Towards Side-Channel Resistant Block Cipher Usage or Can We Encrypt Without Side-Channel Countermeasures?

Jorge Guajardo[1] and Bart Mennink*,[2]

[1] Philips Research Europe, Eindhoven, The Netherlands
jorge.guajardo@philips.com
[2] ESAT/COSIC, Katholieke Universiteit Leuven, Belgium
bart.mennink@esat.kuleuven.be

Abstract. Based on re-keying techniques by Abdalla, Bellare, and Borst [1, 2], we consider two black-box secure block cipher based symmetric encryption schemes, which we prove secure in the physically observable cryptography model. They are proven side-channel secure against a strong type of adversary that can adaptively choose the leakage function as long as the leaked information is bounded. It turns out that our simple construction is side-channel secure against *all* types of attacks that satisfy some reasonable assumptions. In particular, the security turns out to be negligible in the block cipher's block size n, for all attacks. We also show that our ideas result in an interesting alternative to the implementation of block ciphers using different logic styles or masking countermeasures.

1 Introduction

Starting in 1996, a new breed of threats against secure systems appeared in the cryptographic community. These threats received the name of side-channel attacks and include attacks that have taken advantage of the timing [3], power consumption [4], and even the electromagnetic radiation [5] emitted by cryptographic operations performed by physical devices. As a result, numerous algorithmic measures [6, 7] as well as new logic families [8, 9] were developed to counteract such attacks. However, it was apparent that (as it is usually the case in cryptography) this was a cat-and-mouse game in which engineers developed solutions and their colleagues found ways to break them. In fact, it has been clear for a while that a more formal manner to analyze the security of side-channels and associated countermeasures needed to be developed. Micali and Reyzin [10] are the first to try to formalize the treatment of side-channels and introduce the model of physically observable cryptography in which attackers can take advantage of the information leaked during the physical execution of an algorithm in a physical device. Subsequent works, most prominently the work of Standaert et al. [11, 12, 13], introduced a model in which countermeasures can be fairly evaluated and compared based on the amount of information leaked, from an information theoretical point of view. With these tools at hand, the cryptographic community has started to observe the development of primitives whose leaked information can be bound and thus proven secure in a formal model. Examples include pseudo-random generators [10, 14] and stream ciphers [15, 16]. Clearly, pseudo-random generators and stream ciphers can be used to encrypt information. Nevertheless, it seems an interesting question to ask whether we can use block ciphers (e.g. AES) in a provable secure manner against side-channel attacks. To our knowledge, there has not been such attempt.

Our Contributions. In this paper, we analyze re-keying techniques studied in [1, 2] from a side-channel security point of view[3] using a formal model. Abdalla and Bellare consider[4] two types of re-keying: *par-*

* Work done while visiting Philips Research Labs.

[3] We are not aware of any other scientific publication describing similar techniques but it is accepted and mentioned in [1] that re-keying techniques were already part of the "folklore" of security measures used in practice in the year 2000, albeit for reasons other than side-channel security.

[4] Borst [2] considers similar techniques as well as specific instantiations of the constructions based on triple DES, so our treatment is general enough.

