A Block Cipher based PRNG Secure Against Side-Channel Key Recovery

Christophe Petit [1,⋆] , François-Xavier Standaert [1,†],
Olivier Pereira [1,◇], Tal G. Malkin [2], Moti Yung [2,3].

[1] UCL Crypto Group, Université catholique de Louvain.
[2] Dept. of Computer Science, Columbia University., [3] Google Inc.

e-mails: christophe.petit,fstandae,olivier.pereira@uclouvain.be;
tal,moti@cs.columbia.edu

Abstract. We study the security of a block cipher-based pseudorandom number generator, both in the black box world and in the physical world, separately. We first show that the construction is a secure PRNG in the ideal cipher model. Then, we demonstrate its security against a Bayesian side-channel key recovery adversary. As a main result, we show that our construction guarantees that the success rate of the adversary does not increase with the number of physical observations, but in a limited and controlled way. Besides, we observe that, under common assumptions on side-channel attack strategies, increasing the security parameter (typically the block cipher key size) by a polynomial factor involves an increase of a side-channel attack complexity by an exponential factor, making the probability of a successful attack negligible. We believe this work provides a first interesting example of the way the algorithmic design of a cryptographic scheme influences its side-channel resistance.

1 Introduction

Side-channel attacks are a powerful cryptanalytic technique that exploits data-dependent physical leakages (e.g. power consumption or electromagnetic radiation) in order to recover secret data from actual implementations. Following their demonstration in the late 1990s, a number of countermeasures have been proposed to increase the security of cryptographic devices. For example, several proposals attempt to reduce the amount of information provided by any single query to a target device, including noise addition [13], masking [8] or hiding [18]. In this paper, we adopt a different approach in which we do *not* try to affect single query leakages. Assuming that actual side-channel attacks require to combine several queries to reach high success rates, we rather try to make the efficient combination of the leakages difficult. Therefore, the approach we propose here can (and sometimes has to) be efficiently combined with other countermeasures. In contrast with most ad hoc solutions to prevent side-channel attacks, we include our security analysis within a theoretical framework introduced in [16] and

⋆ Research Fellow of the Belgian Fund for Scientific Research (F.R.S.-FNRS). †Postdoctoral researcher of the Belgian Fund for Scientific Research. ◇Research associate of the Belgian Fund for Scientific Research.

provide generic evaluations for the success rate of a side-channel key-recovery adversary. But since the actual security of an implementation can only be shown for practical instances of leakage functions, we also demonstrate exemplary contexts in which our construction provides security, from the frequently considered Hamming weight leakage function to the powerful identity leakage function.

As a case-study, we investigate the design of a pseudorandom generator (PRNG) based on block ciphers. We believe this example is interesting on its own, as PRNGs are standard components in many common applications, including authentication in low-power devices, or re-keying for block ciphers. The construction of a side-channel resistant PRNG was also considered in the "physically observable cryptography" model of Micali and Reyzin [14]. Our study differs from that one by several important aspects. The most important one being that our analysis is based on what Micali and Reyzin call a "specialized model": our model of side-channel leakages and adversarial power is influenced by the experience gained in the practice of side-channel attacks. As a result of this specialization, we expect our construction to be more efficient. We use two chained block ciphers, and our PRNG outputs a number of bits equal to the block size after each round (rather than one bit per round, based on any one-way permutation as in the Blum-Micali construction used by Micali and Reyzin). Our PRNG construction is inspired by well-known re-keying techniques [1] and protocol-based strategies to withstand side-channel attacks [12]. But contrary to the simple arguments given in [11], we use our case-study to illustrate the strong dependency between a leakage function, the structure of an algorithm and the combination of side-channel observations. In addition we turn our theoretical analysis into quantitative metrics to evaluate the security of any implementation of our PRNG.

For these purposes, and as a first step towards the provable security against side-channel attacks, we clearly separate black box and physical security issues. In a first part of the paper, we consider our PRNG construction in the classical cryptographic setting and demonstrate that under the assumption that its component ciphers are ideal, it is a secure PRNG. Then, we investigate a generic implementation of our primitive. We demonstrate that for exemplary but meaningful leakage functions, increasing the number of round observations of the target device does not increase the success rate of a side-channel key recovery adversary but in a limited and controlled way. An interesting consequence of this observation is that, under certain reasonable assumptions about the computational limits and strategy of the side-channel adversary, increasing the ciphers key size by a polynomial factor involves an increase of the side-channel attack complexity by an exponential factor. In addition, our evaluations relate to the amount of randomness (i.e. noise) in the side-channel observations that can consequently be used as an alternative security parameter, since the noise is a typical countermeasure to affect single PRNG round leakages. As a matter of fact, the combination of these results does not imply that our construction is a secure PRNG in the physical world but that (independently): (i) it is a secure PRNG

in the black box world, and (ii) side-channel key recovery against its implementation is hard. We leave the combination of black box and physical security notions in a unified way as an important scope for further research.

Roadmap: This paper is organized as follows. In Section 2, we give an overview of our PRNG construction. In Section 3, we state standard security definitions, and use them to show the security of our PRNG in a black box world. In Sections 4 and 5, we turn to the physical world and investigate the resistance of our PRNG to recovery of its seed, by considering side-channel attacks. Section 6 considers different leakage functions and shows the security of our construction in these specific contexts. Section 7 gives further insights on the practical security impact of our construction strategy. Eventually, Sections 8 and 9 relax certain assumptions used in our analysis and conclusions are in Section 10.

2 Description of the PRNG

The PRNG construction is illustrated in Figure 1. It is a serial combination of two instances of a block cipher, denoted by E_1 and E_2, placed into the Cipher Block Chaining encryption mode. The input of the first block cipher is initialized to a public IV, and each block cipher is initialized with its own master key, denoted k and k^* respectively, these keys playing the role of seed for the PRNG.

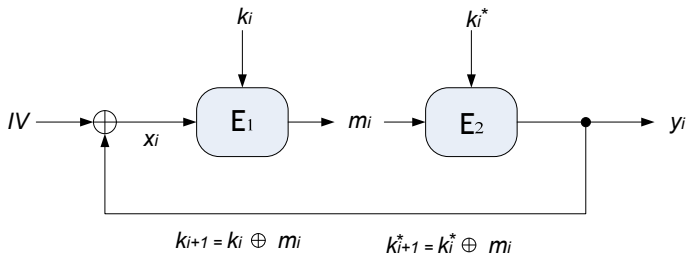

Fig. 1. Block cipher based PRNG.

The execution of one round of our block cipher is as follows: given the input x_i of the first block cipher, and the current value of the keys k_i and k_i^* used by the two block ciphers, an intermediate value m_i is computed as $\mathsf{E}_{k_i}(x_i)$. Then the output of the PRNG is computed as $y_i = \mathsf{E}_{k_i^*}(m_i)$, the keys to be used by the block ciphers in the next round as $k_{i+1} = k_i \oplus m_i$ and $k_{i+1}^* = k_i^* \oplus m_i$, and the new input for the first block cipher as $x_{i+1} = IV \oplus y_i$. In the following, we refer to k, k^* as the master keys and to k_i, k_i^* as the running keys. The construction is generic in the sense that its input/output/key bit sizes are not specified (but identical): they depend on the actual block cipher chosen to instantiate the PRNG. The design of this scheme is based upon the following two principles:

1. If E_1 and E_2 are "good" in the black box world, then so should the PRNG be.
2. Each running key k_i, k_i^* is used to encrypt *only* one message. For this purposes, we assume a *fixed* public IV (i.e. it cannot be selected by the PRNG user). A way to relax this assumption and to initialize the PRNG with a public seed is discussed in Section 8.

The goal of the second principle is to make it computationally difficult to combine the leaked information from different encryption steps. In order to respect that principle, the running keys are updated after each PRNG round.

3 Black-box Security of our PRNG

This section studies the security of our PRNG in an ideal world, where the only interface between the adversary and the PRNG occurs through the PRNG output. We show that, in the ideal cipher model proposed by Shannon [15], an adversary has a negligible probability to distinguish the output of our PRNG from a random sequence of bits. We first describe the ideal cipher model and define the security properties we expect from pseudorandom generators, then discuss the security of our construction.

