Leakage-Resilient Symmetric Encryption via Re-keying

Michel Abdalla[1], Sonia Belaïd[1,2], and Pierre-Alain Fouque[1]

[1] École Normale Supérieure, 45 rue d'Ulm 75005 Paris
[2] Thales Communications & Security, 4 avenue des Louvresses 92230 Gennevilliers
{Michel.Abdalla, Sonia.Belaid, Pierre-Alain.Fouque}@ens.fr

Abstract. In the paper, we study whether it is possible to construct an efficient leakage-resilient symmetric scheme using the AES block cipher. We aim at bridging the gap between the theoretical leakage-resilient symmetric primitives used to build encryption schemes and the practical schemes that do not have any security proof against side-channel adversaries. Our goal is to construct an as efficient as possible leakage-resilient encryption scheme, but we do not want to change the cryptographic schemes already implemented. The basic idea consists in adding a leakage-resilient re-keying scheme on top of the encryption scheme and has been already suggested by Kocher to thwart differential power analysis techniques. Indeed, in such analysis, the adversary queries the encryption box and from the knowledge of the plaintext/ciphertext, she can perform a divide-and-conquer key recovery attack. The method consisting in changing the key for each or after a small number of encryption with the same key is known as re-keying. It prevents DPA adversaries but not SPA attacks which uses one single leakage trace. Here, we prove that using a leakage-resilient re-keying scheme on top of a secure encryption scheme in the standard model, leads to a leakage-resilient encryption scheme. The main advantage of the AES block cipher is that its implementations are generally heuristically-secure against SPA adversaries. This assumption is used in many concrete instantiations of leakage-resilient symmetric primitives. Consequently, if we use it and change the key for each new message block, the adversary will not be able to recover any key if the re-keying scheme is leakage-resilient. There is mainly two different techniques for re-keying scheme, either parallel or sequential, but if we want to avoid the adversary having access to many inputs/outputs, only the sequential method is possible. However, the main drawback of the latter technique is that in case of de-synchronization, many useless computations are required. In our re-keying scheme, we use ideas from the skip-list data structure to efficiently recover a specific key.

Keywords: leakage-resilience, symmetric encryption, re-keying, synchronization

1 Introduction

Most of widely used cryptosystems are secure in the black-box model when the adversary is limited to the observation of the inputs and outputs. However, this model does not faithfully reflect the reality of embedded devices. Introduced in the nineties, a more realistic model in which the attacker can observe the physical leakage of the device has revealed a new class of attacks gathered around the term *Side-Channel Analysis* (SCA for short). These attacks exploit the dependence between sensitive values of an algorithm and the physical leakage of the device (time, power consumption, electromagnetic radiation, ...).

In order to avoid the large variety of side-channel attacks, many countermeasures have been proposed. Most of them are designed to thwart one specific attack. A widely used example is masking [6,12,31] that aims at protecting implementations against *Differential Power Analysis* (DPA) [18] but can be defeated by higher-order attacks [21]. However, over the last few years, significant efforts have been made to define generic models capturing physical attacks in order to provide guarantees of a generic security. Two main examples are the physically observable cryptography [22] and leakage-resilient cryptography [8]. Several pseudorandom functions, generators and permutations have already been proposed in the latter [7,9,27,35,34]. Unfortunately, these primitives are not always relevant to practice. They are often associated to large complexities and are sometimes constructed in a non realistic model in view of current embedded devices. Nevertheless and as stressed in [20], theoretical ideas proposed in the design of these primitives can be used to significantly improve the physical security of cryptographic primitives against side-channel attacks.

In this paper, we propose a more efficient and provably secure symmetric encryption based on fresh re-keying. This technique has first been investigated in [2] in the context of increasing the lifetime of a key and in [17] to thwart side channel attacks by updating regularly the secret key. The principle of re-keying is based on an inherent primitive that, given a master key and a public nonce, generates a new session key. Such schemes have already been designed [19] but no security proof has been given. Here, we rather focus on a mode of operation provably secure in the leakage model. A first requirement for the security is to encrypt each block of message with a different session key. As formally described in [2], the session keys can be generated either from the same master key (in parallel by applying a pseudorandom function on the index with a part of the master key) or sequentially, using the previous session key to derive the current one. Although the choice of the model depends on the underlying primitive, the second one is more suited to avoid DPA as it changes the key at each execution in the re-keying part also. However, the sequential method faces a problem of efficiency when a client and a server need to re-synchronize. For example, servers that operate with many clients (as in electronic cash applications) cannot precompute all the possible session keys. They have to derive them, that is operate as many operations as the difference of indices between the key they currently have and the key they need. As a result, the process of re-keying suffers from the time complexity of the number of similar operations required to obtain the correct session key.

Related Work. The first construction of leakage-resilient symmetric encryption has only been recently proposed in [13] by Hazay *et al.*. However, the main objective of the authors was to propose a generic scheme based on minimal assumptions and the efficiency was not their priority. There are several works on the construction of leakage-resilient symmetric primitives such as block ciphers or stream ciphers [27,7,8]. One of the main assumptions to design such schemes is that AES implementations are heuristically secure against SPA [5,32,23,24,10] or AES can be implemented to be a leakage-resilient block cipher if the number of queries with the same key is small. Consequently, this block cipher is the main practical building block used by theoreticians to instantiate their constructions and namely in [27], Pietrzak proposes to use $AES(0\|p)\|AES(1\|p)$ for constructing a weak PRF with $2n$ bits of outputs. A weak PRF is a PRF when the adversary cannot choose the inputs but only has access to random evaluations of the function. Such weak PRF is a critical ingredient of the design of GGM (Goldreich, Goldwasser and Mical) leakage-resilient PRF [11] which is resistant for many queries and not only two. To construct a leakage-resilient block cipher, Faust *et al.* propose to use this PRF in a three Feistel rounds in [9] but the overall construction has been shown to be inefficient by Bernstein at the CHES'12 rump session [4]. The GGM construction is however very inefficient and in an effort improve it, Medwed, Standaert and Joux in [20] propose a version of the tree by considering byte rather than bit. They analyze the security of this variant with the AES block cipher and lead to the conclusion that AES is not well-suited for this purpose. Indeed, even though the adversary does not control the inputs, she can still efficiently recover the secret key of the first round byte after byte. A similar conclusion has been made by Jaffe in [16] on the AES-CTR mode of operation. Constructing a leakage-resilient PRF is a crucial issue since the construction of a leakage-resilient block-cipher as in [7,9] or a leakage-resilient stream-cipher require this primitive [27]. It is an important problem to design leakage-resilient block ciphers but here, we avoid to consider it when building a practical leakage-resilient symmetric encryption.