allel re-keying where all session keys K_i are computed directly from a master secret key K ($K_i = f(K, i)$ for a suitable function f) and *serial* re-keying where each session key K_i is computed based on the previous key ($K_i = f(k_{i-1}, 0)$, with $k_i = f(k_{i-1}, 1)$ for a suitable function f). For both re-keying mechanisms, executing the block cipher $E_{K_i}(\cdot)$ in the electronic-code-book (ECB) mode of operation yields an IND-CPA (indistinguishable under chosen-plaintext attack) symmetric encryption scheme [17]. The security of these constructions is analyzed in the black-box model in [1]. Although [2] has no *formal* security analysis of the proposed constructions, he seems to be the first to observe that changing the key used to encrypt frequently, can be an effective countermeasure against side-channel attacks. In this paper, we show that these constructions can be implemented in a way in which we can bound the amount of information leaked by the implementation and, under reasonable assumptions, we prove that this is secure against side-channel attacks following [11, 14, 13] and Pietrzak and Dziembowski [15, 16]. We emphasize that the re-keying techniques [1, 2] naively implemented cannot hope to be proven secure against the side-channel adversary of the Dziembowski and Pietrzak (DzPz) model [15]. In particular, the DzPz adversary is allowed to compute *any* polynomial-time computable function Λ of the part of the state that is actually accessed during computation (master secret-key K in our case) with the restriction that the output of the function Λ is bounded to $\{0,1\}^\lambda$, where $\lambda \ll |K|$. Clearly, such adversary can always obtain the whole key in $|K|/\lambda$ runs of the [1, 2] schemes if the session keys have to be computed during each session by processing the master secret key K. A key observation in this paper is that we can implement the parallel scheme described in [1, 2] by storing pre-computed keys in memory. This implementation strategy, in turn, allows us to prove (side-channel attack) security as unused keys stored in memory do not leak information in the DzPz model [15]. Furthermore, we show that such a solution would compare favorably in terms of area against the alternatives: masking schemes or modified logic styles. In addition, there is the added advantage of not having to work with logic styles and/or implement algorithmic masking schemes, which are complicated and prone to errors if not carefully implemented[5]. To our knowledge, this is the first (non-trivial) use of a block cipher, provable secure against side-channel attacks (from an information theoretic point of view). Admittedly, our solution is not without drawbacks. The most prominent of them being that we have a limited number of possible encryptions (fixed by the number of stored session keys). For certain applications this is not a problem in fact, as shown next.

Applications. Consider a coupon scheme where a user buys a certain number of tickets[6] to gain access to some service. One can think of season, ski lift or transport tickets. The protocol idea is depicted in Fig. 1: upon buying the card, it is provided with a tuple of session keys $(K_1, \ldots, K_s)$, and if the user wants to use ticket i, he just uses the i-th key. To guarantee that the ticket can only be queried once

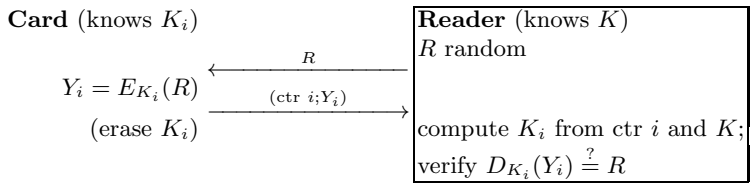

Fig. 1. An abstract application of our protocol. The box indicates a secure environment, where side-channel attacks are not possible.

[5] For example, Mangard et al. [18, 19] showed that in order to minimize the leakage of the masking countermeasure [6, 7], it is necessary to guarantee glitch-free XOR gates.

[6] Traditionally public key cryptography is used for this type of applications, but the problem can also be solved using symmetric cryptography, as noted by Girault et al. [20].

(and not several times to several different readers), one can think of two possible solutions. First, the simplest solution would be to have all readers constantly connected to a single database. The database needs to keep track of the last counter i. If the card makes an authentication request with the same or a lower counter, then access to the service is denied and otherwise the database counter is increased by one. Such solution might work well for season or ski tickets but not for transport tickets where online connection to a database does not seem reasonable to assume. The alternative is to guarantee that the card has a monotonically increasing counter (alternatively a mechanism which erases tuples (i, K_i) once they have been used), which cannot be reset. Notice that we assume that the reader can perform its computations in a secure environment. We believe this to be possible since there are a lot fewer readers and in general, they can be considerably more expensive. Such "assumption" has also been made in [2].

