Practical Leakage-Resilient Pseudorandom Generators

Yu Yu[1], François-Xavier Standaert[1], Olivier Pereira[1], Moti Yung[2].

[1] Université catholique de Louvain, Crypto Group, Belgium.
[2] Columbia University & Google Inc, USA.

ABSTRACT

Cryptographic systems and protocols are the core of many Internet security procedures (such as SSL, SSH, IPSEC, DNSSEC, secure mail, etc.). At the heart of all cryptographic functions is a good source of randomness, and for efficiency, the primitive of pseudorandom generator (PRG). PRG can also be used in the design of stream ciphers, for secure communications. The Internet is nowadays composed of many types of devices with very different hardware and software characteristics. Hence, one of the concerns in such open environments is the information "leakage" and its exploitation via the so-called "side channel attacks".

A very extensive and current research direction is designing basic cryptographic operations that are resistant to such attacks. Recent works on leakage-resilient PRG and stream ciphers did significant progresses in providing tools for the analysis of side-channel attacks in the standard cryptographic setting. But in the absence of a completely sound model for the leakages, the only constructions that can be proven secure require tweaks that do not correspond to the physical intuition. For example, constructions using an alternating structure, in which a key bit-size of $2n$ can only guarantee a security of at most 2^n, have been designed for this purpose.

In this paper, we provide two methodological contributions, allowing to get rid of these tweaks, or to reduce their impact towards negligible performance overheads. First, we show that the leakage-resilience of a natural, i.e. conform to engineering experience, stateful PRG can be proven under a random oracle based assumption. We then discuss the relevance of this assumption, and argue that it nicely captures the reality of actual side-channel attacks. Second, we provide the first construction of a PRG without alternating structure, that exploits the keying material to its full length and that can be proven leakage-resilient in the standard model. For this purpose, we only need to assume a non adaptive leakage function and a small public memory. We also argue that such an assumption is not only realistic, but necessary for any leakage-resilient primitive that grants adversaries with a (stateless) reinitialization capability. Together with weaker requirements for practical implementations, these contributions further reduce the gap between the theory and practice of physically observable cryptography.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Algorithms, Design, Security

1. INTRODUCTION

Side-channel attacks are one of the most dangerous threats against cryptographic algorithm implementations: these attacks circumvent traditional security proofs by going outside the model in which these proofs are realized, and are generally much more effective than traditional cryptanalysis. As a consequence, resilience to these attacks has intensively been studied during the last fifteen years, through the design of countermeasures at low abstraction implementation levels, using gate masking and randomization techniques or specific logic styles for instance, and assessing the effectiveness of these countermeasures through experimental evidence. More recently, an important body of works addressed this issue at a higher abstraction level, proposing models capturing physical attacks and designing new primitives of which the security can be proven within these models [1, 3, 4, 7, 9, 10, 11, 12, 19, 23, 25, 28, 29, 33, 35, 37].

These new models adopt a point of view that is complementary to the traditional one: while the traditional approaches aim at limiting the amount of information leaked by a device, the recent works cited above aim at limiting the impact of information leakage. In the bounded retrieval model [5, 8], one considers an attacker who is not able to get more than a certain amount of information through leakages, this amount of information being bounded for the whole lifetime of the system. Such a model appears to be particularly suitable for mitigating the risks of memory attacks such as [13], for instance. In another line of work, introduced by Dziembowski and Pietrzak [23], the computation performed by a leaking device is partitioned into rounds, and it is assumed that information leakages occur independently in each round, and are a function of the part of the system state that is active during the corresponding round. In this setting, the amount of information leaked per round is still expected to be bounded, but the total amount of information leaked can

CCS '10, October 4-8, Chicago, Illinois, USA,

be much larger than in the bounded retrieval model. This second approach, which we adopt here, is particularly suitable for the analysis of cryptographic primitives that are attacked through power or electromagnetic analysis for instance [21, 30], and are inherently executed iteratively, which is most common for the symmetric cryptography primitives that we are studying in this paper. The advantage of such an approach is that it simplifies the (arguably difficult) task of hardware designers. Rather than asking them to protect an implementation against an adversary who can monitor the leakages corresponding to an arbitrary number of iterations of the primitive, we only require to protect a single iteration, and rely on sound mathematical tools to ensure that the security of one round can be extended to the security of multiple rounds. In other words, we still rely on low abstraction level countermeasures to protect the cryptographic implementations, but in a less demanding way.

The main ingredient for resilience to bounded leakages per round is key update: One designs schemes in such a way that, in each round, leakages occur on different keys (or secret data). This, in turn, ensures that repeated measurements occurring during different rounds cannot be easily combined in order to recover a complete key. The intuition behind needing key update is demonstrated by looking at the side-channel attack resilience of the two pseudorandom generators depicted in Fig. 1. Running the ANSI X9.17 PRG, which is shown in high-level principle in Fig. 1a, requires, in each iteration, the encryption of some values with the key k. As a result, the leakage of even one single bit of k per iteration might be enough to fully compromise this generator after a few iterations. On the other hand, the stateful PRG shown in Fig. 1b involves computation with updated keys in each round, which is expected to make this type of construction much more resilient to side-channel attacks.

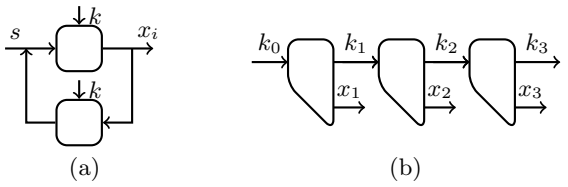

Figure 1: (a) ANSI X9.17 PRG, (b) Stateful PRG.

Related work. In fact, the intuitive idea of combining a bounded leakage per iteration with regular key updates, e.g. for block ciphers, is not new. Shortly after the publication of the first power analysis attack [21], Paul Kocher listed possible countermeasures in which similar principles are found (see [20] for instance). Therefore, the novelty in these previous works mainly lies in the advanced techniques that they provide for evaluating and analyzing physical attacks. But, as extensively discussed in [35], none of these solutions combines all the features that one can wish for a leakage-resilient construction. That is, the proposals that are most satisfying from a theoretical point of view fail to convey some important engineering intuition. And the proposals that are most satisfying from a practical point of view could not be proven secure with general cryptographic techniques.

Let us elaborate on this, focusing on the case of stream ciphers, which are often implemented in small embedded circuits and are a target of choice for side-channel attacks. The construction of Dziembowski and Pietrzak [9], which is proven secure in the standard model, exploits an alternating structure in which the secret key size is doubled, and a combination of extractor and PRG is used to process the iterations. By assuming that the leakages occurring when computing in one part of the alternating structure are independent of the variables manipulated in the other part (i.e., independent computations result in independent leakages), the authors are able to prove the security of their construction, against a very wide class of leakage functions. Namely, a class that includes any polynomial time function of the device's manipulated (i.e., active) state and that can be adaptively chosen by the adversary, provided it satisfies some bounds on the amount of leaked information.

The construction of Pietrzak [29] relies on similar principles, but replaces the extractor and PRG by a single wPRF (weak pseudorandom function), which can be easily instantiated with block ciphers such as the AES Rijndael, at the cost of worse security bounds in the proofs. Finally, a third stream cipher construction was proposed in [37], together with an instance of low complexity extractor, aiming at better practical security and stronger leakage-resilience. This last paper shows that the parallel computation of the two branches in an alternating structure can be exploited for these purposes. It also brings tools that better connect the results of theoretical analysis with the practice of side-channel attacks (e.g., by introducing more realistic assumptions that can be empirically verified by cryptographic engineers).

From an engineering point of view, the main limitation of the three aforementioned papers relates to the difficulty of properly modeling the leakage function. That is, by considering any polynomial time leakage functions in their analysis, the resulting constructions need to face (unrealistic) "future computation attacks", in which parts of a state that will only be reached after dozens of computation rounds can be leaked during the current computation round. Consequently, and in order to incorporate these (admittedly overly strong) attacks, these constructions require an alternating structure which does not seem to be motivated by physical intuition.

In contrast, the earlier forward secure PRG secure against side-channel key recovery attacks presented at ASIACCS 2008 [28] was not limited by such considerations. But it only provides a security analysis for restricted classes of leakage functions which, although representative of the current state-of-the-art attacks, does not imply the generic security that one can hope for in modern cryptographic designs.