Contributions. Our goal is to construct an efficient leakage-resilient symmetric encryption scheme using the AES block cipher without constructing a leakage-resilient block cipher. Since AES is only secure if we encrypt a limited number of plaintexts with the same key, we need to regularly change the key. Therefore, re-keying appears to be an interesting solution as it was earlier proposed by Kocher in [17] to avoid DPA attacks, but here we want to prove that this idea leads to an efficient leakage-resilient symmetric encryption scheme. To this end, we need to construct an efficient re-keying scheme. However, to design such a scheme, one solution is to use a leakage-resilient PRF and we will be back to our initial problem since we want to use AES in an efficient leakage-resilient encryption scheme. Our first solution consists in showing that a leakage-resilient PRF combined with a block cipher is a leakage-resilient encryption scheme. To this end, we can use the GGM construction as proven by Faust et al. in [9]. However, in order to build a more efficient scheme and to solve the synchronization problem, avoiding the computation of all the intermediate keys, we propose a new one. We show that we do not need a PRF to build a secure encryption scheme, but we only need a *leakage-resilient re-keying scheme*. To this end, we use similar ideas from the skip-list data structure [30] proposed by Pugh in the late eighties. In this list, one half of the main list elements are chosen randomly to constitute a new list and from this list, another smaller one are derived and so on using $O(\log n)$ stages with high probability if we have n elements. The idea to look for an element in this *sorted* list consists in beginning with the last floor and identifying the interval where the element is and recurse in the next floor up to identifying the element or finding that it is not in the list. On average, the running time is also $O(\log n)$ which is asymptotically as efficient as a random binary tree. Our problem is similar to finding an element in a sorted list since we have the index of the key we are looking for. It turns out that this construction serves the same purpose than the one proposed by Kocher in [17]. However, the latter does not share the same design mainly because of the multiple use of the same re-keying keys and suffers from the absence of security proof.

Organization. In Sect. 2, we give a theoretical background on leakage-resilient notions. Then, we describe in Sect. 3 our new leakage-resilient symmetric encryption. In Sect. 4, we provide the proof of our construction while in Sect. 5, we evaluate its efficiency in practice.

2 Theoretical Background

In this section, we introduce our security model inspired from most previous works [8,27,9]. We also formally define the functions we use in the following.

2.1 Notations

For the rest of the paper, we introduce some useful notations. In the following, the uppercase letters will be used to denote random variables and lowercase letters to denote their realization. Exceptions are made to define security parameters or sizes. We denote with $R_{m,n}$ the set of uniformly random functions from $\{0,1\}^m$ to $\{0,1\}^n$. For a set $\mathcal{X}$, we eventually write $X \overset{*}{\leftarrow} \mathcal{X}$ to denote the sampling of a uniformly random X from $\mathcal{X}$.

2.2 Preliminaries on Leakage Resilient Cryptography

Bounded Leakage. Under the bounded leakage model introduced in [8], the adversary is limited to the learning of a bounded amount of information. In practice, this model may correspond to a limitation of the number of invocations.

Continuous Leakage. In the continuous leakage model, the attacker has access to an unlimited amount of leakage. There is only a few works on provable security against continuous side-channel attacks and most of them only target one specific attack like probing [15].

Leakage-Resilient Primitives. As in [8], we consider an adversary able to collect a bounded amount of leakage at each invocation of the primitive without being limited in the number of invocations. From the axiom ”Only computation leaks” [22], we assume that only the data involved in an invocation can leak in this invocation. We leave to the adversary the choice of the function f which will be used to compute the leakage. However there is no other choice than restricting the range of this function. Otherwise, the adversary could choose to learn the exact secret state S using the identity function: $f(S) = S$. Hence, we limit to the leakage functions with range $\{0,1\}^\lambda, \lambda \ll |S|$. All the primitives which are secure under these conditions will be referred to as *leakage-resilient primitives*.

Granular Leakage Resilience. In this model introduced in [9], we consider the global cryptographic primitive as a combination of smaller blocks that leak independently. These blocks can be either different primitives or several invocations of the same primitive, following the works [7,34,9]. Let us denote by τ_i the state before step i. The adversary can choose a leakage function f_i before step i and gets $f_i(\tau_i)$ at the end of the step execution. In the following, we will omit the term *granular* but all the schemes will be proven secure under this model.

Non-adaptive Leakage-Resilience (naLR). In this paper and as in [7,9], we allow the attacker to choose a new leakage function per independent block and to learn its output. However, we require these leakage functions to be chosen non-adaptively, that is before obtaining any leakage or outputs. This model actually fits the reality since these functions entirely depend on the embedded devices.

Non-adaptive Function (na). Another relaxation is the notion of non-adaptive function, introduced in [9] for PRFs. In this context, the adversary is expected to choose her input queries in advance, that is before seeing any leakage or output.

2.3 Definitions and Security Notions

Secure and efficient cryptosystems require functions which are indistinguishable from equivalent random objects and which require only a few amount of randomness. A widely used function which fills these criteria is the *pseudorandom random function* (PRF for short). To define the security notion of such a PRF F we consider a *challenge oracle* which is either the PRF $F(K,.)$ instantiated with a uniformly random key K (with probability 1/2) or a uniformly random function $R(.)$. As formalized in Definition 1, the PRF is secure if no adversary is able to distinguish both games with a significant advantage.