3.1 Security Notions

Block cipher security. The ideal cipher model has been used in many works, including [4]. It assumes that block ciphers are random families of permutations. That is, they consist of random permutations chosen independently for each possible key. More precisely, suppose $\mathcal{K}$ and $\mathcal{M}$ are sets. An ideal block cipher is a map $\mathsf{E} : \mathcal{K} \times \mathcal{M}$ where, for each key $k \in \mathcal{K}$, the function $\mathsf{E}_k(\cdot) = \mathsf{E}(k, \cdot)$ is a random permutation on the message set $\mathcal{M}$ (independent of any other permutation). If E is an ideal block cipher, then E^{-1} is its inverse and $\mathsf{E}_k^{-1}(y)$ is the string x such that $\mathsf{E}_k(x) = y$. In the rest of this paper, we assume $\mathcal{K} = \mathcal{M}$: the messages and keys used by our block ciphers belong to the same set.

Pseudorandom generator security. A pseudorandom generator is a deterministic algorithm G that maps elements of a domain $\mathcal{K}$ on elements of a larger domain $\hat{\mathcal{K}}$ with the property that it is hard to distinguish the uniform distribution on $\hat{\mathcal{K}}$ from the distribution on $\hat{\mathcal{K}}$ defined as the image through G of the uniform distribution on $\mathcal{K}$. This hardness is measured through the notion of prng-advantage of adversaries, that we define as follows, after [19].

Definition 1. Let $\mathsf{G} : \mathcal{K} \to \hat{\mathcal{K}}$ be a pseudorandom generator, and let A be an algorithm that takes an element of $\hat{\mathcal{K}}$ as input and returns a bit. Consider:

$$\mathbf{Succ}_{\mathsf{G},\mathsf{A}}^{\mathrm{prng}-1} = \Pr[\mathsf{A}(\hat{k}) = 1 : \hat{k} \xleftarrow{R} \hat{\mathcal{K}}],$$
$$\mathbf{Succ}_{\mathsf{G},\mathsf{A}}^{\mathrm{prng}-0} = \Pr[\mathsf{A}(\hat{k}) = 1 : \hat{k} \leftarrow \mathsf{G}(k); k \xleftarrow{R} \mathcal{K}],$$

where $x \xleftarrow{R} \mathcal{X}$ denotes the selection of an element x of the set $\mathcal{X}$ according to the uniform distribution. The *prng-advantage* of A against G is defined as:

$$\mathbf{Adv}^{\mathrm{prng}}_{\mathsf{G},\mathsf{A}} = |\mathbf{Succ}^{\mathrm{prng}-1}_{\mathsf{G},\mathsf{A}} - \mathbf{Succ}^{\mathrm{prng}-0}_{\mathsf{G},\mathsf{A}}|.$$

We say that G is secure if the prng-advantage of any polynomial time adversary is small in the by now traditional complexity theoretic sense.

3.2 Security of our PRNG

We now justify the security of the PRNG of Section 2 by relating its black box security to the black box security of its underlying ciphers.

Claim 1. *Under the assumption that its component block ciphers are ideal ciphers, the PRNG of section 2 is secure.*

Security of a single round. We first consider the security of any single round of our PRNG. For this purpose, we consider the family of PRNGs $\overline{\mathsf{G}} = \{\mathsf{G}_X\}_{X \in \mathcal{K}}$, where each $\mathsf{G}_X : \mathcal{K} \times \mathcal{K} \to \mathcal{K} \times \mathcal{K} \times \mathcal{K}$ is defined as follows:

$$\mathsf{G}_X(K, K^*) = (\mathsf{E}_K(X) \oplus K, \mathsf{E}_K(X) \oplus K^*, \mathsf{E}_{K^*}(\mathsf{E}_K(X))).$$

The index X represents the value used as input for the first block cipher, the first two parts of the output represent the keys that will be used in the next round, and the last part of the output represents the visible output of the round. Fix now any adversary A against $\mathsf{G}_X \in \overline{\mathsf{G}}$, and consider the probability of success:

$$\mathbf{Succ}^{\mathrm{prng}-0}_{\mathsf{G}_X,\mathsf{A}} = \Pr[\mathsf{A}(\hat{k}) = 1 : \hat{k} \leftarrow \mathsf{G}_X(k, k^*); (k, k^*) \xleftarrow{R} \mathcal{K} \times \mathcal{K}]$$

Unwinding the definition of G_X, It can be rewritten as:

$$\begin{aligned}\mathbf{Succ}^{\mathrm{prng}-0}_{\mathsf{G}_X,\mathsf{A}} = \Pr[&\mathsf{A}(k_1, k_1^*, y) = 1 : k \xleftarrow{R} \mathcal{K}; k^* \xleftarrow{R} \mathcal{K};\\ &m \leftarrow \mathsf{E}_k(X); k_1 \leftarrow m \oplus k; k_1^* \leftarrow m \oplus k^*; y \leftarrow \mathsf{E}_{k^*}(m)].\end{aligned}$$

We first observe that, in the ideal cipher model, the random selection of a key k followed by the use of the permutation E_k is equivalent to the use of a randomly selected permutation P. Therefore, we have that:

$$\begin{aligned}\mathbf{Succ}^{\mathrm{prng}-0}_{\mathsf{G}_X,\mathsf{A}} = \Pr[&\mathsf{A}(k_1, k_1^*, y) = 1 : k \xleftarrow{R} \mathcal{K}; k^* \xleftarrow{R} \mathcal{K};\\ &\mathsf{P} \xleftarrow{R} Perm(\mathcal{K}); \mathsf{P}^* \xleftarrow{R} Perm(\mathcal{K}); m \leftarrow \mathsf{P}(X);\\ &k_1 \leftarrow m \oplus k; k_1^* \leftarrow m \oplus k^*; y \leftarrow \mathsf{P}^*(m)]\end{aligned}$$

Since m and y are computed by applying random permutations on elements of $\mathcal{K}$, they cannot be distinguished from random elements of $\mathcal{K}$. Therefore:

$$\begin{aligned}\mathbf{Succ}^{\mathrm{prng}-0}_{\mathsf{G}_X,\mathsf{A}} = \Pr[&\mathsf{A}(k_1, k_1^*, y) = 1 : k \xleftarrow{R} \mathcal{K}; k^* \xleftarrow{R} \mathcal{K};\\ &m \xleftarrow{R} \mathcal{K}; k_1 \leftarrow m \oplus k; k_1^* \leftarrow m \oplus k^*; y \leftarrow \mathcal{K}]\end{aligned}$$

Eventually, we observe that k_1 and k_1^* are computed as the XOR of independent uniformly chosen values. As a result, these variables cannot be distinguished from independent and uniformly distributed values by the adversary:

$$\mathbf{Succ}^{\mathrm{prng}-0}_{\mathsf{G}_X,\mathsf{A}} = \Pr[\mathsf{A}(k_1, k_1^*, y) = 1 : k_1 \xleftarrow{R} \mathcal{K}; k_1^* \xleftarrow{R} \mathcal{K}; y \xleftarrow{R} \mathcal{K}]$$

But this last probability is also equal to $\mathbf{Succ}^{\mathrm{prng}-1}_{\mathsf{G}_X,\mathsf{A}}$, which shows that, if the block cipher adopted in practice essentially behaves like an ideal cipher, no reasonable adversary can have an important prng-advantage against G_X.

Security of multiple rounds. The behavior of multiple rounds of our PRNG can be seen as the sequential execution of PRNGs taken from $\overline{\mathsf{G}}$ family, as depicted in Figure 2: at each step, we select the PRNG indexed by the XOR of the IV and the last part of the output of the previous step, and use the first two parts of the output of the previous step as seed. Using a standard hybrid argument, e.g. following [7, Theorem 3.3.3], we obtain that the prng-advantage of an adversary against n-rounds of our PRNG is bounded by n times the prng-advantage of a similar adversary against a single instance of any PRNG in $\overline{\mathsf{G}}$. In the rest of this paper, we denote a q-round version of our PRNG as G_q, and omit the superscript q when it is clear from the context.

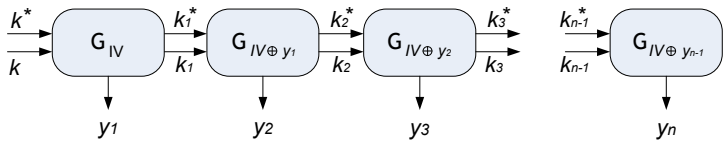

Fig. 2. Multiple rounds of our PRNG.

