in P_2$. It is clear that events $A_1, \ldots, A_{n-1}$ are independent and for all i

$$\Pr[A_i] \leq p + 2\sqrt{3p}.$$

Hence we have the coverage $\mathcal{D}(P_2) \leq \mathcal{D}(R_{p+2\sqrt{3p}})$, and we obtain a coverage relation (see Remark 1):

$$\mathcal{D}(P_R(LR)) \leq \mathcal{D}(P_1) \leq \mathcal{D}(P_2) \leq \mathcal{D}(R_{p+2\sqrt{3p}}),$$

what finishes the proof of the lemma. □

Another crucial property of the refreshing gadget is that when its input and output are fixed, then values carried by all its internal wires are *determined* by these fixed values. We will call this *unique* assignment $\mathit{RefRec}(\vec{x}, \vec{y})$.

Definition 12. *Suppose that a refreshing gadget $\widehat{R}$ takes as input a vector $\vec{x}$ and outputs a vector $\vec{y}$. Then, it is possible to reconstruct* in a unique way *the values assigned to all the internal wires of $\widehat{R}$, i.e. the wires of the refreshing bundle $B_{\widehat{R}}$, and this assignment is determined by $\vec{x}$ and $\vec{y}$. We call this unique assignment $\mathit{RefRec}(\vec{x}, \vec{y})$. Additionally, for a specified subset of leaking wires LR in the refreshing bundle $B_{\widehat{R}}$ we denote the reconstructed assignment of these wires with $\mathit{RefRec}^{LR}(\vec{x}, \vec{y})$.*

Proof. Indeed, it is clear that for $1 \leq k \leq n-1$ the values b_k (see Fig. 2.1) carried by wires in the refreshing bundle $B_{\widehat{R}}$ are determined by the equality $x_k + b_k = y_k$, and in turn all the values carried by wires in $B_{\widehat{R}}$ are determined. □

5.2 Leakage diagrams

The main technical concept of this work is a *leakage diagram* (already introduced informally in Sect. 2).

Consider a transformed circuit $\widehat{C}$, as described in Sect. 4.1. Suppose that LW is the set of leaking wires in $\widehat{C}$. The leakage diagram is a representation of the set LW. As explained in Chapter 2, in the security proof the leakage diagram is used to determine whether the leakage compromises the secret or not. This can be done because of its property that if the *leftmost* and *rightmost sides* of the leakage diagram are disconnected then the privacy of the secret is preserved. This observation is formalized as Claim 6 in the proof of the main theorem.

The leakage diagram is a subgraph of a graph $G(\widehat{C})$ associated with the transformed circuit $\widehat{C}$. The leakage diagram inherits all nodes from $G(\widehat{C})$ and some of its edges, depending on the set of leaking wires LW. The exact construction of graph $G(\widehat{C})$ and the leakage diagram are described in the following paragraphs.

Graph $G(\widehat{C})$ associated with the transformed circuit $\widehat{C}$ Let C be any circuit and $\widehat{C}$ its transformation as described in Sect. 4.1. We define an associated undirected graph $G(\widehat{C})$ as follows. For each regular gadget $\widehat{\Gamma}$ in $\widehat{C}$, $G(\widehat{C})$ contains *a crosswise path* of length n, where n is the security parameter of the construction. We denote the nodes in this path with $N_0^{\widehat{\Gamma}}, \ldots, N_n^{\widehat{\Gamma}}$.

Moreover, for every pair $\widehat{\Gamma}_1, \widehat{\Gamma}_2$ of connected gadgets in $\widehat{C}$, we add to the graph $G(\widehat{C})$ a *vertical matching* consisting of the following $n+1$ edges: $(N_0^{\widehat{\Gamma}_1}, N_0^{\widehat{\Gamma}_2}), \ldots, (N_n^{\widehat{\Gamma}_1}, N_n^{\widehat{\Gamma}_2})$.

Leftmost and rightmost sides of $G(\widehat{C})$. We call all the nodes of the form $N_0^{\widehat{\Gamma}}$, for some gadget $\widehat{\Gamma}$ in $\widehat{C}$, together with the edges between these nodes *a leftmost side* of $G(\widehat{C})$. Similarly, we define *a rightmost side* of $G(\widehat{C})$ as all the nodes of the form $N_n^{\widehat{\Gamma}}$ with all the edges between them.

Decomposition of $G(\widehat{C})$. The graph $G(\widehat{C})$ can be naturally decomposed into separate subsets of edges - its crosswise paths, for each of the gadgets, and vertical matchings, for each pair of connected gadgets. We will call it a *decomposition* of $G(\widehat{C})$.

Leakage diagram. In the experiments, during the computation on circuit $\widehat{C}$ some wires will leak the carried values. Let LW denote the set of all the leaking wires. We will be representing this set with *a leakage diagram* $S(LW)$ - a subgraph of $G(\widehat{C})$. The leakage diagram inherits all the nodes from $G(\widehat{C})$ and some of its edges, as in the following construction.

Each leaking wire $w \in LW$ that belongs to some regular gadget $\widehat{\Gamma}$ is *projected* onto the respective crosswise path in $G(\widehat{C})$. More precisely, if $P_{\widehat{\Gamma}}$ is leakage projection function for the gadget $\widehat{\Gamma}$ then we add to the leakage diagram S the edges in the crosswise path that are in the set

$$\{(N_{i-1}^{\widehat{\Gamma}}, N_i^{\widehat{\Gamma}}) | i \in P_{\widehat{\Gamma}}(w)\}.$$

By construction of the transformed circuit $\widehat{C}$, the rest of the leaking wires in the set LW are part of some refreshing bundle $B_{\widehat{R}}$, where the refreshing gadget $\widehat{R}$ connects some gadgets $\widehat{\Gamma}_1$ and $\widehat{\Gamma}_2$. Let LR be the set of leaking wires in this refreshing bundle, i.e. $LR = B_{\widehat{R}} \cap LW$. It is represented in the leakage diagram S by the subset of the vertical matching between two respective crosswise paths, namely the edges in the set

$$\{(N_i^{\widehat{\Gamma}_1}, N_i^{\widehat{\Gamma}_2}) | i \in P_R(LR)\},$$

where $P_R(LR)$ is the projection of the refreshing bundle leakage LR (defined in Sect. 5.1). An example of a leakage diagram is illustrated on Fig. 2.6.

5.3 Modification vectors

In the security proof for our construction, presented in Chapter 6, we use a sequence of hybrid experiments that produce exactly the same adversarial view. One of the hybrids requires to assign every gadget in $\widehat{C}$ with a *basic modification vector*. They were already informally introduced in Chapter 2. We also generalize this notion to a *modification vector*. Let us now present their formal definition.

Definition 13. *A* basic modification vector *is a vector* $\vec{m} = (m_1, \ldots, m_n)$ *of length* n *satisfying two conditions:*

- $m_i \in \{-1, 0, 1\}$ *for* $i = 1, \ldots, n$, *and*
- $\sum_{i=1}^{n} m_i = 1$.

We will say that a vector $\vec{w}$ is a modification vector *if it is of the form $\vec{w} = v \cdot \vec{m}$ for some scalar value $v \in \mathbb{F}$ and a basic modification vector $\vec{m}$.*

Moreover, we formalize the requirement imposed on the modification vectors for the hybrid argument in the privacy proof to go through, which was already presented informally as the requirement from Eq. (2.8) on p. 29. If a modification vector $\vec{m}$ and a set $A \subset [n]$ satisfy that condition then we say that they are *disjoint*.

Definition 14. *We say that a modification vector $\vec{m} = (m_1, \ldots, m_n)$ is* disjoint *with a set $A \subset [n]$ if $m_a = 0$ for all $a \in A$.*

Modification vectors indistinguishability. In the security proof, one of the steps of the hybrid argument is to add to the vectors of values assigned to the input bundles of the gadgets in the transformed circuit a certain modification vector. It needs to be done in a way that is *not noticeable* to the adversary, and it turns out that it can be done, but one of the conditions (see Def. 21 for the other condition) that we need to impose on the modification vectors is that they are "indistinguishable" under the set that is the projection of the leakage in the refreshing bundle (this projection is defined in Sect. 5.1) between the connected gadgets. Similar condition was imposed on the modification vectors in the informal part in the requirement 2.9. Here we give the definition of vectors indistinguishability.

Definition 15. *Let S be a subset of $\{0, \ldots, n\}$ and let $\vec{m}^1, \vec{m}^2$ be any modification vectors of length n. We will say that $\vec{m}^1$ and $\vec{m}^2$ are* indistinguishable *under the set S if for every $k \in S$ we have that $\sum_{i=1}^{k} m_i^1 = \sum_{i=1}^{k} m_i^2$.*

5.4 Leakage and extended leakage from a gadget

In this section we give a formal definitions of the leakage and *extended leakage* from a gadget, and also prove that every gadget satisfying the general definition from Sect. 4.2 satisfies also the *extended leakage shiftability* property.

One of the hybrid experiments in the proof of the main theorem given in Chapter 6 constructs the adversarial view by "shifting" first the values fed to the gadgets in the transformed circuit $\widehat{C}$, i.e., adding to them certain

carefully chosen modification vectors. This technique was already introduced in the informally in Sect. 2.1 where in $\mathbf{Exp}_C$ we sample $\vec{x}^{j+1,1} \leftarrow \mathrm{Enc}(x^0) + (x^1 - x^0) \cdot \vec{m}^{j+1}$, which is encoding of x^0 shifted by $(x^1 - x^0) \cdot \vec{m}^{j+1}$. In the hybrid experiment, for the proof to succeed, we need to control also the values carried by *all the output wires* of all the gadgets in $\widehat{C}$. It is necessary in order to reason about the *whole leakage* from the circuit $\widehat{C}$, including the refreshing bundles in between the gadgets. Thus, we introduce the notion of the extended leakage, which includes also the output wires evaluation. Moreover, we crucially rely on the introduced extended leakage shiftability property of the gadgets, that allow to shift the input vectors of a gadget without changing the adversarial view.