Related Work. Block ciphers and their use to build symmetric encryption schemes have been studied by Bellare et al. in [17]. Liskov et al. [21] formalized[7] the notion of a *tweakable block cipher*, on which our construction is based. Observe that we have abused the idea of a tweakable block cipher since we do not comply with the efficiency requirement put forth in [21]. Other tweakable cipher constructions are considered in [23, 24, 25] but none of them consider the issue of side-channel resistance. Moreover, all of the constructions use the same key in both calls to the block cipher. We have already mentioned the treatment of re-keying techniques by Abdalla and Bellare [1] and Borst [2]. Regarding leakage-proof constructions, we are aware of two parallel lines of research, which originated from the work of Micali and Reyzin [10]. The information theoretic based analysis of Petit et al. [14], who built a pseudo-random generator, and the somewhat more theoretical framework of [15, 16]. These last works focus on the construction of leakage-resilient stream ciphers. Standaert et al. [11, 13] introduced an information theoretical model in which to analyze and compare side-channel countermeasures. Macé et al. [12] have evaluated and compared different logic families using the framework introduced in [13].

Overview. Mathematical preliminaries and definitions are introduced in Sect. 2. A tweakable block cipher construction and its induced symmetric encryption scheme are described and discussed in Sect. 3, and its side-channel security is considered in Sect. 4. Sections 5 and 6 consider a variant of the scheme of Sect. 3 with serial re-keying, and its side-channel security analysis. Section 7 compares our approach to alternative solutions from an area resources point of view, and we end with conclusions in Sect. 8.

2 Preliminaries

Notation. By 'random' we implicitly mean 'uniformly randomly and independently distributed'. Moreover, we denote by $z \in_R \mathcal{Z}$ the event that z is taken uniformly at random from the set $\mathcal{Z}$. For a tuple of m values $(z_1, \ldots, z_m)$ we use the shorthand notation $(z_i)_{i=1}^m$. Adversaries will be denoted by $\mathcal{A}$. The notation $\mathcal{A}^{O(\cdot)}$ means that the adversary $\mathcal{A}$ has the ability to query an oracle O. If $\mathcal{A}$ plays the role of a distinguisher, his task is given O_b to distinguish between two oracles O_0 and O_1 and output a value in $\{0, 1\}$ corresponding to the correct oracle. If $\mathcal{A}$ does not play the role of a distinguisher, his output is arbitrary, and specified by the context. In any case, the adversary is never allowed to query the oracle twice on the same value. In the context of leakage functions, the notation $\mathbb{F}_{n,m}$ will be used, which is defined to be the set of polynomial time computable functions $f : \{0, 1\}^n \to \{0, 1\}^m$.

Symmetric primitives. A *block cipher* is a family of maps $E : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$, which on input of a *key* $K \in \mathcal{K}$ and a *message* $M \in \mathcal{M}$, outputs a *ciphertext* $C = E_K(M)$. A block cipher is said to be

[7] The first use of a tweak in combination with a block cipher is due to Schroeppel [22] with his hasty pudding cipher.

secure if for each key $K \in_R \mathcal{K}$, it is hard for an adversary $\mathcal{A}$ to distinguish between E_K and a random permutation Π on $\mathcal{M}$, even allowing q oracle queries (either E_K or Π) and t computation time. More formally, the security of E for any random permutation Π is quantified by [26]:

$$\mathrm{Adv}_E(q,t) = \max_{\mathcal{A};K\in_R\mathcal{K}} \left| Pr(\mathcal{A}^{E_K(\cdot)} = 1) - Pr(\mathcal{A}^{\Pi(\cdot)} = 1) \right|, \tag{1}$$

and E is said to be *secure* if Adv_E is negligible in the length of the key. It is called *ideal* if $\mathrm{Adv}_E(q,t) = 0$, meaning that E is a random permutation. A *tweakable block cipher* [21] is a family of algorithms $\bar{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$. Particularly, in addition to the ordinary inputs, $\bar{E}$ requires a tweak $T \in \mathcal{T}$. The idea is to hide the deterministic character of the block cipher at the block-cipher level rather than at the modes-of-operation level. In [21], it is stated that tweak renewal should be cheaper than key renewal. In this paper, we will relax this requirement and consider our construction as a tweakable block cipher[8]. A *cryptographic hash function* is a function $h : \{0,1\}^* \to \{0,1\}^n$ that maps messages of arbitrary length to a digest of fixed length n. For simplicity, we assume that h is modeled as a random oracle, hence h is seen as a publicly accessible random function. For a $t \in \mathbb{N}$ and a $y \in \{0,1\}^*$, we define $h^t(y)$ to be the value obtained after applying the hash function t times iteratively.