4 Physical security model

This section considers the physical security of the previously described PRNG with respect to the notion of side-channel key recovery. We first outline our model and definitions. Then, we detail our assumptions on the physical implementation of the construction, based on standard practice in the side-channel community. In the remaining of this paper, we mostly follow the formalism introduced in [14, 16]; we refer to these papers for definitions of a physical computer, leakage function, and more details and justifications of the model.

4.1 Definition of security

When we move to the physical world, a q-round version of our PRNG $\mathsf{G}_q(K, K^*)$ with seed (K, K^*) is associated with a leakage function $\mathsf{L}_q(K, K^*)$ that describes what can be measured during an actual execution of $\mathsf{G}_q(K, K^*)$ on a specific

physical device[1]. In Sections 5 and 6, we will consider different types of leakage functions $\mathsf{L}_q(K, K^*)$. Following [14], the pair $\mathsf{P}_q(K, K^*) = (\mathsf{G}_q(K, K^*), \mathsf{L}_q(K, K^*))$ constitutes a physical implementation of our PRNG. We want to analyze the security of a physical implementation $\mathsf{P}_q(K, K^*)$ of our PRNG in front of a side-channel key recovery adversary. The goal of such an adversary A is to guess a specific function δ of a master key K, K^* used during the (physical) execution of $\mathsf{P}_q(K, K^*)$. The success rate of A is defined as:

$$\mathbf{Succ}^{\mathrm{sc-kr}-\delta(K,K^*)}_{\mathsf{P}^q(K,K^*),\mathsf{A}} = \Pr[\mathsf{A}(\mathsf{P}_q(k, k^*)) = \delta(k, k^*) : k \xleftarrow{R} \mathcal{K}; k^* \xleftarrow{R} \mathcal{K}]$$

$\delta(K, K^*)$ is traditionally seen as a key classification function, which typically returns one or two key bytes of a running key K_i, K_i^* targeted during the side-channel attack. In the following, we first do the reasonable assumption that the side-channel adversary is bounded. It means that the set $\delta(\mathcal{K}, \mathcal{K})$ has a limited size of typically 2^8, 2^{16}, 2^{32}. Additionally, we consider a standard strategy where physical key parts are targeted independently, i.e. we consider one particular type of classification function δ. The global success rate will then be the product of the success rates on each of the targeted key parts. For instance, assuming identical success rates for all pieces of the key, a successful attack against the full n-bit key requires $n/8$ partial attacks against 8-bit classes:

$$\mathbf{Succ}^{\mathrm{sc-kr}-K}_{\mathsf{P}^q(K,K^*),\mathsf{A}} = (\mathbf{Succ}^{\mathrm{sc-kr}-K_{[0\dots7]}}_{\mathsf{P}^q(K,K^*),\mathsf{A}})^{n/8} \tag{1}$$

As a result, if we can obtain a construction where the success rate on some part of the key is bounded, the global success rate will decrease exponentially with the length of the key. We mention that assuming identical success rates for all pieces of the key may not always be correct. For example, one could imagine a leakage function providing the most significant bit(s) of a key to the adversary which straightforwardly contradicts the assumption. However for the practically meaningful leakage functions that we consider in the next sections, it is expected to hold in a sufficient degree. Similarly, the decision to target physical parts of the key (of 8, 16, 32, ... bits) rather than bits of information is influenced by the practice of side-channel attacks in which the leakage is usually correlated with the values of physical bits. We consequently use this strategy as a reasonable starting point allowing the analysis of our construction.

4.2 Circuit model & working assumptions

Our model for the physical implementation of the PRNG is pictured in Figure 3. We now detail the working assumptions required for its security analysis. Note that a significant part of these assumptions were selected in order to facilitate the formal investigation of our construction and will be relaxed in the following.

[1] In [14], the leakage function was defined as a function of a device's internal configuration, a measurement parameter and a random parameter. For simplicity, our notation considers the measurement parameter as fixed and takes as only input the part of the device internal configuration that is targeted in the attacks, namely the master keys K, K^*. The noise parameter will be explicitly mentioned when required.

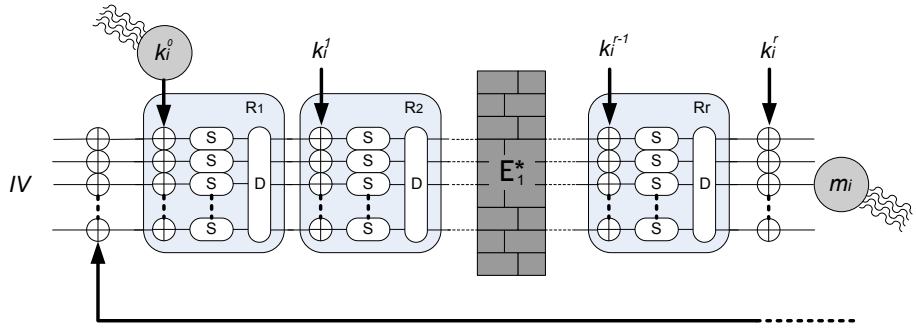

Fig. 3. Physical implementation of the PRNG.

a1. We assume a fixed (meaning read-only) IV in order to avoid the possibility of chosen IV attacks exploiting multiple initializations.
a2. We consider an iterative block cipher with r identical rounds: $\mathsf{R}_1, \mathsf{R}_2, \ldots \mathsf{R}_r$. Each round is made of different operations, e.g. bitwise XORs, s-boxes and diffusion layers in our picture. A typical example is the AES Rijndael [6].
a3. We do not consider the key scheduling algorithm and assume that the cipher initially has $r+1$ independent round keys k_0^j, $j \in [0, r]$, each belonging to $\mathcal{K}$, updated according to the same procedure: $k_{i+1}^j = k_i^j \oplus m_i$.
a4. We only consider the leakage obtained when observing the execution of the first block cipher E_1 in the PRNG.
a5. We consider an adversary targeting the first round key[2] k_i^0. From this point we omit the "0" superscript for all keys, as we will always consider the first round key. We also consider an adversary targeting this first round key for either the first or last PRNG iteration considered in the attack, i.e. k_0 or k_q.
a6. During each iteration of the PRNG, the adversary obtains two leakages l_{K_i} and l_{M_i}. As a matter of fact, this does *not* mean that the adversary is limited in the way he exploits the side-channel information. It just means that all the information obtained from the execution of the rounds is translated into information on these two values. The leakage function abstraction captures the fact that "any kind of information" can in principle be obtained.
a7. Finally and most importantly, we assume that the information on a running key k_i and the information on the middle point m_i cannot be efficiently combined, but through the key update procedure $k_{i+1} = k_i \oplus m_i$. That is, we assume that the cipher E_1's inner rounds constitute a permutation E_1^* that is hard to compute/invert for the adversary.

Among these assumptions, the first one is the most critical from an application point of view. A way to mitigate it by initializing the PRNG securely with public random seeds is discussed in Section 8. Assumptions 3, 4, 5 reduce the amount of information leakage provided to the adversary and are relaxed in Section 9. We now use these definitions and assumptions in an analysis of our PRNG.

[2] Since all rounds are identical and the IV is known, we assume that it is the easiest target, i.e., if k_i^0 cannot be recovered, the other round keys cannot either.

5 Security analysis of a Bayesian side-channel adversary

The objective of the following analysis is to evaluate the physical security of our PRNG. According to [16], such an evaluation generally requires to consider both the amount of information leaked by an implementation and the extent to which an actual adversary can turn this information into a successful key recovery. However, as explained in the introduction of this paper, our PRNG does not intend to affect the amount of information that is provided to the adversary during a single round. In fact, the information available in the physical observations is a parameter of our analysis, hidden in the leakage function abstraction L. By contrast, the PRNG attempts to make the efficient combination of this information a difficult task. How difficult is the leakage combination can consequently be measured with a security metric, e.g. the key-recovery success rate of a side-channel adversary. For this purpose, we now consider the Bayesian side-channel adversary which is the most powerful one from an information theoretic point of view [5], when a perfect knowledge of the noise distribution is available.