First, we define a leakage from a gadget. This notion is only used in the proof of the Lemma 5.

Definition 16. *Let $\widehat{\Gamma}$ be a gadget with i input bundles and o output bundles and let LG be a subset of its wires. We define a function $GadgetLeak^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})$ as the output of the following experiment:*

1. *The gadget $\widehat{\Gamma}$ is fed with input $(\vec{x_1}, \ldots, \vec{x_i})$ resulting with some assignment of the wires of $\widehat{\Gamma}$.*
2. **Output:** *values assigned to the wires in LG.*

Extended leakage. It is a leakage from a subset of wires in $\widehat{\Gamma}$ together with values carried by *all* the output wires of $\widehat{\Gamma}$, including the non-leaking wires.

Definition 17. *Let $\widehat{\Gamma}$ be a gadget with i input bundles and o output bundles and let LG be a subset of its wires. We define a function $ExtLeak^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})$ as the output of the following experiment:*

1. *The gadget $\widehat{\Gamma}$ is fed with input $(\vec{x_1}, \ldots, \vec{x_i})$ resulting with some assignment of the wires of $\widehat{\Gamma}$.*
2. *Let $\vec{y_1}, \ldots, \vec{y_o}$ be the produced output of $\widehat{\Gamma}$.*
3. **Output:** *(values assigned to wires in LG, values assigned to* all *the output wires $\vec{y_1}, \ldots, \vec{y_o}$).*

Extended leakage shiftability. Recall that in Sect. 2.1 one of the main technical tricks was to show that the experiments Exp_C and Exp_B^0 are indistinguishable from the point of view of the adversary. This was done by showing that the vectors encoding the secret can be "shifted" (i.e. a certain vector can be added to it) in way that is not noticeable to the adversary. This idea is formalized and generalized to gadgets below.

First, we define the *shift* function:

Definition 18. *Let $\vec{v_1}, \ldots \vec{v_k}$ and $\vec{m}$ be vectors of the same length. and let $T = (T_1, \ldots, T_k)$ be a sequence of k field elements. We define a $shift_{\vec{m}}^T(\vec{v_1}, \ldots, \vec{v_k})$ as follows: it is a sequence of vectors $\vec{w_1}, \ldots, \vec{w_k}$, with $\vec{w_j}$ being a modified vector $\vec{v_j}$, defined as:*

$$\vec{w_j} := \vec{v_j} + T_j \cdot \vec{m}.$$

Also, when applicable, the function shift takes as input a sequence of scalars $v_1, \ldots, v_k$. Then we write $shift^T(v_1, \ldots, v_k)$ assuming a default basic modification vector (1)*, to denote a sequnce of scalars $w_1, \ldots, w_k$, defined as:*

$$w_j := v_j + T_j.$$

Now we are ready to define the extended leakage shiftability property. Informally speaking, this property says that shifting the values of the wires of index $i \notin P(LG)$ in the input bundles of $\widehat{\Gamma}$ results in shifting the extended leakage *only on the index i* in the output bundles. This is formalized below.

Definition 19. *Let $\widehat{\Gamma}$ be a gadget with i input bundles and o output bundles implementing a function g, and let P be its leakage projection function. We say that a pair $(\widehat{\Gamma}, P)$ satisfies an* extended leakage shiftability *property if the following holds: Let $x_1, \ldots, x_i$ be any input to g and suppose that $g(shift^S(x_1, \ldots, x_i)) = shift^T(g(x_1, \ldots, x_i))$ for some sequences S and T of lengths i and o, respectively. For any fixed encodings $\vec{x_1}, \ldots, \vec{x_i}$ of $x_1, \ldots, x_i$, any subset of leaking wires LG and any basic modification vector $\vec{m}$ that is disjoint with the set $P(LG)$ we have*

$$ExtLeak_{\widehat{\Gamma}}^{LG}(shift_{\vec{m}}^S(\vec{x_1}, \ldots, \vec{x_i})) = shift_{\vec{m}}^T(ExtLeak_{\widehat{\Gamma}}^{LG}(\vec{x_1}, \ldots, \vec{x_i})).$$

Here, the Remark 3 applies.

Remark 3. *Whenever the function shift takes as argument some output of the ExtLeak experiment, it is applied* only to the second part of the experiment output*, i.e. the values assigned ot the output bundles of the gadget (see Def.17).*

Based on the following lemma, every gadget used in our construction satisfies the extended leakage shiftability property.

Lemma 5. *Every gadget $\widehat{\Gamma}$ with its leakage projection function, as described in Section 4.2, satisfies the extended leakage shiftability property.*

Proof. If a basic modification vector $\vec{m} = (0, \ldots, 0, 1, 0, \ldots, 0)$ has 1 as j-th coordinate, and 0 as the rest of the coordinates then we say that it is a *basis vector* and we will denote it with $\vec{m} = \vec{e_j}$. Moreover, we will say that a modification vector is of *weight* w if it has exactly w non-zero coordinates.

Throughout the proof we use the following notation:

- $(x'_1, \ldots, x'_i) := \mathit{shift}^S(x_1, \ldots, x_i)$
- $(\vec{x_1}', \ldots, \vec{x_i}') := \mathit{shift}^S_{\vec{m}}(\vec{x_1}, \ldots, \vec{x_i})$
- $(y_1, \ldots, y_o) := g(x_1, \ldots, x_i)$
- $(y'_1, \ldots, y'_o) := g(x'_1, \ldots, x'_i)$

We prove the Lemma by induction over the weight of the basic modification vector $\vec{m}$. Notice that the weight of a basic modification vector is always an odd natural number.

Base case: Suppose that the weight of $\vec{m}$ equals 1. Then $\vec{m}$ is a basis vector, i.e. $\vec{m} = \vec{e_j}$ for some j. Let us define LG' as the set of wires

$$LG' := LG \cup \bigcup_{k \neq j} OUT_k(\widehat{\Gamma}).$$

By the properties of a gadget, and because, by assumption, $\vec{m}$ is disjoint with the projection $P(LG)$, we have $P(LG') = \{1, \ldots, j-1, j+1, \ldots, n\}$. Because of the form of $\vec{m}$, vectors $\vec{x_l}'$ and $\vec{x_l}$ differ only on the j-th coordinate by S_l, for $1 \leq l \leq i$. More precisely, $\vec{x_l}' - \vec{x_l} = S_l \cdot \vec{e_j}$. Hence, by the simulation property of the gadget, the leakage from the set LG' can be simulated perfectly given the values carried by all the input wires except the j-th wires of each input bundle of $\widehat{\Gamma}$. Therefore we have the equality of the leakages

$$\mathit{GadgetLeak}^{LG'}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i}) = \mathit{GadgetLeak}^{LG'}_{\widehat{\Gamma}}(\vec{x_1}' \ldots, \vec{x_i}').$$

Let A be this random variable assignment of the wires in LG'.

Now, let us observe that with all the input wires fixed to $(\vec{x_1}, \ldots, \vec{x_i})$, the values carried by the wires in the set $OUT_j(\widehat{\Gamma})$ are determined by the values carried by the rest of the output wires, i.e. wires in the set $\bigcup_{k \neq j} OUT_k(\widehat{\Gamma})$. It is due to the fact that, by the correctness of the gadget, the output must encode $(y_1, \ldots, y_o) = g(x_1, \ldots, x_i)$.

Hence, due to the observation above, $\mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})$ is a *deterministic function* of $A = \mathit{GadgetLeak}^{LG'}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})$, parametrized by the output of the function g, here equal to $(y_1, \ldots, y_o)$. Let us call this function $\mathit{complete}_{(y_1,\ldots,y_o)}$. It can be described in details as follows:

1. **Input:** an assignment of wires in the set LG'.
2. Consider the l-th output bundle $B = OUT^l(\widehat{\Gamma})$, where $1 \leq l \leq o$
   - All the wires in B except for the j-th wire $w = OUT^l_j(\widehat{\Gamma})$ are already assigned, as a part of the set LG', with some values $y^1_l, \ldots, y^{j-1}_l, y^{j+1}_l, \ldots, y^n_l$.
   - Assign the wire w with an appropriate value, so that the bundle B carries an encoding of y_l. Namely, it can be done in a unique way by assigning w with
   $$y_l - (\Sigma_{k \neq j} y^k_l)$$
3. **Output:** (values assigned to wires in LG, values assigned to *all* the output wires of the gadged $\widehat{\Gamma}$), like in the Def. 17 of the extended leakage.

Clearly, we have $\mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i}) = \mathit{complete}_{(y_1,\ldots,y_o)}(A)$.

Similarly, $\mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x'_1}, \ldots, \vec{x'_i})$ is a *deterministic function* of the variable $A = \mathit{GadgetLeak}^{LG'}_{\widehat{\Gamma}}(\vec{x'_1}, \ldots, \vec{x'_i})$. More precisely, we have $\mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x'_1}, \ldots, \vec{x'_i}) = \mathit{complete}_{(y'_1,\ldots,y'_o)}(A)$.

From the description above we can conclude that $\mathit{complete}_{(y'_1,\ldots,y'_o)}(A) = \mathit{shift}^T_{\vec{m}}(\mathit{complete}_{(y_1,\ldots,y_o)}(A))$. Indeed, let us consider a *fixed* assignment of the wires in LG'. Suppose that it assigns all the wires in l-th output bundle except the j-th one, i.e. $\{OUT^l_k(\widehat{\Gamma}) | k \neq j\}$, with values $y^1_l, \ldots, y^{j-1}_l, y^{j+1}_l, \ldots, y^n_l$.