Contributions. With respect to the previous related works, this paper contains two contributions to the analysis and design of leakage-resilient stream ciphers. First, we show the leakage resilience of a natural (i.e., conform to engineering experience) block-cipher based stateful PRG, under a random oracle based assumption. More precisely, in the proof of leakage resilience, we model the length-doubling PRG $\mathsf{2PRG}$, that is the core of the construction, by a random oracle that cannot be queried by the leakage function. This guarantees that the leakage function, while being allowed to leak about

the current inputs and outputs of 2PRG, cannot leak about previous or future invocations of 2PRG. Admittedly, this corresponds to a strong black box assumption. However, we argue that it reasonably captures the reality of actual leakages, which do not provide sophisticated functions that are not present in a circuit state (e.g. a future output of the PRG). We also note that, in the absence of leakages, our stream cipher construction is secure in the standard model.

Second, we show that it is actually possible to prove the security of a leakage-resilient stream cipher in the standard model, without alternating structure, and with a marginal construction complexity increase compared to the construction discussed in the previous paragraph. For this purpose, we only need to rely on the additional assumption that the leakage function is not chosen adaptively by the adversary, but is fixed prior to the attack. We describe two designs of leakage-resilient stream ciphers in this new setting, either based on the combination of an extractor and a PRG, or based on a single wPRF, as in previous works.

To summarize, this work addresses the leakage resilience of some of the most natural PRG / stream cipher constructions, either under a non standard assumption, or in the standard model, at the cost of a slight memory overhead.

2. NON ADAPTIVE LEAKAGES

As a starting point, let us clarify the slightly confusing terminology used in previous works on leakage resilience. Most proposed stream ciphers build on an arbitrary length PRG, of which the outputs are used as a keystream. But in practical applications, a stream cipher additionally requires a reinitialization process, i.e. the ability to re-synchronize with a remote device, without sharing a new secret seed. For example, all candidates to the eStream competition have a public Initialization Vector (IV) as input, in addition to the secret seed [27]. And the constructions in [9, 29, 37] miss such a feature. Efficient solutions exist to complement these designs with a secure reinitialization process, as first proposed in [28] and further detailed in [35]. But these solutions and, in fact, any leakage-resilient reinitialization process in a stateless device (i.e., a device that does not save any part of its state between two reinitializations), would be incompatible with an adaptive selection of the leakage function. Indeed, if the adversary is allowed to obtain a different leakage function of the first round internal state after every reinitialization of the cipher, then leakage functions that output one single bit of the key are again sufficient to get the full secret key after a few dozen of reinitializations.

Therefore, at least, one should require that when reinitializing a device to the same state (e.g., with the same IV), the leakage function cannot be modified adaptively. But from an operational point of view, the adaptivity of the leakage function relates to the ability to change the measurement conditions during a side-channel attack (e.g., the antenna's position in an electromagnetic analysis). Hence, whatever modification of the setup that an adversary can do when reinitilizing to a new state can also be done when reinitializing to a constant state. These observations suggest that considering adaptively selected leakage functions provides an overly strong model. We will therefore consider static leakages functions determined before an execution starts, which are reused after each resynchronization. This will provide the important benefit of making it possible to avoid the alternating structure in our construction of Section 4.

We note that this assumption corresponds to the way side-channel attacks are often conducted in practice, i.e. in a setting where the leakage function is determined in advance by the analyzed device and measurement equipment, and not adaptively chosen by the adversary during the measurements. It is also done without loss of generality. In order to reflect the possible adaptivity of the measurement conditions, we can include it in the adversary's abilities and quantify it directly in the bounded leakage assumption (i.e., adaptive leakage functions imply slightly more leakage). This is, in fact, how an actual security evaluation would proceed, i.e. by finding the best probe(s) position(s) and evaluating the resulting leakage as a function of the data complexity.

3. STATEFUL STREAM CIPHERS

3.1 Motivation

Generating pseudorandom streams by following the general idea of a stateful scheme, as pictured in Figure 1b, is intuitively appealing: such a construction guarantees that each key is only used in the system for a very limited amount of rounds – two, actually: in the round where the key is produced and in the round where it is used. This appears to be fairly minimal, and suggests that the implementation of this construction should only guarantee the secrecy of keys that are involved in the measurements that could be performed during only two rounds. Such a construction can however not be proven secure in the model proposed by Dziembowski and Pietrzak (which we will refer to as the DP model) [9]. Indeed, the DP model states that, in each round, a leakage occurs that can be any adversarially chosen polynomial time function that does not decreases too much the (HILL pseudo-) entropy of the output (say, the leakage cannot make the entropy of the key decrease by more than λ bits). Therefore, the adversary might perform a so-called "future computation attack", by requiring the leakage at round i to be the i-th bit of the key that will be used in the $(n+1)$-th round: the full information on that bit is already part of the state at round i. As a result, even though the leakage functions only provides one single bit of information, the full key of the $(n+1)$-th round is obtained by the adversary at the end of the n-th round, and the adversary can, from then on, compute the outputs of all future rounds of this construction

The use of an alternating structure, of the kind depicted in Figure 2, provides a solution to this problem: if one assumes that each of the boxes depicted leak independently, no leakage occurring in one box can be used to compute bits that will be manipulated in future boxes. For instance, following the notations of Fig. 2, leaking about k_0 and x_0 will not allow computing bits of k_4, because k_4 depends on k_1 too.

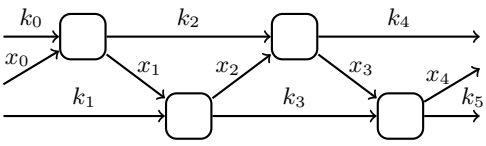

Figure 2: A stream cipher based on an alternating structure.

It however appears unrealistic that such a "future computation attack" has any relation to practice: a circuit will leak about its current state, but will not leak something related to what it will only compute in a dozen of rounds. Besides, the use of this alternating structure has several side effects:

1. The key size is doubled (one needs to initialize the generator with k_0 and k_1) but, independently of leakages, the quality of the randomness produced by the generator corresponds to the use of only one of the keys.
2. It is assumed that the upper and lower parts of the alternating structure leak independently. Still, while one side of the structure is active, the state of the other part must be saved for future use. While it is probably possible to design circuits in such a way that this assumption is satisfied, this would require special care, e.g. splitting the circuit into two insulated parts to avoid coupling effects discussed, e.g. in [2, 35].

The concerns above provide motivation for paying attention to the simple stateful construction that we discussed above:

1. Such constructions should provide randomness of quality directly related to the length of the keys.
2. The requirement about independent leakages can be strongly mitigated: while one would still need to assume that the different rounds leak independently, this appears to be much more natural since the state of past rounds can be erased (explicitly, if needed), and future rounds are not computed nor stored yet.
3. Finally, the insecurity of these constructions in the DP model appears to be the result of an overly strong security model (allowing "future computation attacks") rather than of realistic physical concerns.

3.2 Model and construction

We consider an experiment $\mathsf{Pred}_{\mathsf{A},\overline{\mathsf{L}}}(n)$ to define the security of our stateful PRG. It has three parameters: an adversary A, a vector of leakage functions $\overline{\mathsf{L}}$ that contains one leakage function per PRG round, and a security parameter n. The goal of the adversary in this experiment is to distinguish the output of the stateful PRG at round $q+1$ from a uniformly distributed random value, while given the outputs and leakages of the first q rounds. We then say that a stateful PRG is physically unpredictable if the probability of success of any PPT adversary in this experiment is negligible.

Note that, contrary to what is proposed in the DP model for instance, the leakage functions are a parameter of the experiment: these functions are determined before the experiment starts rather than being determined by the adversary during the experiment – this is what we call non adaptive leakages (see discussion in Section 2). In the following definition, we denote the computation occurring in each box of Figure 1b by the $\mathsf{2PRG}$ function. More precisely, we have:

Experiment $\mathsf{Pred}_{\mathsf{A},\overline{\mathsf{L}}}(n)$:

1. A key k_0 is selected uniformly at random in the set $\{0,1\}^n$, and a counter i is set to 0.
2. On input 1^n, adversary A starts sending $\mathsf{request}$ queries. On each $\mathsf{request}$ query, the counter i is incremented, the pair (k_i, x_i) is computed as $\mathsf{2PRG}(k_{i-1})$, and the leakage $\mathsf{L}_i(k_{i-1})$ is returned to A, together with x_i.
3. When A outputs a test query, a bit $b \in \{0,1\}$ and a value $r \in \{0,1\}^n$ are chosen uniformly at random, and r is given to A if $b = 0$ or x_{i+1} is given otherwise, computed as $(k_{i+1}, x_{i+1}) := \mathsf{2PRG}(k_i)$.
4. A outputs a bit b', and $\mathsf{Pred}_{\mathsf{A},\overline{\mathsf{L}}}(n)$ is set to 1 iff $b = b'$.