More specifically, we model an adversary that is provided with generic leakages under the form of a vector of $2q-1$ components, each of them corresponding to the leakage that can be measured during the use of the i-th round key or of the i-th round value of the middle point m_i. Following [16], we then consider that the random leakage variable obtained from L_q can be expressed as a random vector $\mathbf{L_q}$ of the form $(\mathsf{L}_K(K_0), \mathsf{L}_M(M_0), \mathsf{L}_K(K_1), \mathsf{L}_M(M_1), \ldots, \mathsf{L}_K(K_q))$,[3] in which each $\mathsf{L}_K(K_i)$, $\mathsf{L}_M(M_i)$ is a random variable representing the leakage trace on the use of, respectively, the running key K_i and the middle point M_i. We also write $\mathbf{l_q} = (l_{K_0}, l_{M_0}, l_{K_1}, \ldots, l_{K_q})$ to denote any fixed element in the domain of $\mathbf{L_q}$. Given this specific form of the leakages, a Bayesian adversary observing a leakage $\mathbf{l_q}$ selects the key candidate $K_{i,guess}$ such that $K_{i,guess} := \arg\max_{k_i} \Pr[K_i = k_i | \mathbf{L_q} = \mathbf{l_q}]$. Using the fact that all round keys K_i can be considered as independent and uniformly distributed (following our black box analysis of Section 3), this is equivalent to choosing $K_{i,guess} = \arg\max_{k_i} \Pr[\mathbf{L_q} = \mathbf{l_q} | K_i = k_i]$.

We turn now to the generic evaluation of this expression. In order to simplify our analysis, we first evaluate the probability $\Pr[\mathbf{L_q} = \mathbf{l_q} | K_i = k_i]$ in the context of deterministic leakage functions $\mathsf{L} = \mathsf{L}_{det}$, where the $\mathsf{L}_K(\cdot)$ and $\mathsf{L}_M(\cdot)$ functions are deterministic. Then we extend our analysis to noisy leakage functions of the form $\mathsf{L} = \mathsf{L}_{det} + R$, where R is noise occurring on each leakage component according to a certain noise distribution. That is, the $\mathsf{L}_K(\cdot)$ and $\mathsf{L}_M(\cdot)$ functions are evaluated as the sum of a deterministic function and a random variable selected according to the noise distribution.

[3] The leakage corresponding to M_q is not taken into account in this evaluation, as it is only useful to attack the $q+1$-th round of the PRNG .

5.1 Analysis of deterministic leakages

For each possible values l_{K_i}, l_{M_i} of the deterministic leakage functions $\mathsf{L}_K(K_i)$, $\mathsf{L}_M(M_i)$, let us define a running matrix $A^{l_{K_i}}$ and an update matrix $B^{l_{M_i}}$ as:

$$A^{l_{K_i}}(k_i, k_i') = \begin{cases} 1, \text{ if } k_i = k_i' \text{ and } \mathsf{L}_K(k_i) = l_{K_i}; \\ 0, \text{ otherwise;} \end{cases}$$

$$B^{l_{M_i}}(k_i, k_{i+1}) = \begin{cases} 1, \text{ if } \mathsf{L}_M(k_i \oplus k_{i+1}) = l_{M_i}; \\ 0, \text{ otherwise.} \end{cases}$$

The matrix $A^{l_{K_i}}$ is a diagonal matrix, with one row (resp. column) for each possible key in $\mathcal{K}$, and where elements are set to 1 iff the leakage corresponding to the key indexed by the current line is equal to leakage corresponding to K_i. Similarly, the matrix $B^{l_{M_i}}$ has elements equal to 1 in position (k_i, k_{i+1}) iff the leakage corresponding to $\mathsf{L}_M(k_i \oplus k_{i+1})$ is equal to l_{M_i}. It directly follows from the definitions that $\sum_{l_{K_i}} A^{l_{K_i}} = I_{2^n}$ (identity matrix) and $\sum_{l_{M_i}} B^{l_{M_i}} = \mathbf{1}$ (all ones matrix). Then, for each possible leakage value $\mathbf{l_q}$ of $\mathbf{L_q}$, we define a leakage (directed) graph $G^{\mathbf{l_q}} = (V^{\mathbf{l_q}}, E^{\mathbf{l_q}})$ as follows. The set of vertices $V^{\mathbf{l_q}}$ contains $(2q+2)|\mathcal{K}|$ nodes, referred to by pairs of the form (k, i) where $k \in \mathcal{K}$ and $0 \leq i \leq 2q+1$. The set $E^{\mathbf{l_q}}$ is defined by the edges:

1. $((k, 2i), (k', 2i+1)) \in E^{\mathbf{l_q}}$ (where $0 \leq i \leq q$), iff $A^{l_{K_i}}(k, k') = 1$,
2. $((k, 2i+1), (k', 2i+2)) \in E^{\mathbf{l_q}}$ (where $0 \leq i \leq q-1$), iff $B^{l_{M_i}}(k, k') = 1$.

An example of such a graph is in Figure 4.

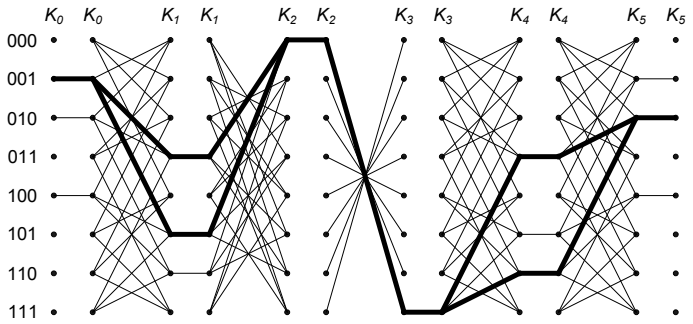

Fig. 4. Leakage graph for n=3, with a Hamming weight leakage function providing a vector $\mathbf{l}_5$=$\{1, 1, 2, 2, 0, 3, 3, 1, 2, 1, 1\}$. The bold edges show the 4 elements of $S_5(001, \mathbf{l}_5)$.

Finally, for every key k_0 and leakage vector $\mathbf{l_q}$, we define the set of keys that possibly gave rise to the leakages:

$$S_q(k_0, \mathbf{l_q}) = \{(K_1, K_2 ..., K_q) \in \{0, ..., 2^n - 1\}^q |$$
$$\mathsf{L}_M(k_0 \oplus K_1) = l_{M_0}, \mathsf{L}_K(K_1) = l_{K_1}, \mathsf{L}_M(K_1 \oplus K_2)$$
$$= l_{M_1}, \mathsf{L}_K(K_2) = l_{K_2}, \ldots, \mathsf{L}_K(K_q) = l_{K_q}\}$$

It follows that $\Pr[\mathbf{L_q} = \mathbf{l_q} | K_0 = k_0] = |S_q(k_0, \mathbf{l_q})|/2^{nq}$. From the graphical representation of Figure 4, $|S_q(k_0, \mathbf{l_q})|$ can be interpreted as the number of paths from left to right starting at $K_0 = k_0$ in the graph associated with $\mathbf{l_q}$. Similarly, for every k_0, $\mathbf{l_q}$ and every $0 \leq p \leq q$, the number of paths from k_0 to k_p in the graph associated to $\mathbf{l_q}$ equals $n_p(k_p, k_0, \mathbf{l_q}) = |\{(K_1, K_2 ...K_p) \in S_p(k_0, \mathbf{l_p}) | K_p = k_p\}|$ where $\mathbf{l_p}$ corresponds to the first p components of $\mathbf{l_q}$. Looking at the example of Figure 4, we have that $n_3(111, 001, \mathbf{l}_5) = 2$ and $n_5(010, 001, \mathbf{l}_5) = 4$. Define the vector $\mathbf{n_p}(k_0, \mathbf{l_q}) = (n_p(0, k_0, \mathbf{l_q}), n_p(1, k_0, \mathbf{l_q}), \ldots, n_p(2^n - 1, k_0, \mathbf{l_q}))^t$ and define e_{k_0} as a column vector containing all zeros but a one in position k_o. We obtain:

$$|S_q(k_0, \mathbf{l_q})| = \sum_{k_q} n_q(k_q, k_0, \mathbf{l_q}) = (1 \ldots 1)\ \mathbf{n_q}(k_0, \mathbf{l_q})$$
$$\mathbf{n_{p+1}}(k_0, \mathbf{l_q}) = A^{l_{K_{p+1}}} B^{l_{M_p}}\ \mathbf{n_p}(k_0, \mathbf{l_q})$$
$$\mathbf{n_0}(k_0, \mathbf{l_q}) = A^{l_{K_0}} e_{k_0}$$