Then the wire $OUT^l_j(\widehat{\Gamma})$ will be assigned with

$$y_l - (\Sigma_{k \neq j} y^k_l)$$

by the function $complete_{(y_1,\ldots,y_o)}$, and with

$$y'_l - (\Sigma_{k \neq j} y^k_l)$$

by the function $complete_{(y'_1,\ldots,y'_o)}$. The difference between the latter value and the former one equals

$$y'_l - y_l = T_l.$$

Thus, we have $complete_{(y'_1,\ldots,y'_o)}(A) = shift^T_{\vec{m}}(complete_{(y_1,\ldots,y_o)}(A))$.

Now, the thesis of the lemma follows from the equality

$$\begin{aligned} ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}}(\vec{x_1},\ldots,\vec{x_i})) &= ExtLeak^{LG}_{\widehat{\Gamma}}(\vec{x'_1},\ldots,\vec{x'_i}) \\ &= complete_{(y'_1,\ldots,y'_o)}(A) \\ &= shift^T_{\vec{m}}(complete_{(y_1,\ldots,y_o)}(A)) \\ &= shift^T_{\vec{m}}(ExtLeak^{LG}_{\widehat{\Gamma}}(\vec{x_1},\ldots,\vec{x_i})), \end{aligned}$$

what finishes the proof of the inductive base case.

Inductive step: Now, let $\vec{m}$ be a basic modification vector of weight at least 3 (recall that it must be an odd number), disjoint with the set $P(LG)$. Then, we can express the vector $\vec{m}$ as

$$\vec{m} = \vec{m}' + \vec{m}^+ - \vec{m}^-,$$

where $\vec{m}^+$ and $\vec{m}^-$ are some basis modification vectors, and $\vec{m}'$ is a basic modification vector of weight smaller than $\vec{m}$, all three disjoint with $P(LG)$ as well.

Notice that because $\vec{m}^+$ and $\vec{m}^-$ are basis modification vectors, the sequence of vectors $shift^S_{\vec{m}^+ - \vec{m}^-}(\vec{x_1},\ldots,\vec{x_i})$ encodes the sequence $(x_1,\ldots,x_i)$. Therefore, based on the inductive assumption we have

$$\begin{aligned} &ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}'}(shift^S_{\vec{m}^+ - \vec{m}^-}(\vec{x_1},\ldots,\vec{x_i}))) \\ &= shift^T_{\vec{m}'}(ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}^+ - \vec{m}^-}(\vec{x_1},\ldots,\vec{x_i}))). \end{aligned}$$

Moreover, the assumed equality $g(shift^S(x_1, \ldots, x_i)) = shift^T(g(x_1, \ldots, x_i))$ implies that $g(shift^{-S}(x'_1, \ldots, x'_i)) = shift^{-T}(g(x'_1, \ldots, x'_i))$, and it is clear that $shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i})$ encodes $(x'_1, \ldots, x'_i)$. Therefore, by the inductive base case we have

$$\begin{aligned} &ExtLeak^{LG}_{\widehat{\Gamma}}(shift^{-S}_{\vec{m}^-}(shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i}))) \\ &= shift^{-T}_{\vec{m}^-}(ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i}))). \end{aligned}$$

Similarly, by the inductive base case we also have

$$ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i})) = shift^T_{\vec{m}^+}(ExtLeak^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})).$$

Now, combining the three above equalities gives us

$$\begin{aligned} ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}}(\vec{x_1}, \ldots, \vec{x_i})) &= ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}'+\vec{m}^+-\vec{m}^-}(\vec{x_1}, \ldots, \vec{x_i})) \\ &= ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}'}(shift^S_{\vec{m}^+-\vec{m}^-}(\vec{x_1}, \ldots, \vec{x_i}))) \\ &= shift^T_{\vec{m}'}(ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}^+-\vec{m}^-}(\vec{x_1}, \ldots, \vec{x_i}))) \\ &= shift^T_{\vec{m}'}(ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{-\vec{m}^-}(shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i})))) \\ &= shift^T_{\vec{m}'}(ExtLeak^{LG}_{\widehat{\Gamma}}(shift^{-S}_{\vec{m}^-}(shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i})))) \\ &= shift^T_{\vec{m}'}(shift^{-T}_{\vec{m}^-}(ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i})))) \\ &= shift^T_{\vec{m}'}(shift^T_{-\vec{m}^-}(ExtLeak^{LG}_{\widehat{\Gamma}}(shift^S_{\vec{m}^+}(\vec{x_1}, \ldots, \vec{x_i})))) \\ &= shift^T_{\vec{m}'}(shift^T_{-\vec{m}^-}(shift^T_{\vec{m}^+}(ExtLeak^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})))) \\ &= shift^T_{\vec{m}'-\vec{m}^-+\vec{m}^+}(ExtLeak^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})) \\ &= shift^T_{\vec{m}}(ExtLeak^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})), \end{aligned}$$

what finishes the inductive step. □

5.5 Refreshed gadget reconstruction

In the security proof for our construction, given in Chapter 6, in some experiments we reason about the leakage from the transformed circuit $\widehat{C}$ when the input values for each individual gadget are fixed. The values assigned to all the leaking wires in $\widehat{C}$ are reconstructed based on these fixed values. In this section we define and prove the properties of such reconstruction.

Refreshed gadget. Suppose that a gadget $\widehat{\Gamma}$ has i input bundles and o output bundles. Consider a subcircuit $\widehat{\Gamma}^R$ which consists of $\widehat{\Gamma}$ with all o output bundles being refreshed. By this we mean that $\widehat{\Gamma}^R$ consists of $\widehat{\Gamma}$ and the refreshing gadgets applied to all its output bundles. We will call it *a refreshed gadget*. See an example on Fig. 5.1.

Firstly, we define a *reconstruct* variable for a given gadget $\widehat{\Gamma}$, that allows us to express the *reconstruction invariability* property, presented in Def. 21.

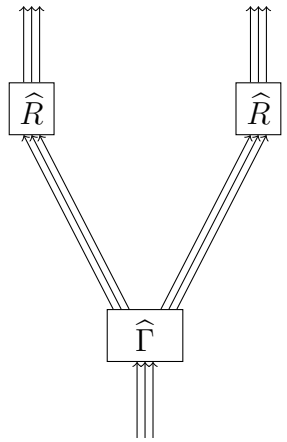


Figure 5.1: An example of a refreshed gadget $\widehat{\Gamma}^R$, where the gadget $\widehat{\Gamma}$ has one input bundle and two output bundles.

Definition 20. *Suppose that a gadget $\widehat{\Gamma}$ implements a function g and that $g(x_1, \ldots, x_i) = (z_1, \ldots, z_o)$. Let $\vec{x_1}, \ldots, \vec{x_i}, \vec{z_1}, \ldots, \vec{z_o}$ be any encodings of these inputs and outputs of g, respectively. Let LG be a set of leaking wires in $\widehat{\Gamma}$ and $LR_1, \ldots, LR_o$ sets of leaking wires in the refreshing bundles of the refreshed gadget $\widehat{\Gamma}^R$, respectively. Let $\vec{x_1}, \ldots, \vec{x_i}$ be fixed vectors assigned to input bundles of $\widehat{\Gamma}^R$ and $\vec{z_1}, \ldots, \vec{z_o}$ be fixed vectors assigned to the output bundles of $\widehat{\Gamma}^R$ i.e. output bundles of $\widehat{\Gamma}$* after *refreshing. We define a random variable*

$$reconstruct_{\widehat{\Gamma}}^{LG, LR_1, \ldots, LR_o}(\vec{x_1}, \ldots, \vec{x_i}, \vec{z_1}, \ldots, \vec{z_o})$$

as the leakage from wires in the set $LG \cup LR_1 \cup \cdots \cup LR_o$ when input and output bundles values of $\widehat{\Gamma}^R$ are fixed to $\vec{x_1}, \ldots, \vec{x_i}$ and $\vec{z_1}, \ldots, \vec{z_o}$, respectively (with the randomness introduced within the gadget $\widehat{\Gamma}$). See also Remark 4.

Remark 4. *Notice that the variable reconstruct*$^{LG,LR_1,\ldots,LR_o}_{\widehat{\Gamma}}(\vec{x_1},\ldots,\vec{x_i},\vec{z_1},\ldots,\vec{z_o})$ does not *include the leakage from the output wires of the refreshed gadget* $\widehat{\Gamma}^R$*, i.e. the output bundles of the gadget* $\widehat{\Gamma}$ after *being refreshed. Such definition is purposeful, as in the security proof these wires are considered a part of another gadgets' input, according to the transformed cirucit* $\widehat{C}$ *construction.*

In other words, *reconstruct*$^{LG,LR_1,\ldots,LR_o}_{\widehat{\Gamma}}(\vec{x_1},\ldots,\vec{x_i},\vec{z_1},\ldots,\vec{z_o})$ are the values assigned to wires in the set $LG \cup LR_1 \cup \cdots \cup LR_o$, conditioned on the inputs of $\widehat{\Gamma}^R$ being $\vec{x_1},\ldots,\vec{x_i}$ and the outputs being $\vec{z_1},\ldots,\vec{z_o}$.

Now we can define the reonstruction invariability property.