Definition 1. *A stateful PRG is* physically unpredictable *for the family of leakage functions $\overline{\mathsf{L}}$ if, for every PPT adversary A, there is a negligible function negl such that:*

$$\mathsf{Pred}_{\mathsf{A},\overline{\mathsf{L}}}(n) = \frac{1}{2} + \mathsf{negl}(n).$$

Obviously, no PRG can be physically unpredictable if we do not place any restriction on the leakage functions: these could simply leak the full key, or be used to perform a "future computation attack". Therefore, we introduce two restrictions: the first one prevents "future computation attacks", while the second one ensures that the leakages occurring in two consecutive rounds do not leak information that would allow a full recovery of the key involved in these rounds.

Preventing future computation attacks. In order to prevent leakage functions from providing unrealistic information on future computations, we model $\mathsf{2PRG} : k_i \rightarrow (k_{i+1}, x_{i+1})$ as a random oracle that cannot be queried by the leakage function (note however that we do not prevent the adversary from querying the oracle). This ensures that the leakage functions will be able to leak information about the round input and output, but not about values that will be computed in further (or have been computed in past) rounds, since computing this information would require oracle queries. More precisely, we consider leakage functions of the following form: $\mathsf{L}_i(k_{i-1}) := (\mathsf{L}_i^i(k_{i-1}), \mathsf{L}_i^o(k_i, x_i))$.[1] The function L_i^i leaks about the input of round i, while the function L_i^o leaks about the output of that round, none of these two functions being allowed to query the random oracle.

Importantly, the use we make of this random oracle substantially differs from the "standard" random oracle model. While we restrict the leakages functions by preventing them to include random oracle queries, we also make a very mild use of this oracle in our proof. Namely, we do not use the programmability of the oracle, which is known to provide a strictly weaker model that the random oracle model [26]. So, it is not clear how our model compares with the random oracle model (nor whether the two models are comparable).

Besides, our model appears to have a natural correspondence to attacks on circuits implementing block ciphers (i.e., for block cipher based PRGs in our case), where the measured leakages can be interpreted as a simple function of the block cipher input and key during the first few rounds of the computation, and/or as a simple function of the block cipher output and key during the last few rounds of the computation, but where any useful function of the block cipher input and output remains elusive (or is the sign of a completely broken implementation, as shown, e.g. in [31, 32]).

[1] k_i and x_i are computed through an oracle call. Considering this single call is however important, since it corresponds to the computation performed in the currently leaking round.

Note that, as the third section of this paper will be devoted to the security analysis of a construction that can be proven leakage-resilient in the standard model, one can wonder why a security analysis using random oracles still makes sense. As previously mentioned, we believe that this is a useful methodological contribution because it directly corresponds to the engineering intuition that a stateful PRG should have good resistance against side-channel attacks. And although the constructions in the next section do not suffer of the need of an alternating structure anymore, it remains that the small performance overheads that they imply are caused by proof technicalities. In other words, while the constructions in Section 4 may be perfectly convenient to use in a practical setting, and bring the security guarantees of a proof in the standard model, it remains that the modeling of the leakage function is imperfect. Namely, we still need to rule out the "future computation attacks" by design rather than by a sound restriction of the physical leakage (e.g., taking into account the fact that they only compute "simple" functions of a devices' state). It is also worth mentioning that making our security analysis rely on random oracles does not simply lead to trivial results. For example, it allows discriminating the two PRGs of Fig. 1, which confirms that it captures at least a part of the intuition about leakage-resilience. Besides, our random oracle-based approach can be used to analyze the physical security of a construction that is secure in the standard model when side-channel attacks are left out of the analysis. Therefore, we believe that the proof technique proposed in this section could be applied to other constructions, either as a preliminary step in the analysis of their leakage resilience, or in the absence of better solutions, e.g. if implementation efficiency is a critical concern.

Eventually, we observe that more standard variants of our random oracle approach could be considered, that could probably provide similar results. For instance, one could model $\mathsf{2PRG}$ as a PRF F with a perfectly secret key k: in this case, $\mathsf{2PRG}(k_{i-1})$ could be computed as $(k_i, x_i) := (\mathsf{F}_k(k_{i-1}), \mathsf{F}_k(k_{i-1} \oplus 1))$ and the leakage functions would still only take k_{i-1}, k_i and x_i as inputs. The locality of the leakages would then be guaranteed by the fact that k is not part of the leakage function inputs. We will however use this random oracle based model for now, for simplicity.

Bounded leakage per iteration. We require that the leakages given to the adversary preserve the secrecy of the PRG seed in the following sense: the probability that an adversary recovers the seed used as input or provided as output during two iterations of the PRG construction should be small. Considering two iterations is minimal since half of the output of an iteration is the input of the next iteration, and there are therefore two leakages taken on each secret variable.

This is formalized through the following definition.

DEFINITION 2. *Let* $(\mathsf{L}^o, \mathsf{L}^i)$ *be a pair of functions,* $\mathsf{A}^{\mathsf{2PRG}}$ *an algorithm representing the side-channel adversary with oracle access to* $\mathsf{2PRG}$*, n a fixed integer, and* $\mathsf{Pr}_{\mathsf{Guess}}(n)$ *the following probability:* $\Pr[\mathsf{A}^{\mathsf{2PRG}}(\mathsf{L}^o(k_1, x_1), x_1, \mathsf{L}^i(k_1)) = k_1 : k_0 \leftarrow \{0,1\}^n; (k_1, x_1) := \mathsf{2PRG}(k_0)]$*. The pair* $(\mathsf{L}^o, \mathsf{L}^i)$ *is said to be* ϵ-seed-preserving *for security parameter n and* $\mathsf{A}^{\mathsf{2PRG}}$ *if* $\mathsf{Pr}_{\mathsf{Guess}}(n) \leq \epsilon$.

A pair of functions $(\mathsf{L}^o, \mathsf{L}^i)$ *is said to be* seed-preserving *if, for every PPT* $\mathsf{A}^{\mathsf{2PRG}}$*, there is a negligible function* ϵ *such that* $(\mathsf{L}^o, \mathsf{L}^i)$ *is* $\epsilon(n)$*-seed-preserving for every security parameter n and* $\mathsf{A}^{\mathsf{2PRG}}$ *running on input* 1^n*.*
A sequence of pairs of functions $(\mathsf{L}^o_1, \mathsf{L}^i_1), \ldots, (\mathsf{L}^o_l, \mathsf{L}^i_l)$ *is said to be* uniformly seed-preserving *if, for every PPT* $\mathsf{A}^{\mathsf{2PRG}}$*, there is a negligible function* ϵ *such that each pair of this sequence is* $\epsilon(n)$*-seed-preserving for every security parameter n and* $\mathsf{A}^{\mathsf{2PRG}}$ *running on input* 1^n*.*

Assuming that adversaries only receive outputs of seed-preserving functions is reminiscent of the assumptions in the cryptography with auxiliary input setting [6]. It is a weaker assumption than requiring a high HILL pseudoentropy [16], as in the DP model. We believe it captures physical leakages particularly well in the sense that we do not put constraint on the form of the leakage (e.g., in terms of length or entropy left): it can be a simple computation time, a huge sequence of power consumption measurements, a map of electromagnetic radiations, or anything else. Moreover, it does not rule out the possibility that the adversary would be able to recognize the correct key if given to him. We only require that the leakage functions cannot be inverted efficiently.

Stronger versions of these notions of seed preservation would allow the adversary to recognize whether a candidate key k_1 is the correct one. This can happen in different contexts in practice: it might be the case that half of $\mathsf{2PRG}(k_1)$ is available as public output of a next round, enabling the adversary to perform some comparisons; it might also be the case that the adversary is able to reinitialize the circuit to a value of his choice, and to compare the leakages observed in the targeted execution to the leakage occurring in an execution triggered after reinitialization. The security of the PRG construction, as claimed next in Theorem 1, could be rephrased in terms of this strengthened notion of seed preservation. Proofs would remain the same, but reductions would become tighter by a factor corresponding to the number of random oracle queries made by the adversary.

3.3 Security analysis

We show that, as long as the pairs of leakage functions that are evaluated on the same keys are uniformly seed-preserving and can be evaluated efficiently, the stateful PRG construction of Fig. 1b is physically unpredictable in our model.