Definition 1. *A function $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^n$ is a (ϵ, s, q)-secure PRF if for any adversary (or statistical test) $\mathcal{A}$ of size s that can make up to q disjoint queries to its challenge oracle, we have:*

$$Adv_F^{prf}(\mathcal{A}) = |\mathop{\mathbb{P}}_{K \overset{*}{\leftarrow} \{0,1\}^k}[\mathcal{A}(F(K,.)) = 1] - \mathop{\mathbb{P}}_{R \leftarrow \mathcal{R}_{m,n}}[\mathcal{A}(R(.)) = 1]| \leqslant \epsilon.$$

A weak PRF (wPRF) shares the same definition except that its inputs are chosen uniformly at random. Contrary to the PRFs and as proven in [27], the wPRFs remain secure whenever they are used with the so-called low keys defined below.

Definition 2. *A α-low key $K \in \{0,1\}^k$ is a key with min-entropy equal to $k-\alpha$:*

$$\forall x \in \{0,1\}^k, \quad \mathbb{P}[K = x] \leqslant 2^{-(k-\alpha)}.$$

Both wPRFs and PRFs can be *leakage-resilient* secure, that are secure even if an adversary observes a bounded amount of leakage at each execution. This second security notion requires the introduction of a second oracle referred to as *leakage oracle* and denoted by $F^f(K,.)$. It returns both the output of the function $F(K,X)$ and the corresponding leakage $f(K,X)$ on an input query X.

Definition 3. *A function $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^n$ is a (ϵ, s, q)-secure leakage-resilient PRF if for any adversary $\mathcal{A}$ of size at most s who can make up to q distinct queries to its two oracles, we have:*

$$\begin{aligned} Adv_F^{lr\ prf}(\mathcal{A}) = |& \underset{K \overset{*}{\leftarrow} \{0,1\}^k}{\mathbb{P}} [\mathcal{A}(F(K,.), F^f(K,.)) = 1] \\ &- \underset{R \leftarrow \mathcal{R}_{m,n}, K \overset{*}{\leftarrow} \{0,1\}^k}{\mathbb{P}} [\mathcal{A}(R(.), F^f(K,.)) = 1]| \leqslant \epsilon. \end{aligned}$$

Although they also provide pseudorandomness, encryption schemes are stronger notions than PRFs. Given the ciphertexts of two different messages, no adversary can decide with a significant confidence which ciphertext is related to which plaintext. In this paper, we focus on an equivalent security notion for encryption schemes introduced in [3] and called the *real-or-random indistinguishability.* The security of an encryption scheme is then verified if no adversary can distinguish the encryption of a real query from the encryption of a random string.

Definition 4. *An encryption scheme $S : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is (ϵ, s, q)-secure in the real-or-random sense, if any adversary $\mathcal{A}$ of size at most s who asks at most q distinct queries, has an advantage bounded as follows:*

$$Adv_S^{enc}(\mathcal{A}) = |\underset{K \overset{*}{\leftarrow} \{0,1\}^k}{\mathbb{P}} [\mathcal{A}(S_K(.)) = 1] - \underset{K \overset{*}{\leftarrow} \{0,1\}^k}{\mathbb{P}} [\mathcal{A}(S_K(\$)) = 1]| \leqslant \epsilon$$

where \$ represents a random string in $\{0,1\}^n$.

Let us now define the leakage-resilient security of an encryption scheme.This notion ensures that even with additional leakage, no adversary should be able to distinguish both games with a significant advantage. As for the PRFs, we consider a leakage oracle referred to as $S_K^f(.)$ for a uniformly random key K.

Definition 5. *An encryption scheme $S : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is a (ϵ, s, q)-secure leakage-resilient encryption scheme if any adversary $\mathcal{A}$ of size at most s*

who asks at most q distinct queries has an advantage bounded as follows:

$$\begin{aligned} Adv_S^{lr\ enc}(\mathcal{A}) = |& \underset{K \overset{*}{\leftarrow} \{0,1\}^k}{\mathbb{P}} [\mathcal{A}(S_K(.), S_K^f(.)) = 1] \\ & - \underset{K \overset{*}{\leftarrow} \{0,1\}^k}{\mathbb{P}} [\mathcal{A}(S_K(\$), S_K^f(.)) = 1]| \leqslant \epsilon. \end{aligned}$$

In the following, we will consider a function secure if the advantage of the attacker is negligible in the key size k and if s and q are superpolynomial in k.

3 Leakage-Resilient Symmetric Encryption Scheme

In this section, we propose to build a non-adaptive leakage-resilient symmetric encryption scheme. As suggested by Kocher in [17], this security can be achieved by key updates, also referred to as re-keying, combined with secure primitives. Following this design principle, we propose in a first part a construction based on a naLR naPRF and a block cipher which yields a naLR encryption scheme secure in the sense of Definition 5. In a second part, we focus on the instantiation of the inherent naLR naPRF. We start with the recent construction of [9] since to the best of our knowledge, it is the most efficient proven one. Based on this construction, we try to improve the efficiency of the whole scheme in the context of re-synchronization. However through the improvements, we observe that the naLR naPRF is not a requirement to build a naLR encryption scheme. In fact, we introduce a new re-keying scheme which does not fulfil the properties of a PRF but surprisingly still yields a naLR encryption scheme. Furthermore, it improves significantly the efficiency of a sequential re-keying scheme when instantiated with the AES in the context of the synchronisation issue exhibited in Sect. 1. Eventually, we conclude the section by discussing the generation and the repartition of the public random values used in the whole encryption scheme.

3.1 Leakage-Resilient Encryption using a naLR naPRF

As outlined in [20], PRFs appear to be good candidates for integration in leakage-resilient re-keying schemes. In this work, we show that a naLR naPRF F associated with a secure block cipher β (in the sense of PRF in Definition 1) yields a naLR encryption scheme. For this purpose, Theorem 1 is proven is Sect. 4.