Definition 21. *We say that a gadget* $\widehat{\Gamma}$ *with a leakage projection function* P*, implementing a function* g*, satisfies* a reconstruction invariability *property if the following holds. Suppose that* $g(x_1,\ldots,x_i) = (z_1,\ldots,z_o)$ *and that vectors* $\vec{x_1},\ldots,\vec{x_i},\vec{z_1},\ldots,\vec{z_o}$ *encode the values* $x_1,\ldots,x_i,z_1,\ldots,z_o$*. Moreover, suppose that* $g(x'_1,\ldots,x'_i) = (z'_1,\ldots,z'_o)$*. Let* $\vec{m},\vec{m}_1,\ldots,\vec{m}_o$ *be some basic modification vectors. We define*

$$\vec{x_k}' = \vec{x_k} + (x'_k - x_k)\cdot\vec{m} \quad \text{for } 1 \le k \le i$$

and

$$\vec{z_k}' = \vec{z_k} + (z'_k - z_k)\cdot\vec{m}_k \quad \text{for } 1 \le k \le o.$$

If the set of leaking wires LG *in* $\widehat{\Gamma}$*, sets of leaking wires* $LR_1,\ldots,LR_o$ *in the refreshing bundles of* $\widehat{\Gamma}^R$ *and basic modification vectors* $\vec{m},\vec{m}_1,\ldots,\vec{m}_o$ *satisfy the two conditions:*

- $\vec{m}$ *is disjoint with the set* $P(LG)$
- $\vec{m}$ *and* $\vec{m}_k$ *are indistinguishable under the set* $P_R(LR_k)$ *for* $1 \le k \le o$,

then the following equality holds:

$$\begin{aligned} &\textit{reconstruct}^{LG,LR_1,\ldots,LR_o}_{\widehat{\Gamma}}(\vec{x_1}',\ldots,\vec{x_i}',\vec{z_1}',\ldots,\vec{z_o}') = \\ &\qquad \textit{reconstruct}^{LG,LR_1,\ldots,LR_o}_{\widehat{\Gamma}}(\vec{x_1},\ldots,\vec{x_i},\vec{z_1},\ldots,\vec{z_o}). \end{aligned}$$

The privacy proof for our construction is crucially based on the following lemma.

Lemma 6. *Every regular gadget $\widehat{\Gamma}$ with its projection function P, as described in Section 4.2, satisfies reconstruction invariability property.*

Proof. First, we prove the following claim on the reconstruction of the wires in the refreshing bundles (see Def. 12):

Claim 5. *Let LR be some subset of the refreshing bundle $B_{\widehat{R}}$. Suppose that the modification vectors $\vec{m}^1$ and $\vec{m}^2$ are indistinguishable under the set $P_R(LR)$. Then for any vectors $\vec{x}, \vec{y}$ encoding the same value we have (see Def. 12 of RefRec)*

$$\mathit{RefRec}^{LR}(\vec{x}+\vec{m}^1, \vec{y}+\vec{m}^2) = \mathit{RefRec}^{LR}(\vec{x}, \vec{y}).$$

Proof. It is enough to show that for any wire $w \in LR$, $\mathit{RefRec}^{LR}(\vec{x}+\vec{m}^1, \vec{y}+\vec{m}^2)$ and $\mathit{RefRec}^{LR}(\vec{x}, \vec{y})$ assign w the same value. Let us denote with v_1 and v_2 the respective assignments. Based on the construction of the refreshing gadget (see Fig. 2.1), there are three possible cases:

Case 1: w is of the form $b_1 \oplus \cdots \oplus b_k$ where $1 \leq k \leq n-1$. Then $k \in P_R(LR)$, and because $\vec{m}^1$ and $\vec{m}^2$ are indistinguishable under the set $P_R(LR)$ we have

$$v_1 = \sum_{i=1}^{k}(x_i+m_i^1-y_i-m_i^2) = \sum_{i=1}^{k}(x_i-y_i)+\sum_{i=1}^{k} m_i^1-\sum_{i=1}^{k} m_i^2 = \sum_{i=1}^{k}(x_i-y_i) = v_2.$$

Case 2: w is of the form $-(b_1 \oplus \cdots \oplus b_{n-1})$. Then the equality $v_1 = v_2$ is implied by Case 1 for $k = n-1$.

Case 3: w is of the form b_k for some $2 \leq k \leq n-1$. Then, by definition of the set $P_R(LR)$, we have $k-1, k \in P_R(LR)$. From the argument used in Case 1 we can conclude that both of the wires $w_1 = b_1 \oplus \cdots \oplus b_k$ and $w_2 = b_1 \oplus \cdots \oplus b_{k-1}$ are assigned with the same value by the reconstruction $\mathit{RefRec}^{LR}(\vec{x}+\vec{m}^1, \vec{y}+\vec{m}^2)$ and by the reconstruction $\mathit{RefRec}^{LR}(\vec{x}, \vec{y})$. Therefore, because the value assigned to w is a difference of these two, it is the same in both reconstructions. □

Now, we prove the equality of the reconstructions $\mathit{reconstruct}_{\widehat{\Gamma}}^{LG,LR_1,\ldots,LR_o}(\vec{x_1}', \ldots, \vec{x_i}', \vec{z_1}', \ldots, \vec{z_o}')$ and $\mathit{reconstruct}_{\widehat{\Gamma}}^{LG,LR_1,\ldots,LR_o}(\vec{x_1}, \ldots, \vec{x_i}, \vec{z_1}, \ldots, \vec{z_o})$, postulated by the Def. 21 with a hybrid argument. We define three hybrid experiments, such that the first produces an assignment equal to one of the reconstructions, the last one produced an assignment equal to the other reconstruction, and any two consecutive hybrids have the same output.

$\underline{\mathsf{Hybrid}_1}$:

1. Compute

$$(A^1, \vec{Y_1^1}, \ldots, \vec{Y_o^1}) = \mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i}),$$

   where A^1 is the assignment to the wires in LG and $\vec{Y_1^1}, \ldots, \vec{Y_o^1}$ is the assignment to the output bundles of $\widehat{\Gamma}$ (compare with Def. 17).

2. Recover the leakage from the sets $LR_1, \ldots, LR_o$. Namely, compute

$$A_k^1 = \mathit{RefRec}^{LR_k}(\vec{Y_k^1}, \vec{z_k}) \quad \text{for } 1 \leq k \leq o.$$

3. **Output:** $(A^1, A_1^1, \ldots, A_o^1)$

Clearly, the output of the experiment Hybrid_1 equals to $\mathit{reconstruct}^{LG, LR_1, \ldots, LR_o}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i}, \vec{z_1}, \ldots, \vec{z_o})$, because in both cases it is *determined*, in the same manner, by the inputs and outputs of the refreshed gadget $\widehat{\Gamma}^R$, and the internal randomness of the gadget $\widehat{\Gamma}$.

$\underline{\mathsf{Hybrid}_2}$:

1. Compute

$$(A^2, \vec{Y_1}', \ldots, \vec{Y_o}') = \mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}', \ldots, \vec{x_i}'),$$

where A^2 is the assignment to the wires in LG and $\vec{Y_1}', \ldots, \vec{Y_o}'$ is the assignment to the output bundles of $\widehat{\Gamma}$.

2. "Shift back" the assignments of the output bundles by appropriate modification vectors, namely compute

$$\vec{Y^2_k} = \vec{Y_k}' - (z'_k - z_k) \cdot \vec{m} \quad \text{for } 1 \leq k \leq i.$$

3. Recover the leakage from the sets $LR_1, \ldots, LR_o$. More precisely, compute

$$A^2_k = \mathit{RefRec}^{LR_k}(\vec{Y^2_k}, \vec{z_k}) \quad \text{for } 1 \leq k \leq o.$$

4. **Output:** $(A^2, A^2_1, \ldots, A^2_o)$

To show that Hybrid_1 and Hybrid_2 produce the same output, let us first define sequences $S = (x'_1 - x_1, \ldots, x'_i - x_i)$ and $T = (z'_1 - z_1, \ldots, z'_o - z_o)$ (compare with Def. 21). Then we have

$$g((x_1, \ldots, x_i) + S) = g(x'_1, \ldots, x'_i) = (z'_1, \ldots, z'_i) = (z_1, \ldots, z_i) + T.$$

Hence

$$g(\mathit{shift}^S(x_1, \ldots, x_i)) = \mathit{shift}^T(g(x_1, \ldots, x_i)).$$

Moreover, by assumption, vector $\vec{m}$ is disjoint with the set $P(LG)$. Thus, we can apply Lemma 5 obtaining the equality

$$\mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\mathit{shift}^S_{\vec{m}}(\vec{x_1}, \ldots, \vec{x_i})) = \mathit{shift}^T_{\vec{m}}(\mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})).$$

Notice also that, by definition of S, we have

$$(\vec{x_1}', \ldots, \vec{x_i}') = \mathit{shift}^S_{\vec{m}}(\vec{x_1}, \ldots, \vec{x_i}).$$

Hence,

$$\begin{aligned}(A^2, \vec{Y_1}', \ldots, \vec{Y_o}') &= \mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}', \ldots, \vec{x_i}') = \mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\mathit{shift}^S_{\vec{m}}(\vec{x_1}, \ldots, \vec{x_i})) \\ &= \mathit{shift}^T_{\vec{m}}(\mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}, \ldots, \vec{x_i})) = \mathit{shift}^T_{\vec{m}}(A^1, \vec{Y_1}, \ldots, \vec{Y_o}) \\ &= (A^1, \vec{Y^1_1} + (z'_1 - z_1) \cdot \vec{m}, \ldots, \vec{Y^1_o} + (z'_o - z_o) \cdot \vec{m})\end{aligned}$$

and therefore, because by definition $\vec{Y_k^2} = \vec{Y_k}' - (z'_k - z_k) \cdot \vec{m}$ for each k, we have

$$\begin{aligned}(A^2, \vec{Y_1^2}, \ldots, \vec{Y_o^2}) =& (A^2, \vec{Y_1}' - (z'_1 - z_1) \cdot \vec{m}, \ldots, \vec{Y_o}' - (z'_o - z_o) \cdot \vec{m}) \\ =& (A^1, \vec{Y_1} + (z'_1 - z_1) \cdot \vec{m} - (z'_1 - z_1) \cdot \vec{m}, \ldots, \\ & \vec{Y_o} + (z'_o - z_o) \cdot \vec{m} - (z'_o - z_o) \cdot \vec{m}) \\ =& (A^1, \vec{Y_1^1}, \ldots, \vec{Y_o^1}).\end{aligned}$$

Thus, the joint distribution of $(A^2, \vec{Y_1^2}, \ldots, \vec{Y_o^2})$ in Hybrid_2 is the same as the joint distribution of $(A^1, \vec{Y_1^1}, \ldots, \vec{Y_o^1})$ in Hybrid_1. Therefore also their outputs $(A^1, A_1^1, \ldots, A_o^1)$ and $(A^2, A_1^2, \ldots, A_o^2)$ are equal, because they are *determined*, in the same manner, by these joint variables.