THEOREM 1. *Let* $\mathsf{A}^{\mathsf{2PRG}}$ *be a PPT adversary playing the* $\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}}, \overline{\mathsf{L}}}(n)$ *experiment with a sequence of leakage functions* $\overline{\mathsf{L}} = ((\mathsf{L}^i_1, \mathsf{L}^o_1), \ldots)$*. Then, we have* $\Pr[\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}}, \overline{\mathsf{L}}}(n) = 1] = \frac{1}{2} + \mathsf{negl}(n)$*, provided that the family of pairs of leakage functions* $(\bot, \mathsf{L}^i_1)$*,* $(\mathsf{L}^o_1, \mathsf{L}^i_2)$*, ... is uniformly seed-preserving and that all leakage functions can be evaluated in probabilistic polynomial time.*
Here, $\mathsf{negl}(n) \leq \frac{p(n)^2}{2^n} + q(n)(p(n)+1)\epsilon(n)$*, where p is an upper bound on the number of* **request** *queries made by* $\mathsf{A}^{\mathsf{2PRG}}$*, q is the number of random oracles queries made by* $\mathsf{A}^{\mathsf{2PRG}}$*, and* ϵ *is the uniform bound coming from the uniform seed-preserving property of the leakage functions.*

A proof is given in Appendix A.

3.4 Practical security analysis

First note that, in order to turn the previous analysis into concrete security bounds, it is essential to propose an instance of 2PRG to implement. For convenience, and following the suggestion of Pietrzak in [29], an easy solution for this purpose is to use a block cipher based construction. As an illustration, assume $\mathsf{BC}_k(x)$ denotes the encryption of a plaintext x under a key k, e.g. with the AES Rijndael. Then, a length-doubling PRG can be instantiated as: $\mathsf{2PRG}(k) = (\mathsf{BC}_k(0^n), \mathsf{BC}_k(1||0^{n-1}))$. From such an instance, the practical security analysis to be performed by hardware designers is straightforward: they need to bound the leakage of an adversary who can only encrypt two known plaintexts, i.e. a 2-limited adversary as defined by Vaudenay in his decorrelation theory [36]. This gives a simple tradeoff between the efficiency and the security of a leakage-resilient stream cipher. That is, by turning the 2PRG of our construction into a 3PRG, 4PRG, ..., the amount of keystream generated per PRG iteration increases, at the cost of more input plaintexts that can be monitored by the adversary. So, depending on the quality and trust of the lower level, one can easily adapt the performances of the construction.

It is interesting to mention that, depending on the block cipher used in the PRG, these instantiations may introduce a gap with the assumptions in the previous section. Just observe that we consider the leakage on the output of a 2PRG $\mathsf{L}^o(k_i, x_i)$ and the one on its input $\mathsf{L}^i(k_{i-1})$ as independent. But if a block cipher with an invertible key scheduling algorithm (e.g., the AES Rijndael) is used, the output leakage may potentially leak on the key that was used to produce this output. This observation is not expected to modify the practical security of the resulting PRG, but suggests that carefully instantiating block cipher based constructions may be important to prevent side-channel attacks. It also recalls that non-invertible key scheduling algorithms (as in the FOX block cipher [18], for instance) are desirable in the context of leaking devices. Alternatively, one may also consider slightly more expensive instantiations, e.g. by replacing $\mathsf{2PRG}(k) := \big(\mathsf{BC}_k(0^n), \mathsf{BC}_k(1||0^{n-1})\big)$ by the following one: $\mathsf{2PRG}(k) := \big(\mathsf{BC}_{\mathsf{BC}_k(0)}(0^n), \mathsf{BC}_{\mathsf{BC}_k(0)}(1||0^{n-1})\big)$.

4. SECURITY IN THE STANDARD MODEL

We now propose two constructions with essentially the same structure and efficiency as the one in Section 3 but with two differences: the security holds in the standard model, and the leakage functions are now restricted to have a range limited to $\lambda < n$ bits (instead of simply being hard to invert as in the previous section). For the rest, we again consider non adaptive leakages, which will play a crucial role in the our security proofs. The main ingredient of these constructions is the use of two public random values p_0 and p_1 that will be used in turn in each round of the constructions. The box of the stateful PRG is then instantiated with a weak pseudorandom function in the first construction, and as a combination of PRG and two-source extractor in the second one. Concrete instances of these functions are proposed too.

4.1 wPRF based construction

In this construction, we will replace the 2PRG that was used in the previous section with a weak pseudorandom function (wPRF) $\mathsf{F}(k, p) : \{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^m$, which is a function with the following property: for a random key $k \in \{0,1\}^\kappa$ no efficient adversary can distinguish $\mathsf{F}(k,\cdot)$ from a random function (with the same range) when queried on random inputs. This is a weaker requirement than what is expected from a standard pseudorandom function (namely, security against adversarially chosen queries).

Definition 3 (wPRF). *An efficient function $\mathsf{F}:\{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^m$ is an (ϵ, s, q)-secure weak pseudo-random function (wPRF) if for all A of size s, for random variables $k \xleftarrow{R} \{0,1\}^\kappa$, $p_1, \cdots, p_q \xleftarrow{R} \{0,1\}^n$, and for random function $\mathsf{R} \xleftarrow{R} \{f : \{0,1\}^n \rightarrow \{0,1\}^m\}$ we have:*

$$|\Pr[\mathsf{A}(\vec{p}, \mathsf{F}(k, \vec{p}))] - \Pr[\mathsf{A}(\vec{p}, \mathsf{R}(\vec{p}))]| \leq \epsilon,$$

where $\vec{p} \stackrel{def}{=} (p_1, \cdots, p_q)$, $\mathsf{F}(k, \vec{p}) \stackrel{def}{=} (\mathsf{F}(k, p_1), \cdots, \mathsf{F}(k, p_q))$, and $\mathsf{R}(\vec{p}) \stackrel{def}{=} (\mathsf{R}(p_1), \cdots, \mathsf{R}(p_q))$, and the probability is taken over the randomness of k, $\vec{p}$, and R.

For $m > \kappa$ and for any public randomness p, $\mathsf{F}(\cdot, p)$ is a PRG. So a straightforward construction is to produce pseudorandom streams by simply iterating this PRG. But this construction would be insecure as, even with non-adaptive leakages (with the same input as F), "future computation attacks" are still possible. This motivates the use of two public random values p_0 and p_1 in our construction, that we will use in alternation (without refreshing) for the pseudorandom stream generation. This is where the non-adaptive selection of the leakage functions is crucial: it guarantees that the leakage functions are independent of p_0 and p_1, even if these values are public. Since either p_0 or p_1 will be part of the leakage function inputs, but not both at the same time, this ensures that no leakage can provide a function of the full system state, preventing "future computation attacks".

As depicted in Figure 3, the initial state of the stream cipher is (p_0, p_1, k_0) for public randomness $(p_0, p_1) \xleftarrow{R} (\{0,1\}^n)^2$ and secret key $k_0 \xleftarrow{R} \{0,1\}^\kappa$. The i-th round of our stream cipher is then computed as: $(k_i, x_i) := \mathsf{F}(k_{i-1}, p_{\rho(i-1)})$, where $\rho(i) := i \mod 2$. The security experiment is essentially identical to $\mathsf{Pred}_{\mathsf{A},\bar{\mathsf{L}}}(n)$, except that the triple (p_0, p_1, k_0) is selected at step 1, that 2PRG is instantiated with the wPRF and public inputs as described above, and that the leakage corresponding to round i is computed as $\mathsf{L}_i(k_{i-1}, p_{\rho(i-1)})$.

This pseudorandom generator can be instantiated with any length-expanding wPRF ($m > \kappa$), which in turn can be realized from any secure block cipher BC: $\{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^\kappa$. That is, if BC is an $(\epsilon, s, 2q)$-secure wPRF, then $\mathsf{F}(k, p_l||p_r) \stackrel{def}{=} \mathsf{BC}_k(p_l) \parallel \mathsf{BC}_k(p_r)$ is an (ϵ, s, q)-secure wPRF (another instantiation technique that does not double the amount of public randomness, but requires BC to be a PRF, can be found in [29]). Thus, compared with the construction in Section 3, the only performance penalty we have to pay is the storage of two public random values, that are used alternatively as inputs to our wPRF. Note that the practical security analysis given in Section 3.4 applies similarly, e.g. we can make a trade-off between security and performance by choosing $m = 2\kappa$, $m = 3\kappa$, and so on. Finally, it is worth mentioning that the requirement of independent