Theorem 1. *Let F denote a naLR naPRF and β a block cipher in the sense of PRF. Then the symmetric encryption scheme $S^{F,\beta}$ is a non-adaptive leakage-resilient encryption scheme. The amount of leakage λ tolerated per time step depends on the inherent naLR naPRF: $\lambda \in O(log(1/\epsilon_F))$.*

The principle is as follows. From an initial secret key k and a public random value p, the PRF outputs a session key $k^\star = F(k, p)$ that is further used by the block cipher for a single block encryption. Figure 1 illustrates this process. Since each block is encrypted with a different secret key, the block cipher is not expected to be resistant against differential attacks.

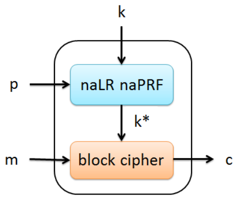

Fig. 1: Non-adaptive leakage-resilient encryption scheme from a naLR naPRF.

3.2 Leakage-Resilient Encryption Scheme without PRF

We have proposed the construction of a naLR encryption scheme from a naLR naPRF. Now we aim to instantiate this naLR naPRF in the most efficient way. Since it is proven secure, we start with the naLR naPRF introduced in [9]. We observe that it has likeable security features but it is not optimal in terms of efficiency since among all the nodes of the binary tree, only the leaves are finally exploited. As a consequence, we propose to improve the efficiency of the whole scheme by also using the intermediate nodes in a suitable order.

A solution to benefit from the intermediate nodes is to slightly change the inherent wPRF. In [9], this wPRF outputs $2n$-bit values from n-bit inputs. In this paper, we refer to as ϕ the wPRF we use to compute n-bit values from n-bit values and as ϕ_2 (resp. ϕ_3) the concatenation of two (resp. three) invocations of ϕ. Instead of deriving two keys generally from two random public values of same size, we could directly use the wPRF ϕ_3 to derive three keys using one more random value. Among these three keys, two would still be used to derive the subsequent tree keys while the third one would be processed in the block cipher. Although this solution exploits the intermediate nodes, it requires a more expensive derivation since the intermediate wPRF is expected to generate one more key with an additional amount of randomness.

A more efficient option is to maintain the binary tree construction with the wPRF ϕ_2 and to use directly the intermediate keys in the block cipher. In this case, a third random value can be used with the intermediate key and the output of the block cipher can be bitwise added to the chosen message[3]. However, the re-keying scheme involved in this new construction is not a PRF anymore, since an adversary could easily take advantage of its outputs to derive the following keys. One may consequently expect the encryption scheme (we refer to as $S^{R_\phi,\phi}$) not to be secure anymore. Surprisingly, this intuition is wrong. By Theorem 2 that will be proven in Sect. 4, we show that we are still able to build a naLR symmetric encryption scheme with relaxed properties on the re-keying scheme.

[3] We could also have chosen to add the message to the random value before the encryption, referring to Lemma 3 from [27] in the case of leakage. However, in case the public random values are known after the first encryption, the message blocks would have been chosen non-adaptively to avoid non random inputs.

However unlike the previous scheme, the proof we established requires the block cipher to be the same primitive than the wPRF used to derive the keys.

Theorem 2. *Let ϕ be a secure wPRF. Let R_ϕ denote the re-keying scheme described above. Then $S^{R_\phi,\phi}$ is a naLR encryption scheme. The amount of leakage λ tolerated per time step depends on ϕ: $\lambda = O(log(1/\epsilon_\phi))$.*

Now that we have presented the security aspects when exploiting all the nodes of the binary tree, we still have to fix a suitable order in the nodes to be as efficient as possible in the re-synchronization scenario. For this purpose, we need to define short-cuts between distant keys to avoid the expensive computation of all the intermediate keys. Inspired by the skip-list data structure introduced in [30], we suggest to organize our re-keying scheme in s stages, $s \geqslant 2$ containing increasingly sequences of keys [4]. This organization in lists, illustrated in Fig. 2

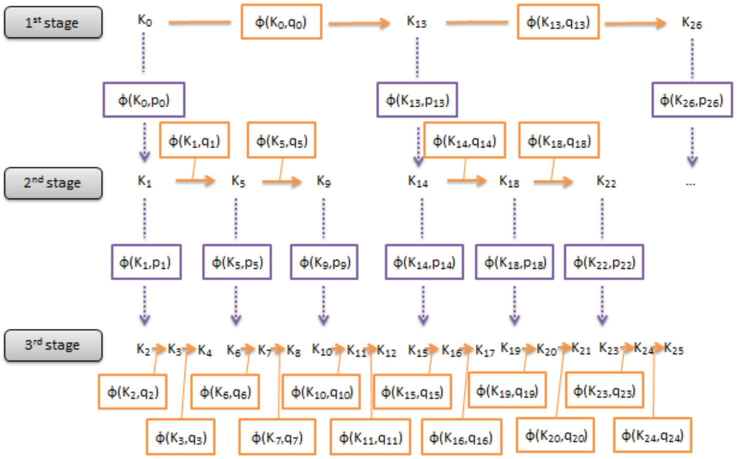

Fig. 2: Stage representation of our new re-keying scheme R_ϕ in the case $s = 3$.

with p_i and q_i public random values, involves a more efficient lookup with a reduction of the complexity from linear to logarithmic. Nevertheless, it is worth noticing that unlike skip-lists, this structure does not expect additional relations between keys. There is still one single computation path to derive each key. Figure 3 illustrates the whole encryption scheme using ϕ_3 for the concatenation of the block cipher and the wPRFs used for the derivation. The values r_i are the public random values used for the encryption, the values m_i represent the blocks of message to encrypt and the values c_i the corresponding ciphertexts.

The very first key K_0 is the entry point of the first stage. Keys from the first stage allow to generate the following first stage keys or to go down to the

[4] In this description, each node generates s nodes at the first upper stage. Although convenient for the analysis, another choice can be made without loss of security.