Hybrid_3:

1. Compute

$$(A^3, \vec{Y_1^3}, \ldots, \vec{Y_o^3}) = \mathit{ExtLeak}^{LG}_{\widehat{\Gamma}}(\vec{x_1}', \ldots, \vec{x_i}'),$$

   where A^3 is assignment to wires in LG and $\vec{Y_1^3}, \ldots, \vec{Y_o^3}$ is assignment to the output bundles of $\widehat{\Gamma}$.

2. Recover the leakage from sets $LR_1, \ldots, LR_o$. More precisely, compute

$$A_k^3 = \mathit{RefRec}^{LR_k}(\vec{Y_k^3}, \vec{z_k}') \quad \text{for } 1 \leq k \leq o.$$

3. **Output:** $(A^3, A_1^3, \ldots, A_o^3)$

Clearly, the output of this experiment equals $\mathit{reconstruct}^{LG, LR_1, \ldots, LR_o}_{\widehat{\Gamma}}(\vec{x_1}', \ldots, \vec{x_i}', \vec{z_1}', \ldots, \vec{z_o}')$, because in both cases it is *determined*, in the same manner, by the fixed inputs and outputs of the refreshed gadget $\widehat{\Gamma}^R$, and the internal randomness of the gadget $\widehat{\Gamma}$.

On the other hand, to show that Hybrid_2 and Hybrid_3 produce the same output, observe that, by construction of the hybrids, we have:

$$(A^3, \vec{Y_1^3}, \ldots, \vec{Y_o^3}) = (A^2, \vec{Y_1^2} + (z'_1 - z_1) \cdot \vec{m}, \ldots, \vec{Y_o^2} + (z'_o - z_o) \cdot \vec{m}).$$

Morever, by assumption, for any k the vectors $\vec{m}$ and $\vec{m}_k$ are indistinguishable under the set $P_R(LR_k)$ and, as a consequence, the vectors $(z'_k - z_k) \cdot \vec{m}$ and $(z'_k - z_k) \cdot \vec{m}_k$ are indistinguishable under that set as well. Hence, using Lemma 5, we have that for any pair of *fixed* vectors $\vec{y^3_k}, \vec{y^2_k}$ satisfying $\vec{y^3_k} = \vec{y^2_k} + (z'_k - z_k) \cdot \vec{m}$:

$$\begin{aligned} \mathit{RefRec}^{LR_k}(\vec{y^3_k}, \vec{z_k}') &= \mathit{RefRec}^{LR_k}(\vec{y^2_k} + (z'_k - z_k) \cdot \vec{m}, \vec{z_k} + (z'_k - z_k) \cdot \vec{m}_k) \\ &= \mathit{RefRec}^{LR_k}(\vec{y^2_k}, \vec{z_k}). \end{aligned}$$

Now, because of how the variables A^2_k in Hybrid_2 and A^3_k in Hybrid_3 are defined, namely as

$$A^2_k = \mathit{RefRec}^{LR_k}(\vec{Y^2_k}, \vec{z_k})$$

and

$$A^3_k = \mathit{RefRec}^{LR_k}(\vec{Y^3_k}, \vec{z_k}'),$$

we can conclude that the output $(A^3,\ A^3_1, \ldots, A^3_o)$ of Hybrid_3 is the same as the output $(A^2,\ A^2_1, \ldots, A^2_o)$ of Hybrid_2. This finishes the proof of the lemma. □

Chapter 6

The main theorem: privacy of the construction

In this chapter we present and prove the main theorem of our work.

Theorem 1. *Let C be any arithmetic circuit and $\widehat{C}$ its transformation as described in Section 4.1. Assume that for all gadgets used in $\widehat{C}$ the projection probability functions are upper-bounded by a function $q : [0,1] \to \mathbb{R}$, which also upper-bounds the function $f(p) = p + 2\sqrt{3p}$. Then $\widehat{C}$ is sound implementation of C (see Def. 4) and $\widehat{C}$ is $(p, |C| \cdot (4q(p))^n)$-private (see Def. 6) for any probability p, where $|C|$ is the number of gates in the circuit C.*

This theorem is proven along the lines of the intuitions presented in Sect. 2. For ease of presentation we give a proof overview first and then the detailed proof below.

Proof overview. The soundness of the transformed ciruit $\widehat{C}$ follows easily from its construction, as each of its gadgets corresponds to a gate in the original circuit C and has the same functionality over the encodings. Thus, $\widehat{C}$ has the same functionality as the original circuit C.

To prove the privacy of our construction, we will show that any two inputs X_1, X_2 to circuit $\widehat{C}$ induce leakages that are close in terms of statistical distance. To do this, we compare these two leakages conditioned on the set of leaking wires being some *fixed* set. Let S be a leakage diagram induced by W. We show that if the leftmost and rightmost sides of the graph S are not connected then the two conditioned leakages are actually identical.

To this end, we use a classic hybrid argument, showing a sequence of so called hybrid experiments and argumenting that every two consecutive ones produce identical output. Here we briefly describe the hybrid experiments:

Hybrid_1 (this corresponds to experiment Exp_A^0 in Sect. 2.1): simply outputs the leakage when $\widehat{C}$ is fed with X_1.

Hybrid_2 (this corresponds to experiment Exp_B^0): in this experiment each gadget in $\widehat{C}$ is evaluated separately, and the assignment of the refreshing bundles between the gadgets are derived from there. To this end, we consider the evaluation of the original circuit C when fed with X_1. If a particular wire w in C, which is an input to a gate Γ, is assigned with a value v then the respective input bundle in the gadget $\widehat{\Gamma}$ in $\widehat{C}$ is fed with a freshly chosen random encoding $\vec{v} \leftarrow \mathrm{Enc}(v)$. Then each gadget in $\widehat{C}$ is evaluated accordingly to the chosen inputs. This determines the assignment of all the refreshing bundles in $\widehat{C}$. The output of the experiment consists of the values assigned to wires in W.

Hybrid_3 (this corresponds to experiment Exp_C): this experiment is the same as experiment Hybrid_2, except for the random vectors that are assigned to the input bundles of each individual gadget. Here, after choosing a random encoding $\vec{v} \leftarrow \mathrm{Enc}(v)$ just as in Hybrid_2, we shift it by carefully chosen modification vector $\vec{m}$. As a result, we feed the particular input bundle with $\vec{v} + \vec{m}$. The modification vector for the input bundles of each gadget is constructed based on inputs X_1 and X_2, and the leakage diagram S. At this point we use the fact that the leftmost and rightmost sides of the leakage diagram S are not connected. The details of the construction of modification vectors are given in the full proof of the theorem.

Based on the properties of the refreshed gadgets subcircuits in $\widehat{C}$ and the construction of the modification vectors, we argue that shifting values that are fed to each gadget actually does not change the leakage from the set of wires W. Hence this experiment outputs the same random variable as Hybrid_2.

Hybrid_4 (this corresponds to experiment Exp_B^1): this experiment is analogous to the Hybrid_2, with input X_2 instead of X_1. We argue that the random vectors assigned to the input bundles of each individual gadget in are actually the same in this experiment and in Hybrid_3. Hence, the two experiments produce identical outputs.

Hybrid_5 (this corresponds to experiment Exp_A^1) : this experiment is analogous to the Hybrid_1, with input X_2 instead of X_1. Also the transition between

Hybrid_4 and this experiment is analogous to the transition for experiments Hybrid_1 and Hybrid_2.

The hybrid argument above essentially shows that unless the leftmost and rightmost sides of the leakage diagram S are connected, the leakage is the same independently of the input X fed to the transformed circuit $\widehat{C}$.

Now, to complete the privacy proof, it is enough to upper-bound the probability of the leftmost and rightmost sides of S being connected. This is a pure probability theory exercise, given that $q(p)$ upper-bounds the leakage projection function of the gadgets used, which means that each edge will be included to the leakage diagram independently with probability at most $q(p)$.

Next, we give a full proof of Thm. 1:

Proof. By construction, $\widehat{C}$ is a sound transformation of circuit C because during transformation each gate is replaced with a gadget with the same functionality for the encodings of the intermediate values (see Sect. 4.1).

To prove the privacy of the transformed circuit, let us define E as an event in the experiment *Leak* (see Def. 8) for circuit C when the leftmost and rightmost sides of the leakage diagram, representing the set of the leaking wires, are connected. The proof is based on the two following claims about the event E.

Claim 6. *Unless the event E occurs in the Leak experiment, the privacy of circuit $\widehat{C}$ is preserved, i.e. adversary does not learn from the leakage any information about the input X. More precisely, for any two inputs X_1, X_2 the two adversarial views are equal:*

$$(Leak(C, X_1, p)|\neg E) = (Leak(C, X_2, p)|\neg E).$$

Proof. Let LW be the random set of the leaking wires in the experiment *Leak*. By definition, the occurrence of event E is determined by this set. Hence, it is enough to show that for any *fixed* set of wires W in $\widehat{C}$ for which the associated leakage diagram S has its leftmost and rightmost sides disconnected, the following holds

$$(Leak(C, X_1, p)|LW = W) = (Leak(C, X_2, p)|LW = W). \tag{6.1}$$

Let W be any of such sets and let S be the corresponding leakage diagram, as defined in Sect. 5.2. Now, based on S, we construct the basic modification vectors (see Sect. 5.3), one for each of the gadgets in $\widehat{C}$. Let LS be the connected component of S that contains its leftmost side. Notice that, by

assumption on event E, LS *does not* contain any nodes of the rightmost side of S.