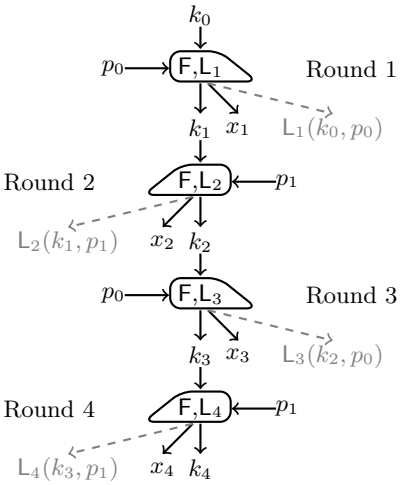

Figure 3: A leakage-resilient stream cipher based on any weak pseudorandom function F:$\{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^{2\kappa}$.

leakages that is found in [9, 29] can also be relaxed here, although not as strongly as for the construction of Section 3. Precisely, our following proofs rely on the fact that, e.g. when computing $\mathsf{F}(k_i, p_0)$, nothing is leaked about p_1. But this is made easier than in [9, 29] for two reasons. First, p_0 and p_1 are public. Hence, they can be manipulated on leaky buses between the round computations, e.g. in order to read them from a part of the chip that does not interfere with the computations. Second, they can be saved once for all, prior to the computations, e.g. in a non-volatile memory. This can be used to reduce possible coupling effects, since non volatile memories do not need to be supplied with energy. The implementation of a similar idea with the constructions of Dziembowski and Pietrzak would be less convenient, since p_0 and p_1 are replaced by secret keys that are modified during the iterations of their stream ciphers. This implies regularly writing in a non volatile memory, which is a highly consuming (and potentially leaking) task.

4.2 Security analysis

Notations. We denote by $\Lambda_i = (x_i, \mathsf{L}_i(k_{i-1}, p_{\rho(i-1)}))$ the information an adversary obtains during the i-th computation round, and let $\mathsf{view}_j(p_0, p_1, k_0) \stackrel{def}{=} \{p_0, p_1\} \cup \{\Lambda_i | i \leq j\}$ be the view of the adversary for all rounds up to j based on initial state p_0, p_1, k_0. We will simply write view_j when this initial state is clear. We use upper-case letters X, Y to denote random variables, and lower-case x, y to denote the values they take. Let $|X|$ denotes the length of X, let U_n denote uniform distribution over $\{0,1\}^n$, and let $X \sim Y$ denote that X and Y are identically distributed. We write $\delta(X;Y)$ for the statistical distance between two distributions X and Y, which is defined as the maximum distinguishing advantage between these distributions with respect to all adversaries. We also use $\mathsf{size}(f)$ to denote the circuit-size complexity of the function f. Finally, we denote the computational analogue of statistical distance by $\delta_s(X;Y)$, which is defined as the maximum distinguishing advantage with respect to all adversaries of size s. If distribution X is over $\{0,1\}^n$, we have: $d(X) \stackrel{def}{=} \delta(X;U_n)$, $d_s(X) \stackrel{def}{=} \delta_s(X;U_n)$, $d(X|Y) \stackrel{def}{=} \delta((X,Y);(U_n,Y))$, $d_s(X|Y) \stackrel{def}{=} \delta_s((X,Y);(U_n,Y))$. The min-entropy of a random variable X, denoted by $\mathrm{H}_\infty(X)$, is given by $-\log \max_x \Pr[X = x]$. A random variable X has HILL pseudoentropy k, denoted $\mathrm{H}^{HILL}_{\epsilon,s}(X)$, if there is a distribution Y with min-entropy k such that $\delta_s(X;Y) \leq \epsilon$.

Security statement. We express the leakage-resilience of the proposed stream cipher in the theorem below. It states that, given the outputs of the stream cipher and the corresponding non-adaptive leakages for any ℓ (polynomial in n) rounds, the $(\ell+1)$-th round output $X_{\ell+1}$ is still pseudorandom with a leakage tolerance of $\log(\epsilon^{-\frac{1}{6}})$ bits per round. This means that for many block ciphers (which are believed to be exponentially secure) the leakage can be a constant portion of the key size κ (see [9] for a discussion). We refer to the appendix for a detailed proof, and outline it below.

Theorem 2 (Leakage-Resilient Security). *Consider the stream cipher introduced in Section 4.1. Let* $\mathsf{F} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^{2\kappa}$ *be any* $(\epsilon, s, n/\epsilon)$*-secure wPRF, and let* L_0, L_1, $\cdots$,$:\{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^\lambda$ *be any sequence of efficient leakage functions. Then for uniform* (P_0, P_1, K_0)*, for any* $\ell \in \mathbb{N}$, $\mathsf{s}_{f,F} \stackrel{def}{=} \mathsf{size}(\mathsf{F}) + \max\{\mathsf{size}(\mathsf{L}_i) | i \leq \ell\}$ *and* $\lambda \leq \log(\epsilon^{-\frac{1}{6}})$*, it holds that:*

$$d_{\frac{\epsilon^2 \cdot s}{2^{\lambda+2} \cdot n^2 \kappa} - (\ell+1) \cdot \mathsf{s}_{f,F}}(X_{\ell+1} | \mathsf{view}_\ell(P_0, P_1, K_0)) \leq (24\ell + 10)\epsilon^{\frac{1}{12}}.$$

Note that, in the above, the output is roughly (keeping only dominating factors) $\ell \cdot \epsilon^{\frac{1}{12}}$-secure against adversaries of much weakened complexity $\epsilon^2 \cdot s - \ell \cdot \mathsf{s}_{f,F}$, which is due to the heavily involved computational reductions. We will discuss in Section 4.3 possible ways to obtain tighter results.

Proof outline. Our proof essentially exploit tools that have been introduced in the previous works [9] and [29].

We first note in Lemma 1 (given in [29]) that a wPRF can be seen as a combination of a strong 2-source pseudorandomness extractor and a PRG, where a two-source pseudorandomness extractor is a function whose output is pseudorandom, even if the two inputs are only weakly random, as long as they are independent and have high min-entropy, and "strong" refers to the additional property that one of its inputs can be public (i.e., p_0 or p_1 in our case).

Besides, even though an arbitrary leakage, say $\mathsf{L}_i(k_{i-1}, p_0)$, will certainly destroy the pseudo-randomness of the wPRF output $(k_i, x_i) := \mathsf{F}(k_{i-1}, p_0)$, it can be shown that, given a bounded leakage, k_i can still keep a high amount of HILL pseudoentropy (see Lemma 2 from [9] for the details).

Therefore, as we use independent randomness p_0 and p_1 in an alternating manner, we can output a pseudorandom k_{i+1} by performing two-source extraction between k_i and p_1, provided that they are independent given the leakages, which is fulfilled by enforcing non-adaptive leakages. That is, L_i takes only (k_{i-1}, p_0) as input and thus, conditioned on it, k_i

and p_1 are independent. Note that k_{i+1} will again lose its pseudorandomness when taking into account the next round leakage, but we can then repeat the proof as above.

Finally, it is important that the security only degrades linearly with the number of rounds, which is confirmed by Lemma 3. It shows that, for every ℓ, the output K_ℓ of the ℓ-th round has high HILL pseudo-entropy given the outputs and leakages up to that round, and that the computating $\mathsf{F}(K_\ell, P_{\rho(\ell)})$ can be seen as a 2-source pseudo-randomness extraction, which completes the proof of Theorem 2.

LEMMA 1 (2-SOURCE EXTRACTION [29]). *Let* $\mathsf{F} : \{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^m$ *be an* $(\epsilon, s, n/\epsilon)$ *secure wPRF for uniform* K *and* X*, then for independent random variables* K' *and* X' *with* $\mathrm{H}_\infty(K') \geq \kappa - \Delta$ *and* $\mathrm{H}_\infty(X') \geq n - \Delta$*, it holds that* $d_{s_a}(\mathsf{F}(K', X')|X') \leq \epsilon_a$*, where* $\epsilon_a = \epsilon \cdot 2^{2\Delta+4}$ *and* $s_a = s \cdot \epsilon^2 / 2n^2$.

LEMMA 2 (PSEUDO-ENTROPY OF PRG OUTPUTS [9]). *For joint random variable* (X,L) *with* $d_{s_a}(X) \leq \epsilon_a$ *and* L *in the range of* $\{0,1\}^\lambda$*, for any* ϵ_b *and* $\Delta > 0$ *satisfying* $\epsilon_a + 2^{-\Delta} \leq \epsilon_b^2 \cdot 2^{-\lambda}$*, there exists a joint random variable* (X', L) *such that:*

1. $\delta_{\epsilon_b^2 \cdot s_a / 8\kappa}((X, L); (X', L)) \leq \epsilon_b$.
2. $\Pr_L[\mathrm{H}_\infty(X'|L) \geq |X| - \Delta] \geq 1 - \epsilon_b$.