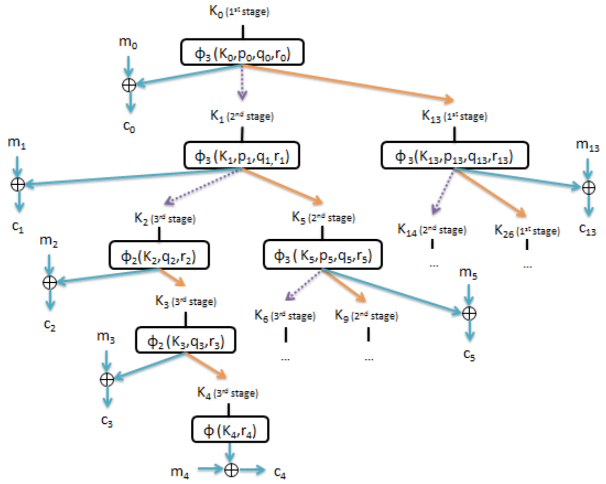

Fig. 3: Tree representation of our new encryption scheme $S^{R_\phi,\phi}$ in the case $s = 3$.

second stage. When going down, the index of the key increases by one whereas when computing the next key in the same stage s_c, the index is increased by $1 + s + \cdots + s^{s-s_c}$. Each derivation requires one public value that we refer to as p_i when going down and q_i otherwise with i referring to the index of the key used for the derivation. In practice, the sender updates his own secret key after each operation, following the indices order. When he wants to perform a transaction with the receiver, he just has to relay his current transaction counter. The receiver then performs a series of operations to derive the expected session key. For this purpose, he can either decides to always start with the very first key K_0 or to start with another key that he already computed and that is compliant with the new index. Algorithm 1[5] depicts the process for the first situation.

3.3 Efficient Generation of Random Values

For efficiency purpose, one tries to minimize the generation of fresh randomness in our construction. We propose several methods to distribute the public random values. In a first attempt, we generate two fresh public random values p and q for the key derivation as illustrated in Fig. 2 and 3 and one additional random value for the input message block. Although the encryption scheme is naLR, the solution is impractical. Another solution is to follow the method from [9] and attribute two fresh random values by tree layer plus one for each block of message. This new proposal reduces the amount of randomness without loss of security since each path uses different random values for each time step.

The work of Yu and Standaert in [35] allows to reduce even more the cost of the generation of randomness. It suggests to replace the randomness by pseudo-

[5] The input s_c can also be directly computed from c.

Algorithm 1 Re-keying Scheme

```
Require: current key $K_c$, index $c$, stage $s_c$, new index $i$
Ensure: key $K_i$, index $i$, stage $s_i$
$(K, ind, s_t) \leftarrow (K_c, c, s_c)$
while $(ind \neq i)$ do
    while $(i > ind + \sum_{j=1}^{s-s_t} s^j)$ do                ▷ Horizontal steps
        $K \leftarrow \phi(K, q_{ind})$
        $ind \leftarrow ind + \sum_{j=0}^{s-s_t} s^j$
    end while
    while $(i > ind)$ & $(i \leqslant ind + \sum_{j=1}^{s-s_t} s^j)$ do   ▷ Vertical steps
        $K \leftarrow \phi(K, p_{ind})$
        $ind \leftarrow ind + 1$
        $s_t \leftarrow s_t + 1$
    end while
end while
return $(K, s_t)$
```

random values generated by a PRF in counter mode from a single public seed. This solution can directly be applied to ours and results in a significant improvement of the performances. The global scheme can still be proven naLR in the peculiar world of `minicrypt` [14], that is either the scheme is secure or we can build public-key primitives from our symmetric functions, which is very unlikely.

Theorem 3. *Let ϕ be a weak PRF and G a PRF. Then the system $S^{R_\phi,\phi,G}$ described above is a naLR encryption scheme or we can build public-key primitives from the PRFs ϕ and G and the related leakage functions.*

4 Leakage-Resilient Security Analysis

In this section, we give the sketch of the security proofs of the two first theorems presented in Sect. 3 and completely proven in the full version of the paper [1] and the proof of the third one from [35].

4.1 Security Analysis of Theorem 1

In Sect. 3, Theorem 1 states the non-adaptive leakage-resilient security of an encryption scheme composed of a naLR naPRF and a block cipher, as illustrated in Fig 1. The following outlines the relevant points of the proof.

Proof Idea for Theorem 1. From the granular leakage-resilient model, our scheme is split into time steps which leak independently. The attacker is allowed to choose non-adaptively a leakage function $f = (f_1, f_2)$ with components for each of these time steps: f_1 for the PRF and f_2 for the block cipher. Then, she can submit q distinct queries to her oracles which can be divided between challenge queries and leakage queries. For each challenge query, she gets back either the real output of her query or the encryption of a random string. For each leakage query, she gets both the real output of her query and the leakage which is exactly the output of the leakage function f she chose. Let us now sketch the proof that the construction is a naLR encryption scheme. As explained in [33], we organize the security proof of Theorem 1 as a sequence of games. The first one, referred to as Game 0, represents the real case, that is when the attacker gets both the leakage and the real outputs of her queries. It directly corresponds to the left-hand side probability in Definition 5 for an adversary $\mathcal{A}$ having access to challenge and leakage oracles:

$$\underset{K \overset{*}{\leftarrow} \{0,1\}^k}{\mathbb{P}} [\mathcal{A}(S_K(.), S_K^f(.)) = 1].$$

We denote by G_0 the event $\mathcal{A}(S_K(.), S_K^f(.)) = 1$ and corresponding to Game 0. The last game Game N represents the random case, that is when all the challenge outputs are generated from random queries. It corresponds to the right-hand side probability in Definition 5 with \$ a random string in $\{0,1\}^n$:

$$\underset{K \overset{*}{\leftarrow} \{0,1\}^n}{\mathbb{P}} [\mathcal{A}(S_K(\$), S_K^f(.)) = 1].$$