Consider any gadget $\widehat{\Gamma}$ in $\widehat{C}$ and its corresponding crosswise path $N_0^{\widehat{\Gamma}}, \ldots, N_n^{\widehat{\Gamma}}$ (see Sect. 5.2). We define the corresponding basic modification vector $\vec{m}^{\widehat{\Gamma}}$ of length n as:

$$\vec{m}^{\widehat{\Gamma}} = (\mathbf{1}_{LS}(N_0^{\widehat{\Gamma}}) - \mathbf{1}_{LS}(N_1^{\widehat{\Gamma}}), \mathbf{1}_{LS}(N_1^{\widehat{\Gamma}}) - \mathbf{1}_{LS}(N_2^{\widehat{\Gamma}}), \ldots, \mathbf{1}_{LS}(N_{n-1}^{\widehat{\Gamma}}) - \mathbf{1}_{LS}(N_n^{\widehat{\Gamma}})), \tag{6.2}$$

where $\mathbf{1}_{LS}(\cdot)$ is an indicator function for the set of nodes in LS. Notice that all coordinates of $\vec{m}^{\widehat{\Gamma}}$ equal -1, 0 or 1. Moreover LS *does not* contain $N_n^{\widehat{\Gamma}}$ because this node belongs to the rightmost side of S. Hence, it is easy to see that the sum of the coordinates of $\vec{m}^{\widehat{\Gamma}}$ equals 1 and it indeed satisfies the basic modification vector definition.

Now we show the following lemmas about the assigned modification vectors, used later in the hybrid argument:

Lemma 7. *Let $\widehat{\Gamma}$ be any gadget in $\widehat{C}$, and LG the set of its leaking wires, i.e. subset of W that are in $\widehat{\Gamma}$. Let $P(\cdot)$ be its projection function (see Sect. 4.2). Then the basic modification vector $\vec{m}^{\widehat{\Gamma}}$ is disjoint (see Sect. 5.3) with the set $P(LG)$.*

Proof. Suppose that $k \in P(LG)$. By construction of the leakage diagram S (see Sect. 5.2), the nodes $N_{k-1}^{\widehat{\Gamma}}$ and $N_k^{\widehat{\Gamma}}$ are connected. Thus, either both of them or none of them belong to the connected component of the leakaged diagram LS. So, by definition of the vector $\vec{m}^{\widehat{\Gamma}}$, its k-th coordinate equals to $\mathbf{1}_{LS}(N_{k-1}^{\widehat{\Gamma}}) - \mathbf{1}_{LS}(N_k^{\widehat{\Gamma}}) = 0$, which ends the proof. □

Lemma 8. *Let $\widehat{\Gamma}_1, \widehat{\Gamma}_2$ be any two connected gadgets in $\widehat{C}$ (see Sect. 4.1) with a refreshing bundle $B_{\widehat{R}}$ (see Sect. 5.1) between. Let LR be a leaking subset of $B_{\widehat{R}}$, i.e. $LR = W \cap B_{\widehat{R}}$. Then $\vec{m}^{\widehat{\Gamma}_1}$ and $\vec{m}^{\widehat{\Gamma}_2}$ are indistinguishable under the set $P_R(LR)$ (see Def. 15).*

Proof. Let $A_1 \subset \{0, \ldots, n\}$ be the set of indices of the nodes in S respective to $\widehat{\Gamma}_1$, i.e. $\{N_0^{\widehat{\Gamma}_1}, \ldots, N_n^{\widehat{\Gamma}_1}\}$, that belong to LS. Similarly, we define $A_2 \subset \{0, \ldots, n\}$ as the set of indices of the nodes belonging to LS respective to $\widehat{\Gamma}_2$.

Let $k \in P_R(LR)$. We will show that $\sum_{i=1}^{k} \vec{m}_i^{\widehat{\Gamma}_1} = \sum_{i=1}^{k} \vec{m}_i^{\widehat{\Gamma}_2}$. By the definition of $\vec{m}^{\widehat{\Gamma}_1}$ and $\vec{m}^{\widehat{\Gamma}_2}$ we have

$$\sum_{i=1}^{k} \vec{m}_i^{\widehat{\Gamma}_1} = \mathbf{1}_{LS}(N_0^{\widehat{\Gamma}_1}) - \mathbf{1}_{LS}(N_k^{\widehat{\Gamma}_1}),$$

and similarly

$$\sum_{i=1}^{k} \vec{m}_i^{\widehat{\Gamma}_2} = \mathbf{1}_{LS}(N_0^{\widehat{\Gamma}_2}) - \mathbf{1}_{LS}(N_k^{\widehat{\Gamma}_2}).$$

Notice that, by definition, the nodes $N_0^{\widehat{\Gamma}_1}$ and $N_0^{\widehat{\Gamma}_2}$ belong to LS, and therefore $\mathbf{1}_{LS}(N_0^{\widehat{\Gamma}_1}) = \mathbf{1}_{LS}(N_0^{\widehat{\Gamma}_2}) = 1$. Also, because $k \in P_R(LR)$, the edge between the nodes $N_k^{\widehat{\Gamma}_1}$ and $N_k^{\widehat{\Gamma}_2}$ belong to the leakage diagram S. Hence, either both of them or none of them belong to LS. Therefore we also have $\mathbf{1}_{LS}(N_k^{\widehat{\Gamma}_1}) = \mathbf{1}_{LS}(N_k^{\widehat{\Gamma}_2})$, which finishes the proof of the lemma. □

Now, we will argue that Eq. (6.1) holds by a hybrid argument in which every experiment in the sequence produces an *identical* output - the adversarial view. Here, C' denotes the circuit C after the preprocessing phase (see Sect. 4.1). The hybrids are as follows.

- Hybrid_1: outputs the leakage from wires in W when $\widehat{C}$ is fed with X_1.
- Hybrid_2: in this experiment, firstly each gadget in $\widehat{C}$ is evaluated separately. In order to assign the values to the input bundles of the gadgets, we consider the evaluation of the circuit C' when fed with X_1. If some wire in C', that is input to a gate Γ, is assigned with the value v then the respective input bundle of the gadget $\widehat{\Gamma}$ in $\widehat{C}$ is fed with a freshly chosen random encoding $\vec{v} \leftarrow \mathrm{Enc}(v)$. Gadget $\widehat{\Gamma}$ is evaluated with these encodings as inputs.

  Then the assignment of the refreshing bundles in $\widehat{C}$ are derived from the values already assigned to the bundle before, with value $\vec{x}$, and the bundle after the refreshing, assigned with $\vec{y}$. Indeed, these values are *uniquely determined* by $\mathit{RefRec}(\vec{x}, \vec{y})$ (see Def. 12).

  The output of the experiment consists of values assigned to the leaking wires in W.

- Hybrid_3: this is the same experiment as Hybrid_2, except for the encodings fed to each of the gadgets as inputs.

  Here, in order to obtain the *modified* encodings we make use of the leakage diagram S induced by the set W of leaking wires (see 5.2). The leakage diagram S determines for each gadget $\widehat{\Gamma}$ in $\widehat{C}$ an associated basic modification vector $\vec{m}^{\widehat{\Gamma}}$, as defined in Eq. 6.2. Let B be an input bundle of a gadget $\widehat{\Gamma}$ in $\widehat{C}$ corresponding to a wire w in the preprocessed circuit C' (see Sect. 4.1). Suppose that w carries value v_1 when C' is fed with X_1 and v_2 when C' is fed with X_2. To assign values to the input bundle B, we draw a fresh random encoding $\vec{v} \leftarrow \mathrm{Enc}(v_1)$ and add to it the modification vector $(v_2 - v_1)\vec{m}^{\widehat{\Gamma}}$ - so it is fed with the vector $\vec{v} + (v_2 - v_1)\vec{m}^{\widehat{\Gamma}}$.

  Then, each of the gadgets is evaluated separately and the assignments of all the refreshing bundles in $\widehat{C}$ are derived just like in Hybrid_2, and the output of this experiment is the assignment of the leaking wires in W.

- Hybrid_4: it is an analogous experiment to Hybrid_2, with the only difference being that here we use the input X_2 instead of X_1 to derive the values carried by the wires of the circuit C'.

- Hybrid_5: outputs the leakage from wires in W when the transformed circuit $\widehat{C}$ is fed with X_2.

We now show that the outputs of every two consecutive hybrids are identical, which completes the proof of the claim.

Hybrid_1 to Hybrid_2: We will argue that in these two experiments not only the assignment to the wires in W is identical (as a random variable), but something more: that the assignment of *all* the wires of $\widehat{C}$, as a random variable, is identical.

Observe that in Hybrid_1 every *intermediate* (i.e. carried by a bundle between two gadgets) encoding of a given value v is refreshed using the randomness introduced by some refreshing gadget $\widehat{R}$. The randomness used is independent of the input to $\widehat{R}$. Moreover, this randomness is *uniquely* determined by the inputs and the outputs of $\widehat{R}$. Therefore, there is an equivalence between choosing the randomness used in $\widehat{R}$ in Hybrid_1 and choosing the encoding at random, $\vec{v} \leftarrow \mathrm{Enc}(v)$, and assigning it to the output of $\widehat{R}$ in Hybrid_2.