And by combining the equations above, we express the leakage probabilities as:

$$\Pr[\mathbf{L_q} = \mathbf{l_q} | K_0 = k_0] = \frac{|S_q(k_0, \mathbf{l_q})|}{2^{nq}}$$
$$= \frac{(1...1)\ A^{l_{K_q}} \cdot B^{l_{M_{q-1}}} ... A^{l_{K_1}} \cdot B^{l_{M_0}} \cdot A^{l_{K_0}} \cdot e_{k_0}}{2^{nq}} \tag{2}$$
$$\Pr[\mathbf{L_q} = \mathbf{l_q} | K_q = k_q]$$
$$= \frac{(1...1)\ A^{l_{K_0}} \cdot B^{l_{M_0}} ... A^{l_{K_{q-1}}} \cdot B^{l_{M_{q-1}}} \cdot A^{l_{K_q}} \cdot e_{k_q}}{2^{nq}} \tag{3}$$

5.2 Analysis of noisy leakages

The previous analysis can be easily extended to noisy leakages by defining the leakage vector as a sum of its deterministic part and a random noise variable vector: $\mathbf{L_q} = \mathbf{L_{qdet}} + \mathbf{R}$. It directly yields:

$$\Pr[\mathbf{L_q} = \mathbf{l_q} | K_0 = k_0]$$
$$= \sum_{\mathbf{l_{qdet}}} \Pr[\mathbf{L_q} = \mathbf{l_q} | \mathbf{L_{qdet}} = \mathbf{l_{qdet}}] \Pr[\mathbf{L_{qdet}} = \mathbf{l_{qdet}} | K_0 = k_0]$$
$$= \sum_{\mathbf{l_{qdet}}} \Pr[\mathbf{R} = \mathbf{l_q} - \mathbf{l_{qdet}}] \Pr[\mathbf{L_{qdet}} = \mathbf{l_{qdet}} | K_0 = k_0]$$

If we define the noisy running matrix $C^{l_{K_i}}$ and noisy update matrix $D^{l_{M_i}}$ as the noisy counterparts of $A^{l_{K_i}}$ and $B^{l_{M_i}}$: [4]

$$C^{l_{K_i}} = \sum_{l_{K_i,det}} \Pr[R_{K_i} = l_{K_i} - l_{K_i,det}] \cdot A^{l_{K_i,det}}$$
$$D^{l_{M_i}} = \sum_{l_{M_i,det}} \Pr[R_{M_i} = l_{M_i} - l_{M_i,det}] \cdot B^{l_{M_i,det}}$$

We then find:

$$\Pr[\mathbf{l_q}|k_0] = \frac{(1 \cdots 1) C^{l_{K_q}} D^{l_{M_{q-1}}} \cdots C^{l_{K_1}} D^{l_{M_0}} C^{l_{K_0}}}{2^{nq}},$$

The expression above is similar to Equation (2). The equivalent of Equation (3) can also be derived. Intuitively, $C^{l_{K_i}}(k_i,\ k_i)$ contains the probabilities that a running key candidate k_i gives rise to an actual leakages l_{K_i}, and $D^{l_{M_i}}(k_i, k_{i+1})$ contains the probabilities that any consecutive running key candidates k_i, k_{i+1} give rise to an actual leakages l_{M_i} (just as in the previous section).

5.3 Generic expression for the success rate

From the above probabilities, it is straightforward to derive a generic expression for the success rate of the Bayesian adversary. For a given leakage value $\mathbf{l_q}$, its probability of right guess for K_i is exactly $\max_{k_i} \Pr[K_i = k_i|\mathbf{L_q} = \mathbf{l_q}]$. Therefore the success rate for K_i is given by:

$$\mathbf{Succ}^{\mathrm{sc-kr}-K_i}_{\mathsf{P}^q(K,K^*),\mathsf{A}} = \sum_{\mathbf{l_q}} \Pr[\mathbf{L_q} = \mathbf{l_q}] \cdot \max_{k_i} \Pr[K_i = k_i|\mathbf{L_q} = \mathbf{l_q}]$$

$$= \sum_{\mathbf{l_q}} \Pr[\mathbf{L_q} = \mathbf{l_q}] \cdot \max_{k_i} \frac{\Pr[\mathbf{L_q} = \mathbf{l_q}|K_i = k_i] \Pr[K_i = k_i]}{\Pr[\mathbf{L_q} = \mathbf{l_q}]}$$

$$= \frac{1}{2^n} \sum_{\mathbf{l_q}} \max_{k_i} \Pr[\mathbf{L_q} = \mathbf{l_q}|K_i = k_i]$$

In particular, using the expressions derived above, we get:

$$\mathbf{Succ}^{\mathrm{sc-kr}-K_0}_{\mathsf{P}^q(K,K^*),\mathsf{A}} =$$
$$\frac{1}{2^{n(q+1)}} \sum_{\mathbf{l_q}} ||C^{l_{K_q}} D^{l_{M_{q-1}}} \cdots C^{l_{K_1}} D^{l_{M_0}} C^{l_{K_0}}||_1 \quad (4)$$

$$\mathbf{Succ}^{\mathrm{sc-kr}-K_q}_{\mathsf{P}^q(K,K^*),\mathsf{A}} =$$
$$\frac{1}{2^{n(q+1)}} \sum_{\mathbf{l_q}} ||C^{l_{K_0}} D^{l_{M_0}} \cdots C^{l_{K_{q-1}}} D^{l_{M_{q-1}}} C^{l_{K_q}}||_1 \quad (5)$$

[4] Again with $\int_{l_K} C^{l_K} = I_{2^n}$ and $\int_{l_M} D^{l_M} = \mathbf{1}$.

Looking at these equations, we see that the success rate strongly depends on the leakage function and probability distributions. For most actual leakage functions, analytical evaluation seems difficult (i.e. computationally intensive) when the number of rounds increases. In order to illustrate the validity of our construction in the physical world, the next section consequently details this success rate for certain practically relevant leakage functions.

6 Security analysis of particular leakage functions

We first show a context in which it is possible to derive asymptotical upper bounds on the success rate, which is enough to prove asymptotic security. Then, we consider two leakage functions for which we provide a simulation-based analysis. In particular, we selected:

1. A Hamming weight leakage function such that all the leakages are of the form $\mathsf{L}(k_i) = W_H(k_i)$. In this context, we demonstrate formally that increasing the number of observed rounds does *not* improve the success rate: $\mathbf{Succ}^{\mathrm{sc-kr}-K_0}_{\mathsf{P}^q(K,K^*),\mathsf{A}} = \mathbf{Succ}^{\mathrm{sc-kr}-K_0}_{\mathsf{P}^1(K,K^*),\mathsf{A}}$, for every q.
2. A (so-called) generalized Hamming weight leakage function, such that all the leakages are of the form: $\mathsf{L}(k_i) = W_H(\mathsf{S}(x_{i,[0\cdots7]} \oplus k_{i,[0\cdots7]})) + R$, where S is a known substitution box, e.g., the AES one, and $R \sim \mathcal{N}(\mu, \sigma^2)$ is a Gaussian distributed random noise with mean μ and variance σ^2. We note that for this example, only 8 key bits are targeted (i.e. a typical S-box size).
3. A noisy identity leakage functions such that the leakages are of the form: $\mathsf{L}(k_i) = k_{i,[0\cdots7]} + R$ with $R \sim \mathcal{N}(\mu, \sigma^2)$. It is potentially the most powerful type of leakage function and typically relates to the context of template attacks [5]. If the noise variance is null, rate $\mathbf{Succ}^{\mathrm{sc-kr}-K_0}_{\mathsf{P}^q(K,K^*),\mathsf{A}} = 1$, for all q.