Symmetric encryption schemes. Symmetric encryption schemes and their security were studied by Bellare et al. [17]. We follow their definitions. A symmetric encryption scheme is a tuple of algorithms $(\mathcal{K}_e, \mathcal{E}, \mathcal{D})$, where $\mathcal{K}_e$ is randomized and $\mathcal{E}$ is either randomized or stateful (updating its state during each execution). For a randomly chosen key $K \in_R \mathcal{K}_e(k)$ where k is the security parameter (usual the key size), and on input of a message M, $\mathcal{E}$ computes ciphertext $C = \mathcal{E}_K(M)$. Under the same key the decryption function gives $\mathcal{D}_K(C) = \mathcal{D}_K(\mathcal{E}_K(M)) = M$.

3 Parallel Re-keying Based Block Cipher and Encryption Scheme

We will introduce the first symmetric encryption scheme which we will prove secure against side-channel attacks in Sect. 4. This scheme corresponds to the parallel re-keying scheme in [1, 2] and the black-box security holds similarly. However, to facilitate the proof of side-channel security, we consider the scheme from a tweakable block cipher point of view. We have included an alternative black-box security proof for the encryption scheme in App. A. We will only consider the case where $\mathcal{K} = \mathcal{T} = \mathcal{M} = \{0,1\}^n$. The tweakable block cipher $\bar{E}$ is defined as $\bar{E} : (K,T,M) \mapsto E_{E_K(T)}(M)$, where E is a block cipher. The scheme is depicted in Fig. 2. In our particular application, the tweak T functions as a counter. Executing the tweakable block cipher $\bar{E}$ of Fig. 2 in ECB mode results in a secure symmetric cipher

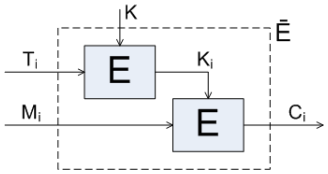

Fig. 2. The tweakable block cipher.

[17]. More formally, $\bar{E}$ can be used to construct the symmetric cipher $(\mathcal{K}_e, \mathcal{E}, \mathcal{D})$ as follows. The function $\mathcal{K}_e$ chooses a key uniformly at random from $\{0,1\}^n$. From now on, this function will be omitted in the scheme description. The participant encrypting will maintain a counter, initially set to 0. The message

[8] The reason for that is purely convenience. It allows us to use the tweakable block cipher setting to analyze our constructions without introducing new definitions.

and ciphertext are composed of m blocks of length n, and $\mathcal{E}$ and $\mathcal{D}$ work as follows (following Bellare et al.'s notation [17]):

function $\mathcal{E}_K(\mathrm{ctr}, M)$	**function** $\mathcal{D}_K(\mathrm{ctr}, C)$
for $i = 1, \ldots, m$	**for** $i = 1, \ldots, m$
$C_i = \bar{E}_K(\mathrm{ctr} + i, M_i)$	$M_i = \bar{D}_K(\mathrm{ctr} + i, C_i)$
return $(\mathrm{ctr} + m; (\mathrm{ctr}, C_1 \cdots C_m))$	**return** $M_1 \cdots M_m$

Recall that $\bar{E}_K(\mathrm{ctr} + i, M_i) = E_{E_K(\mathrm{ctr}+i)}(M_i)$ by definition. An advantage of the scheme is that encryptions can be done in parallel. Obviously, the counter is not allowed to exceed 2^n. This can be resolved by a master key renewal after 2^n encryptions. As shown in App. A, this encryption scheme is secure [1]. In Sect. 4, side-channel security of the scheme is proven.