LEMMA 3. *For* F, L_0, L_1, $\cdots$, *and* $\mathsf{s}_{f,F}$ *defined as in Theorem 2, for uniform* (P_0, P_1, K_0) *and for all* $\ell \in \mathbb{N}^{>0}$*, it holds that* $\{\mathsf{view}_\ell(P_0, P_1, K_0),\ K_\ell\}$ *is* $((12\ell + 2) \cdot \epsilon^{\frac{1}{12}}, \frac{\epsilon^2 \cdot s}{2^{\lambda+2} \cdot n^2 \kappa} - \ell \cdot \mathsf{s}_{f,F})$*-close to a fake distribution* $\{\widetilde{\mathsf{view}}_\ell(P_0,\ P_1,\ K_0),\ \tilde{K}_\ell\}$ *where, conditioned on* $\tilde{T}_\ell \stackrel{def}{=} \{P_{\rho(\ell-1)},\ \mathsf{L}_\ell(\tilde{K}_{\ell-1}, P_{\rho(\ell-1)})\}$ *we have that* $(\tilde{K}_\ell, \tilde{X}_\ell)$ *and* $\widetilde{\mathsf{view}}_\ell \setminus \tilde{X}_\ell$ *are independent, and:*

$$\Pr_{\widetilde{\mathsf{view}}_\ell}[\mathrm{H}_\infty(\tilde{K}_\ell|\widetilde{\mathsf{view}}_\ell) \geq \kappa - \frac{\log(\epsilon^{-1})}{3}] \geq 1 - 2 \cdot \epsilon^{\frac{1}{12}}, \quad (1)$$

$$\Pr_{\tilde{T}_\ell}[\mathrm{H}_\infty(P_{\rho(\ell)}|\tilde{T}_\ell) \geq n - \frac{\log(\epsilon^{-1})}{3}] \geq 1 - 2 \cdot \epsilon^{\frac{1}{12}}. \quad (2)$$

Note that we use Lemma 1 in an extended scenario since $\tilde{K}_\ell$ and $P_{\rho(\ell)}$ may not be completely independent. But it already suffices to have that (i) $(\tilde{K}_\ell, \tilde{X}_\ell)$ and $\widetilde{\mathsf{view}}_\ell \setminus \tilde{X}_\ell$ (which contains $P_{\rho(\ell)}$) are independent conditioned on some $\tilde{T}_\ell$; (ii) the secret input $\tilde{K}_\ell$ has high min-entropy conditioned on the whole view (Equation (1)); and (iii) the public input $P_{\rho(\ell)}$ has high min-entropy conditioned on $\tilde{T}_\ell$ (Equation (2)).

4.3 Alternative construction with extractors

One limitation of the analysis in the previous section is the loose security bound. A simple way to improve this limitation is to replace the wPRF in Figure 3 by a combination of PRG and a strong 2-source extractor, i.e. to use the function: $\mathsf{F}'(K, P) \stackrel{def}{=} \mathsf{prg}(\mathsf{ext}(K, P))$. As first detailed in [9], F' is not a wPRF, but is actually sufficient for the stream cipher in Figure 3 to be proven leakage-resilient. This extractor-based construction has tighter (but still not tight) bounds than the $\mathcal{O}(\epsilon^{\frac{1}{12}})$ obtained in Theorem 2, since it involves less computational reductions. Namely, by using a 2-source extractor, we avoid using Lemma 1 at the cost of introducing a gap term ϵ_{ext} in an additive manner, which can be made exponentially small. As discussed in [37], it is additionally possible to improve the tightness of the security bounds further, in particular when using parameters of practical size (e.g., a 128-bit key), by relying on an assumption of seed-incompressible PRG, originally introduced in [15].

Note that relying on seed incompressibility and/or bounded leakages also raises practical issues, since these assumptions appear to be difficult to fulfill by hardware designers. In order to get rid of this other type of limitation, [37] introduces empirically verifiable assumptions (namely, the next-block-unpredictability of the 2PRG with simulatable leakages), under which security proofs can be obtained in the standard model. But this again comes at the price of worse security bounds. Determining whether it is possible to obtain tight bounds, under assumptions that can be quantified in practice, is an interesting scope for further research.

We finally mention that these improvements come at the price of implementing an extractor on a leaking device. As pointed out in [34], the use of extractors in physically observable cryptography can be paradoxical. On the one hand, they allow recovering (pseudo) entropy losses if their implementation does not leak too much. On the other hand, the implementation of the extractor can become a good target for standard DPA attacks, because it involves repeated arithmetic operations between its public and private inputs. This illustrates the interesting tradeoff between practical security and tight bounds in the proofs. Finding the best compromise will require better knowledge on the implementation of extractors with low abstraction level countermeasures, which is another scope for further investigations.

5. CONCLUSION

Designing leakage-resilient cryptographic primitives implies intricate tradeoffs between meaningful physical assumptions and sound mathematical proofs. It results in an ongoing discussion about how to best combine these two goals, in order to obtain general proofs under minimum assumptions. Taking the example of pseudorandom generators, this paper first analyzes the security of a simple stateful construction, with minimum physical requirements. We show that this construction can be proven secure under a random oracle based assumption. Then, we show that the price to pay for relaxing the random oracle assumption, and have a proof in the standard model, is actually small. That is, one just needs to assume a small public memory and non adaptive leakage functions. More precisely, we reflect the adaptivity of the measurement conditions in the leakage bounds provided by cryptographic engineers, rather than in the adversarial experiment in our proofs, which is conform to the practice of side-channel attacks. Previous discussions in leakage-resilient cryptography also apply to our new constructions. For example, replacing a wPRF by a combination of PRG and extractor allows obtaining better security bounds, at the cost of implementing a (potentially leaky) extractor.

Avenues for further research could be in two main directions. On the one hand, the design of dedicated solutions that efficiently withstand side-channel attacks could be extended towards other cryptographic primitives. Such advances are necessary for the application of leakage-resilient constructions in real-world applications, where good implementation performances are required. On the other hand, and more generally, the introduction of alternative restrictions of the leakage function would also be useful, e.g. in order to rule out unrealistic "future computation attacks" from the model rather than by design tweaks. Such improvements could open the way towards a more generic treatment of physically observable cryptographic implementations.

Acknowledgments. We thank Yevgeniy Dodis for interesting discussions about the role of the random oracle in our proofs. Olivier Pereira and François-Xavier Standaert are associate researchers of the Belgian Fund for Scientific Research (FNRS-F.R.S). This research has been funded in parts by the Walloon region SCEPTIC project.

6. REFERENCES

[1] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.

[2] M.L. Akkar, R. Bévan, P. Dischamp, and D. Moyart. Power analysis, what is now possible.... In *Proceedings of ASIACRYPT 2001*, volume 1976 of *LNCS*, pages 489–502, Kyoto, Japan, dec 2001.

[3] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In Halevi [14], pages 36–54.

[4] Frederik Armknecht, Roel Maes, Ahmad-Reza Sadeghi, Berk Sunar, and Pim Tuyls. Memory leakage-resilient encryption based on physically unclonable functions. In Matsui [22], pages 685–702.

[5] 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, *Third Theory of Cryptography Conference, TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 225–244. Springer, 2006.

[6] Y. Dodis, Y. Tauman Kalai, and S. Lovett. On cryptography with auxiliary input. In *Proceedings of STOC 2009*, pages 621–630, Bethesda, Maryland, jun 2009. ACM.

[7] Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In Micciancio [24], pages 361–381.

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

[9] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302. IEEE Computer Society, 2008.

[10] Sebastian Faust, Eike Kiltz, Krzysztof Pietrzak, and Guy N. Rothblum. Leakage-resilient signatures. In Micciancio [24], pages 343–360.

[11] Sebastian Faust, Leonid Reyzin, and Eran Tromer. Protecting circuits from computationally-bounded leakage. Cryptology ePrint Archive, Report 2009/379, 2009. `http://eprint.iacr.org/`.

[12] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. One-time programs. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 39–56. Springer, 2008.

[13] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: Cold boot attacks on encryption keys. In Paul C. van Oorschot, editor, *USENIX Security Symposium*, pages 45–60. USENIX Association, 2008.

[14] Shai Halevi, editor. *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*. Springer, 2009.

[15] Shai Halevi, Steven Myers, and Charles Rackoff. On seed-incompressible functions. In Ran Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 19–36. Springer, 2008.

[16] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[17] Antoine Joux, editor. *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science*. Springer, 2009.

[18] Pascal Junod and Serge Vaudenay. Fox : A new family of block ciphers. In Helena Handschuh and M. Anwar Hasan, editors, *Selected Areas in Cryptography*, volume 3357 of *Lecture Notes in Computer Science*, pages 114–129. Springer, 2004.