6.1 Hamming weight leakages

In this section, we consider an example of leakage function for which the sums in Equations (4) and (5) can be computed analytically. It yields the statement:

Claim 2. *In the setting of Section 4.2, the success rate of a Bayesian side-channel adversary exploiting a Hamming weight leakage function against our PRNG is independent of the number of PRNG rounds q observed by the side-channel adversary.*

Proof. We assume that all keys are bit strings of length n. For each leakage values $0 \leq l_{K_{i+1}}, l_{M_i} \leq n$, let us define the matrices $Z^{l_{K_{i+1}},l_{M_i}} := A^{l_{K_{i+1}}} B^{l_{M_i}}$. Since Hamming weight leakages are distributed as binomials, these matrices $Z^{l_{K_{i+1}},l_{M_i}}$ have $C_n^{l_{K_{i+1}}}$ non-zeros rows with the same Hamming weight. Moreover, for every $0 \leq l \leq n$, the Hamming weight of each column of Z of which the index has Hamming weight equal to l is the same, which we denote by h_Z^l. We now show by induction that $\Pr[K_q = k_q | \mathbf{L_q} = \mathbf{l_q}] = 1/C_n^{l_{K_q}}$ if $W_H(k_q) = l_{K_q}$ and 0 otherwise.

Equivalently (by Bayes' law), we show that $\Pr[\mathbf{L_q} = \mathbf{l_q}|K_q = k_q] = 2^n \cdot \Pr[\mathbf{L_q} = \mathbf{l_q}]/C_n^{l_{K_q}}$ or 0 depending if $W_H(k_q) = l_{K_q}$ or not. The assertion is trivial for $q = 0$. Using Equation (3), we compute:

$$\begin{aligned}
&\Pr[\mathbf{L_{q+1}} = \mathbf{l_{q+1}}|K_{q+1} = k_{q+1}] \\
&= \frac{1}{2^n} \sum_{k_q} Z^{l_{K_{q+1}}, l_{M_q}}(k_q, k_{q+1}) \cdot \Pr[\mathbf{L_q} = \mathbf{l_q}|K_q = k_q]; \\
&= \begin{cases} \frac{\Pr[\mathbf{L_q}=\mathbf{l_q}]}{C_n^{l_{K_q}}} \cdot h_Z^{l_{K_{q+1}}} & \text{if } W_H(k_{q+1}) = l_{K_{q+1}} \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

Then, we have:

$$\begin{aligned}
\Pr[\mathbf{L_{q+1}} = \mathbf{l_{q+1}}] &= \sum_{k_{q+1}} \Pr[\mathbf{L_{q+1}} = \mathbf{l_{q+1}}|K_{q+1} = k_{q+1}] \\
&= \sum_{k_{q+1}|W_H(k_{q+1})=l_{K_{q+1}}} \frac{h_Z^{l_{K_{q+1}}} \Pr[\mathbf{L_q} = \mathbf{l_q}]}{C_n^{l_{K_q}}}
\end{aligned}$$

$$= \begin{cases} C_n^{l_{K_{q+1}}} \Pr[\mathbf{L_{q+1}} = \mathbf{l_{q+1}}|K_{q+1} = k_{q+1}] & \\ \qquad \text{if } W_H(k_{q+1}) = l_{K_{q+1}}; & \\ 0 \qquad \text{otherwise.} & \end{cases}$$

The success rate can finally be computed as:

$$\begin{aligned}
\mathbf{Succ}^{\mathrm{sc-kr}-K_q}_{\mathsf{P}^q(K,K^*),\mathsf{A}} &= \sum_{\mathbf{l_q}} \Pr[\mathbf{L_q} = \mathbf{l_q}] \cdot \\
&\qquad \max_{k_q} \Pr[K_q = k_q|\mathbf{L_q} = \mathbf{l_q}] \\
&= \sum_{l_{k_q}=0}^{n} \Pr[L_{K_q} = l_{K_q}] \cdot \frac{1}{C_n^{l_{k_q}}} = \sum_{l_{k_q}=0}^{n} \frac{1}{2^n} = \frac{n+1}{2^n}
\end{aligned}$$

This expression (that is also found in [17] following a different approach) is independent of q: it demonstrates that the success rate of the Bayesian adversary does not increase if he gets more leakages. We note that the result is quite intuitive: if we know $W_H(k_0)$ and learn $W_H(k_1)$ and $W_H(k_0 \oplus k_1)$ for some random k_1, the information we get about the value of k_0 is null. Interestingly, this conclusion does not depend on a divide-and-conquer strategy (the complete n-bit key is targeted at once) nor on the amount of randomness in the physical observations (that are here considered noise-free). But it also holds if the adversary receives the Hamming weight of b-bit parts of the n-bit key. In the latter case, the success rate on this part of the key would be turned into $\frac{b+1}{2^b}$.

6.2 Generalized Hamming weight and identity leakage functions

The previous section shows that for a Hamming weight leakage function, the success rate of a Bayesian side-channel adversary against our PRNG is independent of the number of PRNG rounds observed by the adversary. But this is not a generally true statement for other practical leakage functions. In this section, we consequently intend to investigate other examples. Namely, we consider the previously defined generalized Hamming weight and a noisy identity leakage functions. As already mentioned, a practical drawback of actual leakage functions is that the sums in Equations (4) and (5) may be hard to compute exhaustively. Therefore, as a first step towards the analysis of practical leakage functions, we approximate them by covering only a statistically meaningful part of the sums. It yields the following statement:

Empirical claim 3. *In the setting of Section 4.2, there exists a value* $0 < u < 1$ *such that the success rate of a Bayesian side-channel adversary against one byte of our PRNG key, exploiting a generalized Hamming weight or identity leakage function, both affected by a sufficient amount of noise in the physical observations, is bounded by* u *for any number of PRNG rounds* q *observed by the side-channel adversary.*

Contrary to the analysis in the previous section, our analysis depends both on the adversarial divide-and-conquer strategy (the claim is stated for one byte of the PRNG key) and the amount of noise in the observed leakages. Since this scenario is complex to investigate analytically, we show empirical evidence that it holds in the following simulation environment.

First, we consider an adversary who targets 8-bit key bytes. As will be emphasized later on, secure implementations of our PRNG also exist against more powerful adversaries (e.g. targeting 16, 32, ...-bit parts of the key).

Second, we simulated different architectures for the PRNG:

1. An 8-bit architecture in which the adversary is provided with respectively $W_H(\mathsf{S}(x_{i,[0\cdots7]} \oplus k_{i,[0\cdots7]}))$ and $W_H(m_{i,[0\cdots7]})$ in the generalized Hamming weight case and with $k_{i,[0\cdots7]}$ and $m_{i,[0\cdots7]}$ in the identity leakage function case. The latter example gives rise to a straightforward 100% success rate.
2. A 16-bit architecture in which the adversary is provided with respectively $W_H(\mathsf{S}(x_{i,[0\cdots7]} \oplus k_{i,[0\cdots7]})) + R(8)$ and $W_H(m_{i,[0\cdots7]}) + R(8)$ in the generalized Hamming weight case and with $k_{i,[0\cdots7]} + R(8)$ and $m_{i,[0\cdots7]} + R(8)$ in the identity leakage function case. In this context $R(8)$ represents the leakage of the 8 bits that are not targeted by the adversary and consequently produce what is usually referred to as algorithmic noise.
3. A 32-bit architecture in which the adversary is provided with respectively $W_H(\mathsf{S}(x_{i,[0\cdots7]} \oplus k_{i,[0\cdots7]})) + R(24)$ and $W_H(m_{i,[0\cdots7]}) + R(24)$ in the generalized Hamming weight case and with $k_{i,[0\cdots7]} + R(24)$ and $m_{i,[0\cdots7]} + R(24)$ in the identity leakage function case. In this context $R(24)$ represents the leakage of the 24 bits that are not targeted by the adversary.

4. 64-bit, 128-bit and 256-bit architectures that are defined similarly.

Since the PRNG inputs are not under control of the adversary, it is not possible to switch the algorithmic noise source off (e.g. by feeding the device with constant inputs). In our simulations and for any b-bit architecture, we assumed the leakage of the un-targeted $b-8$ bits in the implementation (i.e. the $R(b-8)$ parameter) to be normally distributed with mean $\frac{b-8}{2}$ and variance $\frac{b-8}{4}$. We note that there exist other ways to introduce noise in the physical observations.