4 Side-Channel Security of Parallel Re-keying Symmetric Cipher

In this section, side-channel security of the parallel re-keying scheme of Sect. 3 [1, 2] is proven. We consider a model in which during each execution of $\mathcal{E}_K$ the system might leak information, either about the master key K, a session key K_i, or about the message being encrypted. This leakage is denoted by the function Λ, whose input is the current system state. Before we specify this function, we discuss the assumptions needed to specify our model of physical environment. We will later show that in this model the information leakage is only of negligible value to the adversary.

Assumptions. We state the assumptions needed to facilitate the side-channel security proof. These were also used in [10, 14, 15, 16]. We stress that all of these assumptions are reasonable, as argued in the following:

1. The amount of information leaked per execution of $\bar{E}$ is bounded, say of size at most $\lambda \ll n$ bits. This is an absolute prerequisite. Clearly, if the leakage is $\lambda \approx n$, the adversary can learn either the whole key or a significant part of the key and then use exhaustive search for the rest.
2. Only memory accessed leaks information. This guarantees that session keys not accessed during an encryption operation do not leak.
3. The messages are assumed to provide at least $H_\infty(M_i) \geq p \gg \lambda$ bits of uncertainty and are independently distributed, unless the adversary is explicitly allowed to adaptively choose the M_i's. Also this assumption is reasonable, indeed if an attacker already knows that the message is a bit (e.g. for voting), and $\lambda = 1$, he can simply take as leakage function the least significant bit. Also, if the messages M_i and M_{i+1} are related by *any* polynomial time computable function, for $i = 1, \ldots, t = \lceil p/\lambda \rceil$, the adversary can compute λ uncertain bits of M_{t+1} in the i-th round $(i = 1, \ldots, t)$ [15]. This assumption becomes relevant when we consider the information leaked about the message.

In addition to these assumptions, we assume that the session keys have been pre-computed, and that this is done in a secure environment. Clearly, this is crucial to our scheme, since otherwise the adversary could take advantage of the leakage from the master key K and completely break the system. Moreover, it guarantees that our adversary cannot observe any leakage from K directly but rather, all the leakage he observes is from the K_i's. We will make use of this observation to prove security. We observe that in implementing a system this way, we have an advantage over the alternatives [7, 27, 8, 9], namely *simplicity* and *cost efficiency*. Notice that storing the session keys does not harm the security of the system by assumption 2. For convenience, we will only focus on an adversary eavesdropping *encryption* executions. This is justified by the fact that *decryption* is just the inverse of the encryption function, which moreover we assume to be executed in a secure environment. Also, as implied by Thm. 4, it suffices to consider

messages of block size $m = 1$. This is no problem due to the independent character of the encryption function: anything an adversary can learn from q executions on m blocks, he could have learned from qm executions on 1 block. Now, given the leakage length λ, we consider a side-channel adversary $\mathcal{A}$ that

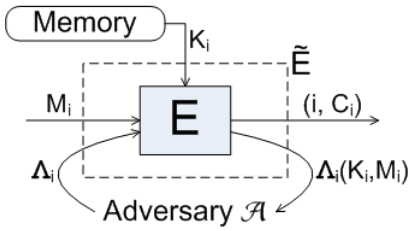

Fig. 3. One execution in the side-channel model. Adversary $\mathcal{A}$ may be allowed to choose M_i.