[19] Jonathan Katz and Vinod Vaikuntanathan. Signature schemes with bounded leakage resilience. In Matsui [22], pages 703–720.

[20] P. Kocher. Leak resistant cryptographic indexed key update. US Patent 6539092.

[21] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[22] Mitsuru Matsui, editor. *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, volume 5912 of *Lecture Notes in Computer Science*. Springer, 2009.

[23] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 278–296. Springer, 2004.

[24] Daniele Micciancio, editor. *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010,*

Zurich, Switzerland, February 9-11, 2010. Proceedings, volume 5978 of *Lecture Notes in Computer Science*. Springer, 2010.

[25] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In Halevi [14], pages 18–35.

[26] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126. Springer, 2002.

[27] ECRYPT Networck of Excellence in Cryptology. The estream project. http://www.ecrypt.eu.org/stream/, 2008.

[28] Christophe Petit, François-Xavier Standaert, Olivier Pereira, Tal Malkin, and Moti Yung. A block cipher based pseudo random number generator secure against side-channel key recovery. In Masayuki Abe and Virgil D. Gligor, editors, *ASIACCS*, pages 56–65. ACM, 2008.

[29] Krzysztof Pietrzak. A leakage-resilient mode of operation. In Joux [17], pages 462–482.

[30] Jean-Jacques Quisquater and David Samyde. Eddy current for Magnetic Analysis with Active Sensor. In *Esmart 2002, Nice, France*, 2002.

[31] M. Renauld and F.-X. Standaert. Algebraic side-channel attacks. to appear in the proceedings of Inscrypt 2009, Lecture Notes in Computer Science, Behing, China, December 2009, Springer, Cryptology ePrint Archive, Report 2009/279. `http://eprint.iacr.org/2009/279`.

[32] M. Renauld, F.-X. Standaert, and N. Veyrat-Charvillon. Algebraic side-channel attacks on the aes: Why time also matters in dpa. In *Proceedings of CHES 2009*, volume 5746 of *LNCS*, pages 97–111, Lausanne, Switzerland, sep 2009. Springer.

[33] François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Joux [17], pages 443–461.

[34] Francois-Xavier Standaert. How leaky is an extractor? in the proceedings of LatinCrypt 2010, Lecture Notes in Computer Science, vol 6212, Puebla, Mexico, August 2010.

[35] Francois-Xavier Standaert, Olivier Pereira, Yu Yu, Jean-Jacques Quisquater, Moti Yung, and Elisabeth Oswald. Leakage resilient cryptography in practice. in "Towards Hardware Intrinsic Security: Foundation and Practice", pp 105- 139, Springer, 2010, Cryptology ePrint Archive, Report 2009/341, 2009. `http://eprint.iacr.org/`.

[36] Serge Vaudenay. Decorrelation: A theory for block cipher security. *J. Cryptology*, 16(4):249–286, 2003.

[37] Yu Yu, Olivier Pereira, and Francois-Xavier Standaert. Leakage-resilient stream ciphers: Bridge the gap. UCL Crypto Group Technical Report, 2010.

APPENDIX

A. PROOF OF THEOREM 1

PROOF OF THEOREM 1. Let $\mathsf{A}^{\mathsf{2PRG}}(1^n)$ be an adversary who wins the $\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}},\bar{\mathsf{L}}}(n)$ game with probability $\frac{1}{2}+\eta(n)$, and let p be a polynomial such that $p(n)$ is an upper bound on the number of request queries made by $\mathsf{A}^{\mathsf{2PRG}}(1^n)$. Let Query_l (resp. Query_a) be the event that $\mathsf{A}^{\mathsf{2PRG}}(1^n)$ makes a query to 2PRG on the last key k_i (resp. any key) computed by the challenger before the test query is made.

We distinguish between the cases where the Query_l event happens or not: $\Pr[\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}},\bar{\mathsf{L}}}(n)=1] \leq \Pr[\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}},\bar{\mathsf{L}}}(n) = 1 \wedge \neg\mathsf{Query}_l] + \Pr[\mathsf{Query}_l]$.

The probability $\Pr[\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}},\bar{\mathsf{L}}}(n) = 1 \wedge \neg\mathsf{Query}_l]$ is bounded by $\frac{1}{2} + \frac{p(n)^2}{2^n}$, which is the sum of the probability of a pure guess and an upper bound on the probability that a collision happens between PRG's last output and an output of a previous round.

We now show that $\Pr[\mathsf{Query}_l]$ is negligible. To this purpose we build an adversary $\mathsf{A}'^{\mathsf{2PRG}}$ as follows.:

Adversary $\mathsf{A}'^{\mathsf{2PRG}}$:

1. On input 1^n, start an instance of $\mathsf{A}^{\mathsf{2PRG}}$ with input 1^n, and record all interactions between $\mathsf{A}^{\mathsf{2PRG}}$ and the 2PRG oracle.
2. Pick $j \leftarrow [0, p(n)]$ and $r_0 \leftarrow \{0,1\}^n$ uniformly at random, and set a counter i to 0.
3. Ask a challenger to pick $k_0 \in \{0,1\}^n$ uniformly at random, to compute $(k_1, x_1) := \mathsf{2PRG}(k_0)$ and to provide $(\mathsf{L}^o_j(k_1, x_1),\ x_1,\ \mathsf{L}^i_{j+1}(x_1))$.
4. On each request query from $\mathsf{A}^{\mathsf{2PRG}}$, proceed as follows: increment the counter i, select $(r_i, y_i) \leftarrow (\{0,1\}^n)^2$ uniformly at random, and submit $\mathsf{L}^i_i(r_{i-1})$, y_i and $\mathsf{L}^o_i(r_i, y_i)$ to $\mathsf{A}^{\mathsf{2PRG}}$, unless $i = j$ in which case $\mathsf{L}^o_j(k_1, x_1)$, x_1 and $\mathsf{L}^i_{j+1}(x_1)$ are submitted instead.
5. On the test query from $\mathsf{A}^{\mathsf{2PRG}}$, pick $y_{i+1} \leftarrow \{0,1\}^n$ uniformly at random and submit that value to $\mathsf{A}^{\mathsf{2PRG}}$.
6. Let $\{z_1, \ldots, z_q\}$ be the set of requests made by $\mathsf{A}^{\mathsf{2PRG}}$ to 2PRG until it halts. Output an element z selected uniformly at random into that set.

The strategy of adversary $\mathsf{A}'^{\mathsf{2PRG}}$ is based on the assumption that, in a normal run of the $\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}},\bar{\mathsf{L}}}(n)$ experiment, $\mathsf{A}^{\mathsf{2PRG}}$ would make a query on (at least) one of the keys involved in the experiment. So, $\mathsf{A}'^{\mathsf{2PRG}}$ makes a uniform guess on the index of the first key on which such a query is made; guessing the first queried key ensuring that that key will only be correlated to one thing: the corresponding leakages (and not any previous call on 2PRG). This guess will be correct with probability $\frac{1}{p(n)+1}$. Then, $\mathsf{A}'^{\mathsf{2PRG}}$ provides leakages to $\mathsf{A}^{\mathsf{2PRG}}$ computed from random values of its own choice, except for the j index, for which the leakages and PRG output are replaced by those obtained from a challenger for the seed-preserving property. $\mathsf{A}'^{\mathsf{2PRG}}$ also provides a random value y_{l+1} as final input to $\mathsf{A}^{\mathsf{2PRG}}$. If the guess on the index j is correct, all the inputs sent to $\mathsf{A}^{\mathsf{2PRG}}$ are distributed exactly as in the $\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}},\bar{\mathsf{L}}}(n)$ experiment, as long as $\mathsf{A}^{\mathsf{2PRG}}$ does not make a query on the value k_1 computed by the challenger. Therefore, when $\mathsf{A}^{\mathsf{2PRG}}$ halts, $\mathsf{A}'^{\mathsf{2PRG}}$ can select one of the

inputs of the q queries made by $\mathsf{A}^{\mathsf{2PRG}}$ and, if $\mathsf{A}^{\mathsf{2PRG}}$ made a query on k_1, that guess will be correct with probability $\frac{1}{q}$. So, eventually, we have that $\Pr[z = k_1|\mathsf{Query}_a] = \frac{1}{q(p(n)+1)}$.

Now, we observe that $\Pr[z = k_1|\mathsf{Query}_a] \leq \frac{\Pr[z=k_1]}{\Pr[\mathsf{Query}_a]}$, and that $\Pr[\mathsf{Query}_l] \leq \Pr[\mathsf{Query}_a]$, which implies that $\Pr[\mathsf{Query}_l] \leq q(p(n)+1)\Pr[z = k_1]$.