Thus, in the experiment Hybrid_2 the values assigned to all the wires in $\widehat{C}$ form the same random variable as in Hybrid_1 and the outputs of the experiments are identical.

Hybrid_2 to Hybrid_3:

Both hybrids begin constructing the adversarial view (the leakage from wires in W) by choosing a random encodings assigned to all the input bundles of the gadgets in $\widehat{C}$. In Hybrid_3, before evaluating each gadget, they are shifted by appropriately constructed modification vectors. So it is enough to show that for any such *fixed* initial (i.e., before shifting) input encodings the adversarial view is the same in both hybrids.

Consider any refreshed gadget subcircuit $\widehat{\Gamma}^R$ in $\widehat{C}$ (see Sect. 5.5). We will show that the leakage from this subcircuit, as a random variable (with a randomness possibly introduced during evaluation of the gadget $\widehat{\Gamma}$ if it is randomized), is the same in both hybrids for any fixed initial input encodings. Here, we consider the leakage from the refreshed gadget subcircuit $\widehat{\Gamma}^R$, *excluding* its output wires, just like in Def. 20 of the *reconstruct* variable (see also Remark 4). Notice also that, in both hybrids, such leakages from the refreshed gadgets across the whole circuit $\widehat{C}$ are mutually independent, because each of them is *determined* by the randomness used during the evaluation of the gadget $\widehat{\Gamma}$. Therefore, showing that leakages from just refreshed gadgets in both hybrids are equal will imply that the whole adversarial view, which is a joint distribution of such random variables, is the same in both hybrids.

We will do that using previously proven lemmas on the modification vectors properties and the reconstruction invariability property (see Def. 21) of the gadget $\widehat{\Gamma}$, satisfied based on Lemma 6. Let $\widehat{\Gamma}_1, \ldots, \widehat{\Gamma}_o$ be the gadgets connected with $\widehat{\Gamma}$ by its output bundles. Let $LR_1, \ldots, LR_o \subset W$ be the sets of leaking wires in the refreshing bundles between $\widehat{\Gamma}$ and $\widehat{\Gamma}_1, \ldots, \widehat{\Gamma}_o$, respectively. Let $LG \subset W$ be the set of the leaking wires in the gadget $\widehat{\Gamma}$. Suppose that gate Γ in C' corresponding to the gadget $\widehat{\Gamma}$ in $\widehat{C}$ takes as inputs $x_1, \ldots, x_i$ and outputs $z_1, \ldots, z_o$, when C' is fed with input X_1. Similarly, suppose that Γ takes input values $x'_1 \ldots, x'_i$ and outputs $z'_1, \ldots, z'_o$, when C' is fed with X_2.

Let the fixed input encodings of the refreshed gadget $\widehat{\Gamma}^R$, chosen initially in both hybrids, be $\vec{x_1}, \ldots, \vec{x_i}$. Its output encodings are also fixed, as they are the inputs to $\widehat{\Gamma}_1, \ldots, \widehat{\Gamma}_o$. Let them be $\vec{z_1}, \ldots, \vec{z_o}$. Notice that the aforemen-

tioned leakage from the subcircuit $\widehat{\Gamma}^R$ in Hybrid_2 equals exactly (see Def. 20)

$$reconstruct_{\widehat{\Gamma}}^{LG,LR_1,\ldots,LR_o}(\vec{x_1},\ldots,\vec{x_i},\vec{z_1},\ldots,\vec{z_o}). \tag{6.3}$$

In Hybrid_3, before evaluating the gadget $\widehat{\Gamma}$ the vectors $\vec{x_1},\ldots,\vec{x_i}$ and $\vec{z_1},\ldots,\vec{z_o}$ are shifted by appropriate modification vectors, so that the leakage from $\widehat{\Gamma}^R$ equals

$$reconstruct_{\widehat{\Gamma}}^{LG,LR_1,\ldots,LR_o}(\vec{x'_1},\ldots,\vec{x'_i},\vec{z'_1},\ldots,\vec{z'_o}), \tag{6.4}$$

where

$$\vec{x_k}' = \vec{x_k} + (x'_k - x_k)\cdot\vec{m}^{\widehat{\Gamma}} \quad \text{for } 1 \le k \le i,$$
$$\vec{z_k}' = \vec{z_k} + (z'_k - z_k)\cdot\vec{m}^{\widehat{\Gamma}_k} \quad \text{for } 1 \le k \le o.$$

Based on Lemma 8, the vectors $\vec{m}^{\widehat{\Gamma}}$ and $\vec{m}^{\widehat{\Gamma}_k}$ are indistinguishable under the set $P_R(LR_k)$, for each $k = 1,\ldots,o$. Moreover, based on Lemma 7, the vector $\vec{m}^{\widehat{\Gamma}}$ is disjoint with the set $P(LG)$. Therefore we can use the reconstruction invariability property of the gadget $\widehat{\Gamma}$, which states directly (see Def. 21) that the two variables in 6.3 and in 6.4 are equal.

Hence, the two adversarial views are equal, what justifies the transition from Hybrid_2 to Hybrid_3.

Hybrid_3 to Hybrid_4: Notice that the vectors fed to the input bundles of the gadgets in $\widehat{C}$ in both hybrids are actually drawn from the same distribution, what justifies the transition. Let $\widehat{\Gamma}$ be any gadget in $\widehat{C}$ and B be any of its input bundles, that correspond to the gate Γ and its input wire w in the circuit C'. Suppose that w carries value v_1 when C is fed with the input X_1 and v_2 when it is fed with X_2. In Hybrid_3 we assign bundle B with vector $\vec{v} + (v_2 - v_1)\vec{m}^{\widehat{\Gamma}}$ for a random encoding $\vec{v} \leftarrow \mathrm{Enc}(v_1)$. It is easy to see that, because the sum of the coordinates in the basic modification vector $\vec{m}^{\widehat{\Gamma}}$ is equal to 1, this assignment is a random encoding of the value v_2, exactly like in Hybrid_4. Therefore these two experiments produce the same adversarial view.

Hybrid_4 to Hybrid_5: This transition is perfectly analogous to the transition between Hybrid_1 and Hybrid_2.

The sequence of transitions between the hybrid experiments finishes the proof of the claim. □

The second claim about the event E, on which the theorem proof is based, states:

Claim 7. *The probability of the event E in the experiment $Leak(C, X, p)$ (see Def. 8)is upper-bounded by $|C| \cdot (4q(p))^n$.*

Proof. Recall that we denote by S the leakage diagram representing the leakage in the experiment $Leak(C, X, p)$. In the following we treat it as a random variable, which is determined by the set of leaking wires LW. To prove the claim we describe another random graph, whose set of edges covers (see Def. 1) the set of edges of S.

Namely, let us consider the following experiment $RandJ$:

1. Start with a graph J having the same nodes as graph $G(\widehat{C})$ (see Sect. 5.2) and the edges on its leftmost and rightmost sides.
2. For every edge e in $G(\widehat{C})$ outside of its leftmost and rightmost sides, add e to J *independently* with probability $q(p)$.
3. **Output:** graph J.

We make two observations about this experiment, which combined prove Claim 7.

Observation 1: the set of the edges of $RandJ$, treated as a random variable, has a distribution that covers (as in Def. 1) the distribution of the set of the edges of the leakage diagram S.

To justify this observation firstly notice that, by construction, in both cases the subsets of each vertical matching and the subsets of each crosswise path (see Sect. 5.2) are generated independently. Thus, it is enough to argue that for each component of the decomposition (as in Sect. 5.2) of $G(\widehat{C})$, the distribution of its edges belonging to the leakage diagram S is covered (as in Def. 1) by the distribution of its edges belonging to the graph $RandJ$. For the crosswise paths, this follows directly from the fact that $q(p)$ upper-bounds the projection probability function for each gadget used in the construction of $\widehat{C}$. For the vertical matchings, it follows from Lemma 4 combined with the assumption that $q(p) \geq p + 2\sqrt{3p}$.

Now, let us denote by E^J an event in the experiment $RandJ$ when the leftmost and rightmost sides of J are connected. Notice that the following inequality holds

$$\Pr[E] \leq \Pr[E^J]. \tag{6.5}$$

It is due to the Observation 1, and the fact that adding an edge to any set of edges preserves all the paths in the graph, including the ones connecting its leftmost and rightmost sides (compare with Def. 1).

Let $\widehat{\Gamma}_1, \ldots, \widehat{\Gamma}_{|C|}$ be all the gadgets in circuit $\widehat{C}$. For $1 \leq i \leq |C|$ let us denote by event E_i^J the occurrence of a path in J that begins in the node $N_0^{\widehat{\Gamma}_i}$ on the leftmost side, does not contain any other node from the leftmost side, and ends on the rightmost side of J. For any path connecting the leftmost and the rightmost sides in J, the last node of the form $N_0^{\widehat{\Gamma}_i}$ on this path begins a path in J that connects its leftmost and rightmost sides and does not contain any other nodes from the leftmost side. Therefore, from the union bound we have

$$\Pr[E^J] \leq \Pr[E_1^J] + \cdots + \Pr[E_{|C|}^J].$$

Observation 2: For any i, the probability $\Pr[E_i^J]$ is upper-bounded by $(4q(p))^n$.

To justify this observation notice that, by construction, the degree of each node in graph $G(\widehat{C})$ associated with the circuit $\widehat{C}$ is at most 5. Thus, there are at most 4^n different paths of length exactly n in $G(\widehat{C})$ that begin in the node $N_0^{\widehat{\Gamma}_i}$ on the leftmost side and does not contain any other node from that side. Notice also that for the event E_i^J to occur, at least one of these paths must be included in the random graph J because the distance between the leftmost and the rightmost sides in graph $G(\widehat{C})$ equals exactly n. For each of these paths, the probability of it being included into J equals $q(p)^n$. Hence, by union bound $\Pr[E_i^J] \leq 4^n \cdot q(p)^n = (4q(p))^n$.