has t time, and actively eavesdrops the physical data of q executions. This 'eavesdropping' is formalized as follows. Recall that $\mathbb{F}_{n,\lambda}$ is the set of polynomial time functions from $\{0,1\}^n$ to $\{0,1\}^\lambda$. Before each encryption operation i, $\mathcal{A}$ decides on a function $\Lambda_i \in \mathbb{F}_{2n,\lambda}$. The system reads its i-th key K_i from its memory, and on input M_i it computes and publishes $C_i = E_{K_i}(M_i)$. Moreover, the value $\Lambda_i(K_i, M_i)$ is computed (representing the leakage) and sent to the adversary. In some scenarios, $\mathcal{A}$ is allowed to adaptively choose the plaintexts M_i, in which case the function Λ_i just works on $\{0,1\}^n$ and the value $\Lambda_i(K_i)$ is computed instead. The model is visualized in Fig. 3. From now on, the counter i in the output will be implicit. The scheme will be denoted by $\tilde{E} : \{0,1\}^n \times \mathbb{F}_{2n,\lambda} \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^\lambda$ defined as:

$$\tilde{E} : (M_i, \Lambda_i; K_i) \mapsto (E_{K_i}(M_i), \Lambda_i(K_i, M_i)).$$

It is clear that the above assumptions justify this representation for the encryption scheme. Indeed, the session keys K_i are pre-computed, so they are stored in memory, and we implemented the leakage in the broadest possible way: it can be any function, as long as its range is included by $\{0,1\}^\lambda$ (similar to [15, 16]). In the next sections, we will consider the possible attacks the adversary can carry out. Roughly speaking, we will first study in what sense the master key K and the session keys K_i are secure. This is done in Sect. 4.1, and follows the ideas of Standaert et al. [11, 14, 13]. In Sect. 4.2, we additionally consider in what sense the system leaks information about the M_i's.

4.1 Side-Channel Security Against Key Recovery

As usual, the main target of the adversary is to recover the master key K, which would imply a complete break of the system. It means that we need to be sure that the master key cannot be recovered using the leakages, but this is trivial as the keys are pre-computed outcomes of an encryption function. The security of this function also implies that without loss of generality we can assume that the K_i's are uniformly randomly distributed. Secondly, we also need that the session keys K_i are secure, i.e., the leakages do not suffice to recover the session keys K_i. Indeed, if the adversary can manage to recover the session key K_i, he would be able to decrypt C_i. Afterward, we exemplify the results using a particular type of leakage function, namely the well-known Hamming distance. We will consider the strongest possible scenario, that is: the adversary can adaptively choose the messages and in each round he obtains a tuple $(C_i, \Lambda_i(K_i))_{i=1}^q$ for adaptively chosen leakage functions $(\Lambda_i)_{i=1}^q$.

Session Key Security. The question is, what $\Lambda_i(K_i)$ tells an adversary about K_i, for $i = 1, \ldots, q$. As the session keys are independently distributed, we only need to consider what the adversary learns in

that round i. In particular, this can be seen as a 'single query key recovery attack', where the adversary gets only one shot to recover the key. Notice that Standaert et al. [11, 13] have considered this type of attack as well. Moreover, they argue for the use of two different metrics when analyzing a side-channel attack: (i) an information theoretic metric (in order to measure the maximum total leakage) and (ii) a security metric (to analyze the odds of an adversary). As information theoretic metric they advise to use the *mutual information* between the key and its leakage: $I(K;\Lambda) = H(K) - H(K|\Lambda)$, where H is an entropy function. Preferably, H should be the Shannon entropy, rather than the min-entropy, but in case of single query key recovery attacks these are equivalent [13, Sect. 7.3]. For the security metric, they introduce two variants. Following [11, 14], we will consider a so-called 'Bayesian adversary', that selects the most likely key candidate given the leakage result. More formally, he outputs a guessed key[9] $K_i^* = \arg\max_k Pr(K_i = k \mid \Lambda_i(K_i))$. Now, the 'single query success rate' of recovering K_i is defined to be[10]:

$$\mathrm{SR}_{\tilde{E}}^{K_i} = \sum_{l;|d_l|\neq 0} Pr(\Lambda_i(K_i) = l)\frac{1}{|d_l|}, \text{ with } d_l = \{k \mid \Lambda_i(k) = l\}. \tag{2}$$

Indeed, if the leakage function equals l, the Bayesian adversary considers all key candidates that satisfy $\Lambda_i(k) = l$, hence collects the most likely key candidates, and will then take a value from this set. Intuitively, the success rate of recovering K_i is the expectation of the success taken over all possible leakage values l. Using this framework, we prove session key security. We only need to prove that the success rate is sufficiently small, as the mutual information between the key and its leakage is $\leq \lambda$ by definition of Λ_i. W.l.o.g. we only prove this for $i = 1$.