Eventually, we observe that $\mathsf{A}'^{\mathsf{2PRG}}$ runs in PPT: $\mathsf{A}^{\mathsf{2PRG}}$ runs in PPT, and the leakage functions can be evaluated in PPT too. Therefore, since the leakage function family $\overline{\mathsf{L}}$ is uniformly seed-preserving, there is a negligible function ϵ such that $\Pr[z = k_1] \leq \epsilon(n)$. As a result, $\Pr[\mathsf{Query}_l] \leq q(p(n)+1)\epsilon(n)$, which is negligible.

So, we have that $\Pr[\mathsf{Pred}_{\mathsf{A}^{\mathsf{2PRG}},\overline{\mathsf{L}}}(n) = 1] \leq \frac{1}{2} + \frac{p(n)^2}{2^n} + q(p(n)+1)\epsilon(n)$, as desired. □

B. PROOFS OMITTED IN SECTION 4.2

We will use the following well-known lemmas in the proofs.

LEMMA 4 (TRIANGLE INEQUALITY). *If $\delta_{s_1}(X;Y) \leq \epsilon_1$, and $\delta_{s_2}(Y;Z) \leq \epsilon_2$, then $\delta_{\min\{s_1,s_2\}}(X;Z) \leq \epsilon_1 + \epsilon_2$.*

LEMMA 5 (REPLACEMENT LEMMA). *For $\delta_s(X;Y) \leq \epsilon$, and for function f with circuit-size complexity $\mathsf{size}(f)$, it holds that $\delta_{s-\mathsf{size}(f)}(f(X);f(Y)) \leq \epsilon$.*

PROOF OF THEOREM 2. First, by applying two 2-source extraction on $\tilde{K}_\ell$ and $P_{\rho(\ell)}$ (see the parameter settings in the proof of Lemma 3), we have:

$$d_{\frac{\epsilon^2\cdot s}{2^{\lambda+2}\cdot n^2\kappa}}(\tilde{X}_{\ell+1}|\widetilde{\mathsf{view}}_\ell) \leq 6\cdot\epsilon^{\frac{1}{12}}.$$

We have also by Lemma 3 and the replacement lemma that:

$$\delta_{\frac{\epsilon^2\cdot s}{2^{\lambda+2}\cdot n^2\kappa}-\ell\cdot\mathsf{s}_{f,F}}((\tilde{X}_{\ell+1},\widetilde{\mathsf{view}}_\ell);(X_{\ell+1},\mathsf{view}_\ell)) \leq (12\ell+2)\cdot\epsilon^{\frac{1}{12}},$$

which also implies that:

$$\delta_{\frac{\epsilon^2\cdot s}{2^{\lambda+2}\cdot n^2\kappa}-\ell\cdot\mathsf{s}_{f,F}}((U_\kappa,\widetilde{\mathsf{view}}_\ell);(U_\kappa,\mathsf{view}_\ell)) \leq (12\ell+2)\cdot\epsilon^{\frac{1}{12}}.$$

Therefore, we complete the proof by applying triangle inequalities to the above. □

PROOF OF LEMMA 3. We denote by sim_{i+1} the algorithm that on input (view_i,K_i), simulates the physical implementation of the stream cipher for round $i+1$ and outputs the updated view view_{i+1}. The case for $\ell = 1$ holds by Lemma 2 (we will deal with the parameters at the end of the proof). It then remains to show by induction on ℓ that if the above statement holds for $\ell = i$ with computational distance (ϵ_i, s_i) then it must hold for $\ell = i+1$ with distance $(\epsilon_i + 12\cdot\epsilon^{\frac{1}{12}}, s_i - \mathsf{s}_{f,F})$. Assume without loss of generality that i is even, and hence the case for $\ell = i$ (by applying sim_{i+1}) implies:

$$\delta_{s_i-\mathsf{s}_{F,f}}(\ (\mathsf{view}_{i+1}(P_0,P_1,K_0),K_{i+1}),\ (\widetilde{\mathsf{view}}_i(P_0,P_1,K_0),\mathsf{L}_{i+1}(\tilde{K}_i,P_0),\underbrace{\tilde{X}_{i+1},\overline{K}_{i+1}}_{\mathsf{F}(\tilde{K}_i,P_0)})\ ) \leq \epsilon_i. \quad (3)$$

By hypothesis $(\tilde{K}_i,\tilde{X}_i)$ and $\widetilde{\mathsf{view}}_i \setminus \tilde{X}_i$ are independent given $\tilde{T}_i$, and that (1) and (2) hold for $\ell = i$, it thus follows by Lemma 1 that with probability $1-2\epsilon_b$:

$$d_{s_a}((\tilde{X}_{i+1},\overline{K}_{i+1})|\widetilde{\mathsf{view}}_i(P_0,P_1,K_0)) \leq \epsilon_a, \quad (4)$$

which in turn implies that:

$$\delta_{s_a}(\ \widetilde{\mathsf{view}}_i(P_0,P_1,K_0);\ \widetilde{\mathsf{view}}_i(P_0,U_n,K_0) \mid\ (\tilde{X}_{i+1},\overline{K}_{i+1},P_0)\ ) \leq 2\cdot\epsilon_a. \quad (5)$$

Then, by Lemma 2 when (5) is additionally conditioned on $\mathsf{L}_{i+1}(\tilde{K}_i,P_0)$ it yields:

$$\delta_{\epsilon_b^2\cdot s_a/8\kappa}(\ \widetilde{\mathsf{view}}_i(P_0,P_1,K_0);\ \widetilde{\mathsf{view}}_i(P_0,P_1',K_0) \mid\ (\overline{K}_{i+1},\tilde{X}_{i+1},P_0,\mathsf{L}_{i+1}(\tilde{K}_i,P_0))\ ) \leq 2\cdot\epsilon_b, \quad (6)$$

where with probability $1-\epsilon_b$ (taken over $(\widetilde{\mathsf{view}}_i,\tilde{K}_i)$):

$$\mathrm{H}_\infty(P_1'|\ (\overline{K}_{i+1},\tilde{X}_{i+1},P_0,\mathsf{L}_{i+1}(\tilde{K}_i,P_0))) \geq n-\Delta, \quad (7)$$

and $\widetilde{\mathsf{view}}_i(P_0,P_1',K_0)$ is independent of $(\overline{K}_{i+1},\tilde{X}_{i+1})$ conditioned on $(P_0,\mathsf{L}_{i+1}(\tilde{K}_i,P_0))$ since L_{i+1} takes only $\tilde{K}_i$ and P_0 (i.e., not P_1) as input. Again by applying Lemma 2 to (4) with $\widetilde{\mathsf{view}}_i(P_0,P_1,K_0)$ replaced with independent $\widetilde{\mathsf{view}}_i(P_0,P_1',K_0)$, we have:

$$\delta_{\epsilon_b^2\cdot s_a/8\kappa}(\overline{K}_{i+1};\tilde{K}_{i+1}|\underbrace{\widetilde{\mathsf{view}}_i(P_0,P_1',K_0),\tilde{X}_{i+1},\mathsf{L}_{i+1}(\tilde{K}_i,P_0)}_{\widetilde{\mathsf{view}}_{i+1}(P_0,P_1',K_0)}) \leq 2\cdot\epsilon_b, \quad (8)$$

where with probability $1-\epsilon_b$ over: $\widetilde{\mathsf{view}}_{i+1}(P_0,P_1',K_0)$

$$\mathrm{H}_\infty(\tilde{K}_{i+1}|\widetilde{\mathsf{view}}_{i+1}(P_0,P_1',K_0)) \geq \kappa-\Delta, \quad (9)$$

and $\widetilde{\mathsf{view}}_{i+1}(P_0,P_1',K_0) \setminus \tilde{X}_{i+1}$ and $(\tilde{K}_{i+1},\tilde{X}_{i+1})$ are independent conditioned on $\tilde{T}_{i+1} \stackrel{def}{=} (P_0,\mathsf{L}_{i+1}(\tilde{K}_i,P_0))$. We thus prove the case for $\ell = i+1$ by applying triangle inequalities to (3), (6) and (8), and the min entropy conditions for $\ell = i+1$ hold by (7) and (9).

Parameter settings. Following [29], set $\Delta = 2\lambda$, $\epsilon_b = 2^{-\lambda+1}$, and thus $\epsilon_b \leq 2\epsilon^{\frac{1}{12}}$, and $\lambda = \log(\epsilon^{-1})/6$. □