For each selected architecture and leakage function, we generated 2000 random keys and computed the success rate from their corresponding leakages for different number of PRNG rounds (from 1 to 20) with their 95% confidence intervals. The results are in Figures 5, 6, from which we conclude:

1. Increasing the size of the architecture generally decreases the success rate as long as the adversary cannot guess all the bits. This is caused by the increased amount of algorithmic noise in the side-channel measurements. As previously mentioned, it is also true for more powerful adversaries. For example, an adversary targeting 16 bits of a 32-bit (resp. 64-bit) architecture would have its observations affected by 16 bits (resp. 48 bits) of noise. Importantly, the meaningful parameter to reduce the success rate against one part of the key is the size of the architecture (not algorithm). This point emphasizes that our PRNG will hardly be secure when implemented in small (e.g. 8-bit) controllers while provide a higher security level when implemented in larger devices (e.g. ASICs, FPGAs).

2. For the investigated leakage functions, increasing the number of observed rounds does not significantly improve the success rate, which saturates after very few rounds. The exact saturation value of the success rate is typically dependent on the structure of the leakage function. For instance, considering the identity leakage function, the success rate saturates after less than 10 PRNG rounds, around 0.18 for a 128-bit architecture implementing AES-128 and around 0.08 for a 256-bit architecture implementing AES-256.

7 Practical consequences

The simulations that we performed in the previous section show that our construction allows bounding the success rate on a single byte of the key. This result is of practical importance as it allows using our observation of Equation (1) and to deduce that the global success rate will increase exponentially with the length of the key. For instance, if we go back to the architectures we mentioned in the previous paragraphs, we obtain that $\mathbf{Succ}^{\text{sc-kr-K}_q}_{\text{AES128,A}} \simeq (0.18)^{16} \simeq 2^{-40}$ and $\mathbf{Succ}^{\text{sc-kr-K}_q}_{\text{AES256,A}} \simeq (0.08)^{32} \simeq 2^{-116}$ for the corresponding two architectures, which seems quite reasonable for practical applications. We see that the success rate decreases both because of more algorithmic noise: $\mathbf{Succ}^{\text{sc-kr-}K_{q,[0\cdots7]}}_{\text{AES128,A}} \simeq 0.18 \rightarrow \mathbf{Succ}^{\text{sc-kr-}K_{q,[0\cdots7]}}_{\text{AES256,A}} \simeq 0.08$, and because of the increased key size (which is the

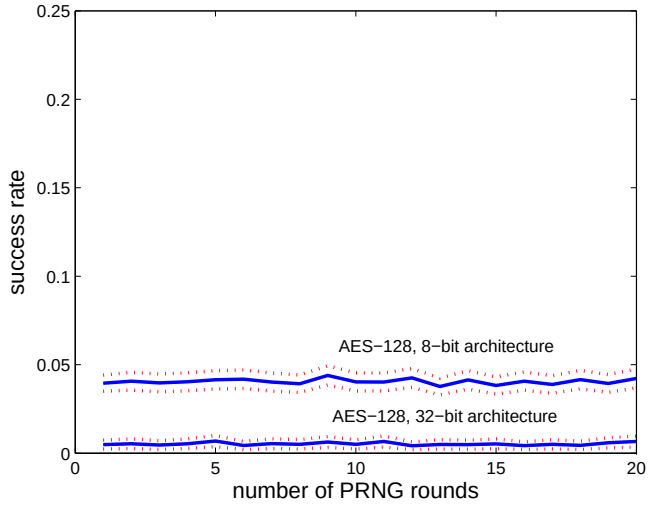

Fig. 5. Attack success rate: generalized Hamming weight leakage function.

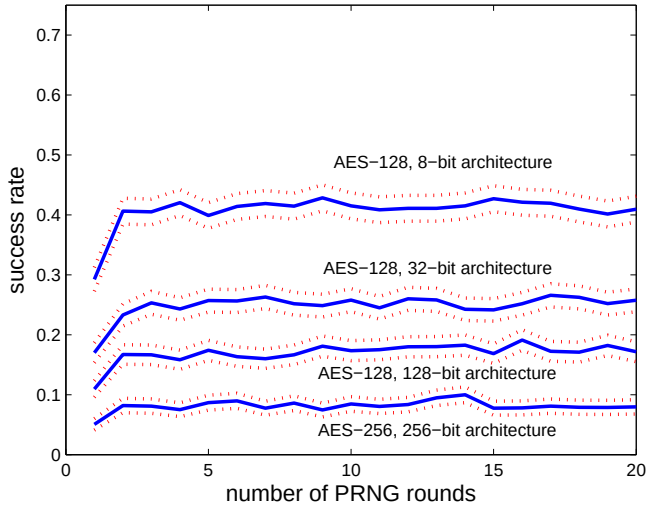

Fig. 6. Attack success rate: identity leakage function.

dominant factor). Therefore, contrary to the previous section, the key size of the algorithm (not implementation) has a significant impact to the global success rate. These results show that even a pessimistic identity leakage function combined with a reasonable amount of (algorithmic or other) noise allows reaching low success rates and therefore, implementations with sufficient security.

Let us finally mention that Claims 2 and 3 in the previous sections only state that the success rate of the Bayesian side-channel adversary against our PRNG is either constant or bounded for any number of observed rounds. But this does not involve actual security if this constant value or upper bound is close to one. Actual security requires that the success rate after the observation of a single PRNG round is sufficiently small (as in the previous examples). This is a requirement for the cryptographic hardware designers. We conjecture that this condition holds for many practical instances of our construction and therefore leads to implementations practically secure against side-channel attacks.

8 Secure initialization of the PRNG with a public seed

A practical limitation in the assumptions of Section 4.2 is the fixed IV that prevents the straightforward initialization of the PRNG by a regular user. In this section, we consequently illustrate the possibility to initialize our PRNG with a public seed in the side-channel context, as usually required in higher-level protocols. For example, such an initialization is useful in a side-channel resistant authentication process, since it allows to challenge the PRNG with various random seeds. Similarly, it can be used to re-synchronize two devices securely. Looking back at Figure 1, the main constraint is that the initialization should not allow the adversary to encrypt an arbitrary number of plaintexts with the same running key k_i, k_i^*, as in a standard side-channel attack. A solution to this problem, illustrated in Figure 7, is to use two initial vectors IV_0 and IV_1. Then, a public n-bit random number r is selected of which we denote the different bits as $r(i)$. This solution holds in two steps:

1. Initialization: n cycles of the PRNG are executed, without outputting any block y_i. The initial vector is selected as follows:

$$z_i = IV_0 \text{ if } r(i) = 0 \text{ and } z_i = IV_1 \text{ if } r(i) = 1$$

2. Generation: after the initialization process, the IV if fixed at IV_0 again and the PRNG outputs the y_i blocks, as in the original description in Section 2.

The black box properties of this initialization process are mainly similar to those of the original PRNG description. Assuming "good" block ciphers in our construction, it is expected that the 2^n possible random numbers r give rise to 2^n different internal states of the PRNG after the initialization. Because of place constraints, we let the formal investigation of this process as a scope for further research. Similarly, our physical security analysis also holds. The only difference

is that the adversary now obtains the leakages corresponding to two input values $x_{i,0}$ and $x_{i,1}$, for each running key k_i, k_i^*. Since in our previous analysis for the PRNG, the amount of information provided to the side-channel adversary is hidden in the leakage function abstraction, the PRNG with initialization process just has to consider more (but still limited) information leakages. Therefore, if sufficient noise is present in the leakage measurements, a sufficient security level can be reached, as in the previous sections.

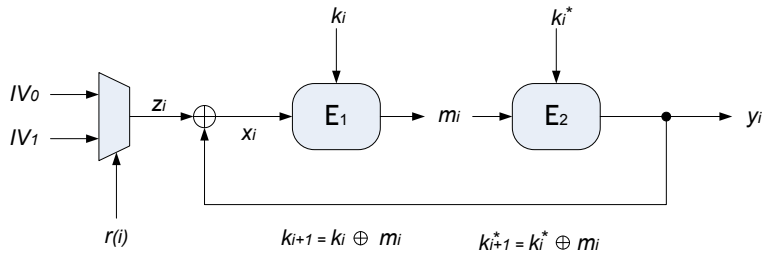

Fig. 7. Secure initialization of the PRNG.