Similarly, we denote by G_N the event: $\mathcal{A}(S_K(\$), S_K^f(.)) = 1$. We aim to show that the difference between these probabilities (which is exactly the advantage of the attacker according to Definition 5) is negligible. To proceed, we go through intermediate games and we iteratively prove that the probability $\mathbb{P}(G_i)$ corresponding to Game i is very close to the probability $\mathbb{P}(G_{i+1})$ corresponding to Game $i+1$ for $i \in [N-1]$. . Concretely, we use three kinds of games transition: the transitions based on indistinguishability, the transitions based on failure events and the so-called bridging steps. Basically, we first replace the original game with a game in which the indices are not only random but also pairwise distinct. Then, we replace the session keys computed by the naLR naPRF by uniformly random keys. After that, we define a new game in which the keys are not only random but also pairwise distinct. When all the invocations of the block cipher are totally independent, we replace them one by one by invocations of uniformly random functions. Eventually, we reach the last game, that is when the attacker gets outputs of random strings encryptions. This last game corresponds to the right-hand side probability in Definition 5. Since the differences between successive games are negligible, we finally show that the advantage of any attacker is bounded by a value negligible in the key size.

4.2 Security Analysis of Theorem 2

Unfortunately, our new re-keying scheme is not a PRF since the adversary could easily take advantage of the outputs of the intermediate nodes to recover the following keys. However we prove in the full version [1] that instantiated with a specific wPRF, it still yields a naLR encryption scheme.

Proof Idea for Theorem 2. To prove Theorem 2, we first prove the security of the independent time steps in the new re-keying scheme. Let us consider an intermediate node of the re-keying scheme. If a challenge or a leakage query is defined on this node, the related operation will be the concatenation ϕ_3 of three wPRFs: two for the derivation of the next keys and one for the encryption. In the case no query is defined on this node, the concatenation ϕ_2 of only two wPRFs is required. As a result, we prove that the concatenation of two or three invocations of the wPRF ϕ using the same key (two for the derivation and in some cases one for the block cipher) still forms a secure wPRF.

Proposition 1. *Let $\phi : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(\epsilon, s, 2q)$-secure wPRF.*

$$\begin{aligned} \phi_2 : \{0,1\}^n \times \{0,1\}^{2n} &\to \{0,1\}^{2n} \\ (k, p, q) &\mapsto (\phi(k,p) \| \phi(k,q)) \end{aligned}$$

is then a (ϵ', s, q)-secure weak PRF with $\epsilon' \leqslant \frac{2q(2q-1)}{2^{n+1}} + \epsilon$ [6].

Now we have independent time steps, we can build the proof on the security model previously established. The attacker is still allowed to choose a global leakage function $f = (f_1, f_2)$ but this time f_1 is related to each invocation of the wPRF. With a leakage query, the adversary receives the leakage of the intermediate nodes computed using function ϕ_2 and both the leakage and the output of the last node computed using ϕ_3. From the paper of Shoup [33], we propose a proof as a sequence of games. We modify successively all the nodes involved in leakage and challenge queries. The wPRFs are first replaced node by node by wPRFs instantiated with low keys using Lemma 2 in [27]. Then, the nodes involved in challenge queries are replaced by random functions. Through the different reductions, we eventually prove that the difference between the real game (left-hand side probability in Definition 5) is negligibly close to the random game (right-hand side probability). Basically, it depends on the number of nodes affected by the queries (including the intermediate nodes). Eventually, the new encryption scheme is non-adaptive leakage-resilient.

4.3 Security Analysis of Theorem 3

In 1995, Impagliazzo defined five complexity worlds [14]: `algorithmica` in which $P = NP$ with all the amazing consequences, `heuristica` world in which on the contrary NP-complete problems are hard in the worst-case ($P \neq NP$) but are efficiently solvable on average and the three worlds on the existence of the cryptographic functions. In the `pessiland` world, there exist average-case NP-complete problems but one-way functions do not exist, which implies that we cannot generate hard instances of NP-complete problem with known solution.

[6] Similarly, the security parameter of ϕ_3 is bounded by $\frac{3q(3q-1)}{2^{n+1}} + \epsilon$.

In the `minicrypt` world, one-way functions exist but public-key cryptographic schemes are impossible and finally in the `cryptomania` world, public-key cryptographic schemes exist and secure communication is possible. These worlds have been used positively to establish security proofs in many papers [26,28,35].

In this section, we follow the work of Yu and Standaert who show in [35] how to improve the efficiency of our re-keying scheme, maintaining its leakage-resilient security in the `minicrypt` world. In fact, our new construction currently requires a large amount of fresh randomness since we need to generate a new fresh random value for each new session key. Yu and Standaert show that tweaking a similar design to use only a small amount of randomness can still be leakage-resilient in the world of `minicrypt`. That is, either the new design is leakage-resilient or it becomes possible to build public-key primitives from the involved symmetric-key blocks and the related leakage functions, which is very unlikely. Their technique directly applies to our symmetric encryption scheme and only requires a public seed s that is randomly chosen. Instead of being randomly generated, our public values p_i's and q_i's are now computed from a PRF G in counter mode.

Proof of Theorem 3 from [35]. The scheme is trivially secure if the seed is secret since it is like replacing the outputs of the PRF G by a true random values. Let us now prove the leakage-resilience security when the seed is public. For this purpose, we assume by contradiction that there exists an adversary $\mathcal{A}$ against our scheme. If the scheme is not a naLR encryption scheme, there exists an adversary able to distinguish with a significant advantage the encryption of a real query from the encryption of a random string with the same size given the previous leakage and outputs. Let us now consider a protocol between two parties which we refer to as `Alice` and `Bob` who want to communicate over an authenticated channel. The protocol is a secure bit-agreement if an adversary, refer to as `Eve`, cannot recover the output bit of `Alice`. We construct it as follows:

1. `Bob` generates a random initial key for the re-keying scheme.
2. `Alice` generates the public random seed s and compute the required amount of public values using the PRF G. She sends these values to `Bob`.
3. `Bob` encrypts the message using the random values in the encryption scheme. He obtains the ciphertext c. He then generates a random bit b_B and sends to `Alice` either c if $b_B = 0$ or the encryption of a random value otherwise and in both cases the current view containing the leakage.
4. `Alice` finally fixes the bit b_A with the result of the distinction between the true output and the encryption of a random input.