The two Observations imply together that

$$\Pr[E] \leq \Pr[E^J] \leq \Pr[E_1^J] + \cdots + \Pr[E_{|C|}^J] \leq |C| \cdot (4q(p))^n,$$

what finishes the proof of Claim 7. □

Claim 7 states that the event E will occur during the *Leak* experiment (see Def. 8) with probability not greater than $|C| \cdot (4q(p))^n$. Claim 6 states that unless E occurs the leakage from circuit $\widehat{C}$ can be simulated perfectly by feeding *any* input to the circuit. Therefore, based on the two claims, for a given input X, the output of the experiment $\mathit{Leak}(C, X, p)$ can be simulated up to $(4q(p))^n$ in statistical distance just by feeding $\widehat{C}$ with any input and leaking each wire with probability p. This finishes the proof of the theorem. □

6.1 Concrete results

In this section we present the concrete results implied by Theorem 1. These are immediate consequences of the theorem. For affine circuits we obtain the following.

Proposition 1. *Assume that a circuit C is an affine circuit. Our transformation $\widehat{C}$, as described in Section 4.1, is $(p, |C| \cdot (4p + 8\sqrt{3p})^n)$-private for any probability p.*

Proof. As stated in the Section 4.3, for every gadget used in $\widehat{C}$ its projection probability function is upper-bounded by $3p$ and hence by $p + 2\sqrt{3p}$. Thus, the Proposition is a consequence of the Theorem 1 for the function $q(p) = p + 2\sqrt{3p}$. □

For the general circuits we have the following.

Proposition 2. *Assume that a circuit C is an arithmetic circuit. Our transformation $\widehat{C}$, as described in Section 4.1, is $(p, |C| \cdot (32np + 4n\sqrt{3p})^n)$-private for any probability p.*

Proof. From the Section 4.3, and the Lemma 3 on ISW multiplication gadget in particular, we conclude that for every gadget used in $\widehat{C}$ its projection probability function is upper-bounded by $n(8p + \sqrt{3p})$. Assuming $n \geq 2$, this function also upper-bounds $p + 2\sqrt{3p}$. Thus, the Proposition is a consequence of the Theorem 1 for the function $q(p) = n(8p + \sqrt{3p})$. □

Finally, let us state the result for the multi-round refreshing circuits.

Proposition 3. *Consider a k-round refreshing circuit (see Sect. 2). This circuit is $(p, k \cdot (4p + 8\sqrt{3p})^n)$-private for any probability p.*

Proof. As stated in the Section 4.3, the projection probability function of the identity gadgets $\widehat{ID}$ used in the circuit equals p and hence is upper-bounded by $p + 2\sqrt{3p}$. Thus, the Proposition is a consequence of the Theorem 1 for the function $q(p) = p + 2\sqrt{3p}$. □

6.2 Open problems

We propose two interesting research directions.

One question that remains open is if the ISW multiplication gadget can be replaced by some a gadget with better parameters, i.e. gadget implementing the multiplication function $g(x, y) = x \cdot y$ with a smaller projection probability function than $f(p) = n(8p + \sqrt{3p})$ (see Lemma 3). Ideally, the function f would not depend on the security parameter n. Using such gadget in the construction would improve the result of Proposition 2.

Second line of research poses the question if we can combine the simple refreshing with masked multiplications that are secure for constant p, e.g., the schemes from [1, 3].

Chapter 7

Conclusion

In this work we introduce a new method to analyze the security of masking schemes in the noisy leakage model of Prouff and Rivain [34]. Our approach enables us to show the security of a simple refreshing scheme which is optimal in terms of randomness complexity (it requires only $n - 1$ random values), and uses a small number of arithmetic operations. Our results are achieved by introducing a new technique for analyzing masked circuits against noisy leakages, which is of independent interest.

We believe that our results are of practical importance to the analysis of side-channel resistant masking schemes. The reason for this are twofold. First, our refreshing scheme is very simple and efficient, and reduces the overheads of the masking countermeasure significantly – in particular, for certain types of computation. For example in the case of a secure key update mechanism as used in any cryptographic scheme, we can reduce randomness and circuit complexity from $O(n^2)$ using ISW-like refreshing to $O(n)$, where the asymptotic in the latter is with nearly optimal constants. Second, while in [7] it was shown how to construct a very simple refreshing scheme (similar to the one used in our work), the security analysis was in a more restricted model (the bounded moment model), and carried out only for small n. In our case, the analysis works for any n and in the standard noisy model that is considered generally to accurately model physical side-channel leakage, and hence our result implies that the simple refreshing can securely replace more complex and expensive schemes in practice.

Interesting questions for future research include to extend our analysis to other masking schemes [4], to explore the tightness of our bounds and to verify our results experimentally in practice (e.g., by providing simulations

on the practical resistance of the countermeasure and its efficiency).

Bibliography

[1] Miklós Ajtai. Secure computation with information leaking to an adversary. In Lance Fortnow and Salil P. Vadhan, editors, *43rd Annual ACM Symposium on Theory of Computing*, pages 715–724. ACM Press, June 2011.

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

[3] 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. Springer, Heidelberg, May 2016.

[4] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, and Ingrid Verbauwhede. Theory and practice of a leakage resilient masking scheme. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 758–775. Springer, Heidelberg, December 2012.

[5] 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. Springer, Heidelberg, April 2015.

[6] 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 16: 23rd Conference on Computer and Communications Security*, pages 116–129. ACM Press, October 2016.

[7] 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. Springer, Heidelberg, April / May 2017.

[8] Sonia Belaïd, Dahmun Goudarzi, and Matthieu Rivain. Tight private circuits: Achieving probing security with the least refreshing. In *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part II*, pages 343–372, 2018.

[9] 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. Springer, Heidelberg, August 1999.

[10] Jean-Sébastien Coron. Formal verification of side-channel countermeasures via elementary circuit transformations. In *Applied Cryptography and Network Security - 16th International Conference, ACNS 2018, Leuven, Belgium, July 2-4, 2018, Proceedings*, pages 65–82, 2018.

[11] 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. Springer, Heidelberg, March 2014.

[12] Giovanni Di Crescenzo, Richard J. Lipton, and Shabsi Walfish. Perfectly secure password protocols in the bounded retrieval model. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 225–244. Springer, Heidelberg, March 2006.

[13] 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. Springer, Heidelberg, May 2014.

[14] Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete - or how to evaluate the security of any leaking device. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 401–429. Springer, Heidelberg, April 2015.

[15] Stefan Dziembowski. Intrusion-resilience via the bounded-storage model. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 207–224. Springer, Heidelberg, March 2006.

[16] Stefan Dziembowski and Sebastian Faust. Leakage-resilient cryptography from the inner-product extractor. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 702–721. Springer, Heidelberg, December 2011.

[17] Stefan Dziembowski, Sebastian Faust, and Maciej Skorski. Noisy leakage revisited. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 159–188. Springer, Heidelberg, April 2015.

[18] Stefan Dziembowski, Sebastian Faust, and Karol Zebrowski. Simple refreshing in the noisy leakage model. Lecture Notes in Computer Science, pages 315–344. Springer, Heidelberg, December 2019.

[19] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *49th Annual Symposium on Foundations of Computer Science*, pages 293–302. IEEE Computer Society Press, October 2008.

[20] Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting circuits from leakage: the computationally-bounded and noisy cases. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 135–156. Springer, Heidelberg, May / June 2010.

[21] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic analysis: Concrete results. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 251–261. Springer, Heidelberg, May 2001.

[22] Shafi Goldwasser and Guy N. Rothblum. Securing computation against continuous leakage. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 59–79. Springer, Heidelberg, August 2010.

[23] Dahmun Goudarzi, Antoine Joux, and Matthieu Rivain. How to securely compute with noisy leakage in quasilinear complexity. In *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part II*, pages 547–574, 2018.

[24] Dahmun Goudarzi, Ange Martinelli, Alain Passelègue, and Thomas Prest. Unifying leakage models on a rényi day. Cryptology ePrint Archive, Report 2019/138, 2019. `https://eprint.iacr.org/2019/138`.

[25] 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. Springer, Heidelberg, August 2003.

[26] Ali Juma and Yevgeniy Vahlis. Protecting cryptographic keys against continual leakage. In Tal Rabin, editor, *Advances in Cryptology –*

CRYPTO 2010, volume 6223 of *Lecture Notes in Computer Science*, pages 41–58. Springer, Heidelberg, August 2010.

[27] Yael Tauman Kalai and Leonid Reyzin. A survey of leakage-resilient cryptography. *IACR Cryptology ePrint Archive*, 2019:302, 2019.

[28] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. Spectre attacks: Exploiting speculative execution. In *2019 IEEE Symposium on Security and Privacy*, pages 1–19. IEEE Computer Society Press, 2019.

[29] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, Heidelberg, August 1996.

[30] 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. Springer, Heidelberg, August 1999.

[31] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown: Reading kernel memory from user space. pages 973–990, 2018.

[32] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold implementations against side-channel attacks and glitches. In *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*, pages 529–545, 2006.

[33] Krzysztof Pietrzak. A leakage-resilient mode of operation. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 462–482. Springer, Heidelberg, April 2009.

[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. Springer, Heidelberg, May 2013.

[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. Springer, Heidelberg, August 2010.

[36] François-Xavier Standaert. *Introduction to Side-Channel Attacks*, pages 27–42. Springer US, Boston, MA, 2010.

[37] Yu Yu, François-Xavier Standaert, Olivier Pereira, and Moti Yung. Practical leakage-resilient pseudorandom generators. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10: 17th Conference on Computer and Communications Security*, pages 141–151. ACM Press, October 2010.