9 Relaxing the assumptions

As previously mentioned, the assumptions 3, 4 and 5 in Section 4.2 reduce the power of the side-channel adversary. In this section, we discuss how relaxing these assumptions has the same effect as initializing the PRNG: it increases the amount of leakages provided to an adversary. But as long as a sufficient amount of noise can be inserted in the physical observations, this does not change our conclusions. In particular, we have for the three considered assumptions:

a3. Actual block ciphers have a key scheduling and its execution generally leaks information. Considering this additional leakage source can be integrated thanks to the leakage function abstraction in the values l_{K_i} and l_{M_i}. Note that if the execution of the key scheduling leaks too much, it is possible to implement the assumption as such, by just increasing the key material in the component ciphers.

a4. Similarly, exploiting the leakages of the block cipher E_2 in our construction can give rise to additional leakages on k_i^0. For example, if a master key is such that $k_0^0 = k_0^r$, the key update involves that this equality will hold for any pair k_i^0, k_i^r. This leads to more information leakages which can again be reflected in the values l_{K_i} and l_{M_i}. Note that a way to improve this is to have different updates for the keys k_i and k_i^*, e.g. using both XOR and modular additions.

a5. Finally, our analysis considers an adversary targeting the first or last iteration of the PRNG. An improved adversary would try to recover an intermediate

key, taking advantage of both the leakage of the previous and forthcoming iterations. This has a similar effect as the observation of an additional plaintext in the initialization process: it increases the information leakages in a limited way.

In summary, those assumptions have to be considered in practice, if an actual implementation is to be designed and its security is to be quantified (e.g. by determining the maximum success rate allowed). But they do not affect our main theoretical result, i.e. for reasonable (meaning practically meaningful) leakage functions, *the success rate of a partial key recovery is bounded.*

10 Conclusions

A block cipher-based PRNG secure against side-channel key recovery is presented. It is based on a re-keying strategy that allows keeping the information leaked to a side-channel adversary under control. Compared to most recent ad hoc countermeasures to prevent side-channel attacks, our proposal has the security advantage of being systematically analyzed against a Bayesian side-channel adversary, which is usually assumed to be the strongest one from an information theoretic point of view. Compared to the physically secure PRNG proposed in [14] by Micali and Reyzin, our proposal is inspired by considerations from experience in side-channel analysis, and is expected to be much more efficient.

Our analysis is based on a hybrid approach, considering the black box computational security and the physical security (modelled by the notion of side-channel key recovery) separately. Our construction allows bounding the success rate of side-channel adversaries when a divide-and-conquer strategy is used to target specific parts of the key. As a result, we obtain that the physical security against side-channel adversary can be increased exponentially, by polynomially increasing the PRNG security parameter, making the probability of a successful attack a negligible function. We believe this analysis technique is not specific to our construction but could be re-used on schemes where the analyzed leakages are associated to rekeying through a XOR operation.

Open problems first include the further investigation of the different working assumptions introduced in this work in order to allow the formal analysis of the PRNG. Considering different adversarial strategies, alternative internal structures for the PRNG (e.g. by changing the key update procedure) and studying its physical implementation in various devices are typical examples. In particular, an interesting question is to determine the minimum architecture size (e.g. 8-bit, 32-bit, 128-bit, ...) required for the PRNG to provide actual security. The impact of the key scheduling algorithm is an important issue to consider with this respect. Since our PRNG can be combined with former countermeasures against side-channel attacks, another practically important question is to determine how to provide security at the lowest implementation cost. We note that the leakage functions (Hamming weight and identity) analyzed in this work correspond to

powerful types of leakage. However, our analysis assumed physical dependencies on all the internal configuration of the target device. An experimental evaluation of the PRNG is consequently required to evaluate the extent to which this could be contradicted in practice. For example, such global dependencies reasonably model the power consumption of a device, but the electromagnetic analysis can provide more local dependencies. The investigation of such more powerful leakages is therefore required. As a first target, we suggest an AES Rijndael-based FPGA implementation of the PRNG using a 128-bit loop architecture.

From a more theoretical point of view, additional research goals include the sound combination of the black box and physical security in a unified way and the reduction of the black box assumptions for the component ciphers in the PRNG (presently considered as ideal ciphers). Extending our security analysis towards the recent work in [9] is another direction for further study. In particular, evaluating the guessing entropy of our construction appears as an interesting open question in order to determine the extent to which combining a side-channel attack against our PRNG with computational power would lead to similar conclusions on the security of the primitive. In a more speculative manner, it would finally be interesting to consider the use of our approach for the construction of other cryptographic primitives than a PRNG.

Acknowledgements: We thank Krzysztof Pietrzak, Dennis Hofheinz and Eike Kiltz from CWI for having pointed out a mistake in a previous version of this paper. We also thank Benoit Libert from UCL for interesting discussions.

References

1. M. Abdalla, M. Bellare, *Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-Keying Techniques*, in the proceedings of Asiacrypt 2000, LNCS, vol 1976, pp 546-559, Kyoto, Japan, December 2000.
2. M. Bellare, J. Kilian, P. Rogaway, *The Security of the CBC Message Authentication Code*, Journal of Computer Systems, vol 61, num 3, pp 362-399, 2000.
3. M. Bellare, T. Kohno, *A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and applications*, in the proceedings of Eurocrypt 2003, LNCS, vol 5656, pp 491-506, Warsaw, Poland, May 2003.
4. J. Black, P. Rogaway, T. Shrimpton, *Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV*, in the proceedings of Crypto 2002, LNCS, vol 2442, pp 320-335, Santa Barbare, USA, August 2002.
5. S. Chari, J. Rao, P. Rohatgi, *Template Attacks*, in the proceedings of CHES 2002, LNCS, vol 2523, pp 13-28, Redwood City, CA, USA, August 2002.
6. FIPS 197, *"Advanced Encryption Standard,"* Federal Information Processing Standard, NIST, U.S. Dept. of Commerce, November 26, 2001.
7. O. Goldreich, *Foundations of Cryptography, vol 1*, Cambridge U. Press, 2001.
8. L. Goubin, J. Patarin, *DES and Differential Power Analysis*, in the proceedings of CHES 1999, LNCS, vol 1717, pp 158-172, Worcester, MA, USA, August 1999.
9. B. Köpf, D. Basin, *An Information-Theoretic Model for Adaptive Side-Channel Attacks*, to appear in the proceedings of ACM CCS 2007.
10. M. Luby, C. Rackoff, *How to Construct Pseudorandom Permutations from Pseudorandom Functions*, SIAM J. of Computing, vol 17, num 2, pp 373-386, 1988.
11. P. Kocher, *Design and Validation Strategies for Obtaining Assurance in Countermeasures to Power Analysis and Related Attacks*, in the proceedings of the NIST Physical Security Workshop, Honolulu, Hawai, September 2005.
12. P. Kocher, *Leak Resistant Cryptographic Indexed Key Update*, US Patent 6539092.
13. S. Mangard, *Hardware Countermeasures against DPA - A Statistical Analysis of Their Effectiveness*, in the proceedings of CT-RSA 2004, Lecture Notes in Computer Science, vol 2964, pp 222-235, San Francisco, CA, USA, February 2004.
14. S. Micali, L. Reyzin, *Physically Observable Cryptography*, in the proceedings of TCC 2004, Lecture Notes in Computer Science, vol 2951, pp. 278-296, Cambridge, Massachusetts, USA, February 2004.
15. C.E. Shannon, *Communication theory of secrecy systems*, in Bell Systems Technical Journal, vol 28, num 4, pp 656-715, 1949.
16. F.-X. Standaert, T.G. Malkin, M. Yung, *A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks*, Version 2.0, Cryptology ePrint Archive, Report 2006/139, 2007.
17. F.-X. Standaert, E. Peeters, C. Archambeau, J.-J. Quisquater, *Towards Security Limits in Side-Channel Attacks*, in the proceedings of CHES 2006, Lecture Notes in Computer Science, vol 4249, pp. 30–45, Yokohama, Japan, October 2006.
18. K. Tiri, M. Akmal, I. Verbauwhede, *A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards*, ESSCIRC 2003.
19. A.C. Yao, *Theory and Applications of Trapdoor Functions (Extended Abstract)*, in the proceedings of FOCS 1982, pp. 80–91.