As `Eve` only has access to the communication, she only gets knowledge of the intermediate public value (but not the seed), the current view and the correct or false result of the encryption. Hence she cannot guess the bit b_A without breaking the scheme with secret seed. From the non negligible advantage of the adversary $\mathcal{A}$, the bit agreement we established achieves correlation ($\mathbb{P}[b_A = b_B]$ is greater than $1/2$). As a consequence, this protocol is equivalent to a bit-PKE in which the secret key corresponds to the seed generated by `Alice` and the public key to the intermediate public values.

5 Practical Aspects

In previous sections, we have shown that our construction instantiated with a weak PRF ϕ and combined with a PRF G yields a non-adaptive leakage-resilient encryption scheme. We now focus on the practical aspects.

5.1 Instantiation

Our encryption scheme requires two primitives: a weak PRF ϕ for the derivation and the block cipher and a PRF G for the generation of random values.

Weak PRF ϕ The concatenation of invocations of the weak PRF ϕ with random inputs is a suitable solution for the key derivation and the block cipher. Such a weak PRF can be built from any secure block cipher, like AES. Hence, inspired by [27], we propose the constructions $\phi_2(k,p) = \mathrm{AES}(k,p\|0)\|\mathrm{AES}(k,p\|1)$, for the key derivation and $\phi_3(k,p,r) = \phi_2(k,p)\|\mathrm{AES}(k,r)$, for also the encryption which benefit from the reuse of one public random input.

PRF G Following [35], we instantiate G with a secure block cipher, e.g. the AES. Since the AES is already implemented for the weak PRF ϕ, this choice benefits from the feature of limiting the code size. As proved in [35], only $\log(1/\epsilon)$ bits of fresh pseudo-randomness are required for each public value, with ϵ the security parameter of the weak PRF ϕ (e.g. AES). As a consequence, we only need one additional call of the AES every $\lfloor n/\log(1/\epsilon)\rfloor$ invocations of ϕ.

5.2 Complexity Evaluation

Let us now focus on the complexity of encrypting a n-block message using our construction. We denote by τ_{AES} the complexity in time of one AES calls either as a PRF for the re-keying or as a block cipher for the encryption. First, note that without updating the secret key and without any mode of operation, the complexity of the encryption is exactly $\mathcal{C} = n \cdot \tau_{AES}$. Then, let us compute the same complexity in our leakage-resilient construction by first omitting the generation of randomness. For the sake of simplicity and because it is negligible, we will omit the complexity of the bitwise addition which is performed once per block encryption. Furthermore, we will start with the initial key K_0 without loss of generality since what counts is the distance between the current index and the targeted one. We recall that the distance between two keys indices from the same stage s_c is equal to $1 + s + \cdots + s^{s-s_c}$. We denote by N_s this distance which is also the number of children of a key from the same stage plus one. As a result, the number of AES executions N required to reach the key K_i is bounded as follows: $\frac{i}{N_1} \leqslant N \leq \frac{i}{N_1} + s(s-1)$ with $s(s-1)$ the maximum number of executions needed to reach a child from a first stage key. These bounds can be squeezed with the parameters related to the other stages. Table 1 presents the number of AES executions required to re-synchronize from

Table 1: Number of AES executions to derive a key from K_0 given its index

	K_{10}	K_{10^2}	K_{10^3}	K_{10^4}	K_{10^5}
#stages = 2	4	34	$3.3 \cdot 10^2$	$3.3 \cdot 10^3$	$3.3 \cdot 10^4$
#stages = 3	4	10	82	$7.7 \cdot 10^2$	$7.7 \cdot 10^3$
#stages = 4	6	8	16	$1.2 \cdot 10^2$	$1.2 \cdot 10^3$
#stages = 5	5	10	15	20	$1.4 \cdot 10^2$
sequential scheme	10	10^2	10^3	10^4	10^5

K_0 to keys with increasing indices. For comparison purpose, when the keys are updated sequentially, 10,000 invocations of the re-keying primitives are required to compute K_{10^4} from K_0. When using our construction with five stages, only $N = 20$ invocations are necessary that is five hundred times less. In the general case, one also needs to consider the generation of random values. Since the generation is also performed with the AES, the complexity of encrypting a n-block message is: $\mathcal{C} = (2N + 4n - 2)\tau_{AES}$ if we consider one invocation of G for each key derivation and each block encryption. From [35] we could reduce the number of invocations of the generator until one every $\lfloor n/\log(1/\epsilon)\rfloor$ invocations of ϕ without loss of security:

$$\mathcal{C} = \left(N + 2n - 1 + \frac{N + 2n - 1}{\lfloor n/\log(1/\epsilon)\rfloor}\right)\tau_{AES}.$$

6 Conclusion

In this paper, we have studied the problem of constructing an efficient and provably-secure symmetric encryption scheme based on re-keying ideas. In particular, we have proven that a naLR naPRF combined with a block cipher yields a non-adaptive leakage-resilient symmetric encryption scheme. Then we have shown that such an encryption scheme does not actually require this level of security for its re-keying scheme. In fact, we have introduced a new re-keying process with relaxed security properties that still yields a secure encryption scheme. Furthermore, it is much more efficient than a sequential scheme when both parts of the symmetric communication need to re-synchronize. We have proven the security based on this new re-keying scheme and evaluated the global complexity.

This work shows that it is possible to use the security of the mode of operations in order to construct leakage-resilient encryption scheme. The previous approach in this area tries to construct leakage-resilient block ciphers but it turns out that they are very inefficient. One of the main drawback of this scheme is that we need to compute the keyschedule algorithm for each message block. One interesting idea would be to avoid it by using a more secure mode of operations such as OCB. Indeed, this mode is interesting since the adversary cannot know what is the real input of the block cipher and consequently classical DPA attack are thwarted. However, the security proof of this mode is a real challenge.

References

1. Michel Abdalla, Sonia Belaïd, and Pierre-Alain Fouque. Leakage-Resilient Symmetric Encryption via Re-keying, 2013. Full version of this paper. Available from the Authors' webpages.
2. Michel Abdalla and Mihir Bellare. Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-keying Techniques. In *ASIACRYPT*, pages 546–559, 2000.
3. Mihir Bellare, Anand Desai, E. Jokipii, and Phillip Rogaway. A Concrete Security Treatment of Symmetric Encryption. In *FOCS*, pages 394–403, 1997.
4. Daniel J. Bernstein. Implementing ”Practical leakage-resilient symmetric cryptography”. CHES '12 rump session, 2012. Available at http://cr.yp.to/talks/2012.09.10/slides.pdf.
5. Alex Biryukov and Dmitry Khovratovich. Two New Techniques of Side-Channel Cryptanalysis. In Paillier and Verbauwhede [25], pages 195–208.
6. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In *CRYPTO*, pages 398–412, 1999.
7. Yevgeniy Dodis and Krzysztof Pietrzak. Leakage-Resilient Pseudorandom Functions and Side-Channel Attacks on Feistel Networks. In *CRYPTO*, pages 21–40, 2010.
8. Stefan Dziembowski and Krzysztof Pietrzak. Leakage-Resilient Cryptography. In *FOCS*, pages 293–302, 2008.
9. Sebastian Faust, Krzysztof Pietrzak, and Joachim Schipper. Practical Leakage-Resilient Symmetric Cryptography. In *CHES*, pages 213–232, 2012.
10. Benoît Gérard and François-Xavier Standaert. Unified and Optimized Linear Collision Attacks and Their Application in a Non-profiled Setting. In Prouff and Schaumont [29], pages 175–192.
11. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.
12. Louis Goubin and Jacques Patarin. DES and Differential Power Analysis (The ”Duplication” Method). In *CHES*, pages 158–172, 1999.
13. Carmit Hazay, Adriana Lopez-Alt, Hoeteck Wee, and Daniel Wichs. Leakage-Resilient Cryptography from Minimal Assumptions. Cryptology ePrint Archive, Report 2012/604, 2012. http://eprint.iacr.org/, accepted at Eurocrypt 2013.
14. Russell Impagliazzo. A Personal View of Average-Case Complexity. In *Structure in Complexity Theory Conference*, pages 134–147, 1995.
15. Yuval Ishai, Amit Sahai, and David Wagner. Private Circuits: Securing Hardware against Probing Attacks. In *CRYPTO*, pages 463–481, 2003.
16. Joshua Jaffe. A first-order dpa attack against aes in counter mode with unknown initial counter. In Paillier and Verbauwhede [25], pages 1–13.
17. Paul C. Kocher. Leak-resistant cryptographic indexed key update. Patent, 03 2003. US 6539092.
18. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *CRYPTO*, pages 388–397, 1999.

19. Marcel Medwed, François-Xavier Standaert, Johann Großschädl, and Francesco Regazzoni. Fresh Re-keying: Security against Side-Channel and Fault Attacks for Low-Cost Devices. In *AFRICACRYPT*, pages 279–296, 2010.
20. Marcel Medwed, François-Xavier Standaert, and Antoine Joux. Towards Super-Exponential Side-Channel Security with Efficient Leakage-Resilient PRFs. In Prouff and Schaumont [29], pages 193–212.
21. Thomas S. Messerges. Using Second-Order Power Analysis to Attack DPA Resistant Software. In *CHES*, pages 238–251, 2000.
22. Silvio Micali and Leonid Reyzin. Physically Observable Cryptography (Extended Abstract). In *TCC*, pages 278–296, 2004.
23. Amir Moradi. Statistical Tools Flavor Side-Channel Collision Attacks. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 428–445. Springer, 2012.
24. Amir Moradi, Oliver Mischke, and Thomas Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In Stefan Mangard and François-Xavier Standaert, editors, *CHES*, volume 6225 of *Lecture Notes in Computer Science*, pages 125–139. Springer, 2010.
25. Pascal Paillier and Ingrid Verbauwhede, editors. *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*. Springer, 2007.
26. Krzysztof Pietrzak. Composition implies adaptive security in minicrypt. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 328–338. Springer, 2006.
27. Krzysztof Pietrzak. A Leakage-Resilient Mode of Operation. In *EUROCRYPT*, pages 462–482, 2009.
28. Krzysztof Pietrzak and Johan Sjödin. Weak pseudorandom functions in minicrypt. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 423–436. Springer, 2008.
29. Emmanuel Prouff and Patrick Schaumont, editors. *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *Lecture Notes in Computer Science*. Springer, 2012.
30. William Pugh. Skip Lists: A Probabilistic Alternative to Balanced Trees. In *WADS*, pages 437–449, 1989.
31. Matthieu Rivain and Emmanuel Prouff. Provably Secure Higher-Order Masking of AES. In *CHES*, pages 413–427, 2010.
32. Kai Schramm, Gregor Leander, Patrick Felke, and Christof Paar. A Collision-Attack on AES: Combining Side Channel- and Differential-Attack. In Marc Joye and Jean-Jacques Quisquater, editors, *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 163–175. Springer, 2004.
33. Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004:332, 2004.
34. François-Xavier Standaert, Olivier Pereira, Yu Yu, Jean-Jacques Quisquater, Moti Yung, and Elisabeth Oswald. Leakage Resilient Cryptography in Practice. *Towards Hardware-Intrinsic Security, Information Security and Cryptography*, pages 99–134, 2010.
35. Yu Yu and François-Xavier Standaert. Practical Leakage-Resilient Pseudorandom Objects with Minimum Public Randomness. In *CT-RSA*, 2013.