Theorem 1 (Session key security (parallel re-keying)). *Let $\mathcal{A}$ be any Bayesian adversary that can query a single tuple $(M_1, \Lambda_1) \in \{0,1\}^n \times \mathbb{F}_{n,\lambda}$ to his oracle $\tilde{E}(\cdot,\cdot;K_1)$. Then, the success rate of recovering K_1, $SR_{\tilde{E}}^{K_1}$ from eqn. (2), is negligible in n for fixed λ.*

Proof. For the quantity expressed in (2), we have

$$\mathrm{SR}_{\tilde{E}}^{K_1} = \sum_{\substack{l\in\{0,1\}^\lambda \\ |d_l|\neq 0}} Pr(\Lambda_1(K_1) = l)\frac{1}{|d_l|} = \sum_{\substack{l\in\{0,1\}^\lambda \\ |d_l|\neq 0}} Pr(K_1 \in d_l)\frac{1}{|d_l|} = \sum_{\substack{l\in\{0,1\}^\lambda \\ |d_l|\neq 0}} \frac{|d_l|}{2^n}\cdot\frac{1}{|d_l|},$$

where the second equality is by definition of d_l, and the third since $K_1 \in_R \{0,1\}^n$. This value is clearly $\leq \frac{1}{2^{n-\lambda}}$, so is negligible in n (for our λ). □

Example 1. As a way of example, we investigate what the above results mean in a concrete scenario. We consider the case where the leakage function is the Hamming weight, following [11]. Then, the outcome of the leakage function $\Lambda_i(K_i)$ is $\in \{0,\ldots,n\}$. Clearly, we need $\lambda \geq \lg(n+1)$ as is assumed henceforth. Let us re-investigate $\mathrm{SR}_{\tilde{E}}^{K_1}$ from the proof of Thm. 1. This time, $\mathrm{SR}_{\tilde{E}}^{K_1}$ equals:

$$\mathrm{SR}_{\tilde{E}}^{K_1} = \sum_{\substack{l\in\{0,1\}^\lambda \\ |d_l|\neq 0}} \frac{1}{2^n} = \frac{|\{l \in \{0,1\}^\lambda \mid |d_l| \neq 0\}|}{2^n}.$$

But as $|d_l| \neq 0$ only holds for $l \in \{0,\ldots,n\}$, it follows that $\mathrm{SR}_{\tilde{E}}^{K_1} = \frac{n+1}{2^n}$ (which is exactly the same result as in [11, Sect. 3.1]).

[9] Note that K_i^* need not be unique. In that case, he just randomly selects one.

[10] Observe that according to [11, 14], this quantity would be $\sum_l \sum_k Pr(\Lambda_i(K_i) = l \mid K_i = k)Pr(K_i = k)\frac{1}{|d_l|}$, which clearly equals (2) by the law of total probability.

4.2 Side-Channel Security of the plaintext values

In Sect. 4.1 we proved that the master key as well as the session keys are protected sufficiently against side-channel attacks. But an adversary can try to do more! In particular, the adversary can try to recover (a significant part of) a message M_i given that he has been provided with the leakage $\Lambda_i(K_i, M_i)$ and the corresponding ciphertext C_i. Security against this attack will be proven using the same ideas as in the proof of Thm. 1. In the proof, we make use of the fact that the leakage values as well as the different encryptions are independent. This is due to the fact that the values (K_i, M_i, K_j, M_j) are assumed to be mutually independently distributed, and Λ_i as well as $\tilde{E}$ only depend on these values. More formally, for any i the target is to recover M_i given C_i and $\Lambda_i(K_i, M_i)$. As the leakages are independent, it follows that we only have to consider it for one i, which shows the similarities with the 'single query key recovery attack', but now subject to the message. Also now, the mutual information between the message and its leakage is $\leq \lambda$ by definition of Λ_i. Without loss of generality, as the attacker's target is to recover M_i, we assume that $\Lambda_i(K_i, M_i) =: \Lambda_i(M_i)$ is a function in M_i only. As previously, the success rate is:

$$\mathrm{SR}_{\tilde{E}}^{M_i} = \sum_{l;|d_l|\neq 0} Pr(\Lambda_i(M_i) = l)\frac{1}{|d_l|}, \text{ with } d_l = \{m \mid \Lambda_i(m) = l\}. \tag{3}$$

Similarly to Thm. 1, security can be proven, using (3). However, it is trivial that we cannot obtain the same bound, as the message may have entropy $p \leq n$ (ass. 3). Security turns out to be negligible in p instead. In case p is polynomial in n, e.g. $p = n/2$, we have negligibility in n anyway.

Theorem 2 (Plaintext security). *Let $\mathcal{A}$ be any Bayesian adversary that can query a single function $\Lambda_1 \in \mathbb{F}_{n,\lambda}$ to his oracle $\tilde{E}(M_1, \cdot; K_1)$, for any M_1 such that $H_\infty(M_1) \geq p$. Then, the success rate of recovering M_1, $SR_{\tilde{E}}^{M_1}$ from eqn. (3), is negligible in p for fixed λ.*

Proof. By the law of total probability, $Pr(\Lambda_1(M_1) = l)$ can be written as

$$\sum_{m\in\{0,1\}^n} Pr(\Lambda_1(M_1) = l \mid M_1 = m)Pr(M_1 = m) \leq \frac{1}{2^p}\sum_{m\in\{0,1\}^n} Pr(\Lambda_1(M_1) = l \mid M_1 = m) = \frac{|d_l|}{2^p},$$

where the inequality holds as $-\log_2 \max_m Pr(M_i = m) \geq p$ by the definition of H_∞, and by the definition of d_l. The final result $\mathrm{SR}_{\tilde{E}}^{M_1} \leq \frac{1}{2^{p-\lambda}}$ is now easily obtained. □

5 Serial Re-keying Optimizations

The scheme of Sect. 3 is simple and efficient, but requires the session keys to be pre-computed and stored in memory. By using serial re-keying this problem can be solved as shown in [1, 2]. The idea is that now each session key K_i is computed from the previous session key as $K_i = h(K_{i-1})$, where $K_0 := K$ is the master key. However, this approach fails in the model of [15] as an adversary can perform a side-channel attack on the session keys to obtain a future key [15]: the adversary can take $\Lambda_i(\cdot) = h^{t-i+1}(\cdot)_{[(i-1)\lambda,\ldots,i\lambda-1]}$ for $i = 1, \ldots, t$, where $t = \lceil n/\lambda \rceil$. This results in the adversary learning K_{t+1} after round t, and all future encryptions are broken (note that the scheme is still forward secure as h is one-way). By hashing a counter r (initially 0) along the master key and re-initializing the hash tree after α number of iterations (for a fixed $\alpha \in \mathbb{N}$), this attack can be prevented. This results in the following key scheduling.

$$K_i = \begin{cases} h(K\|r), \text{ if } i = r\alpha + 1 \text{ for some } r \in \mathbb{N}, \\ h(K_{i-1}), \text{ otherwise.} \end{cases} \tag{4}$$

The resulting tweakable block cipher $\bar{G}$ is depicted in Fig. 4. Its security proof follows a similar argument as Prop. 1, and relies on the security of the hash function. Also this tweakable block cipher results in a symmetric encryption scheme in the ECB mode. We end this section by noticing that the key derivation

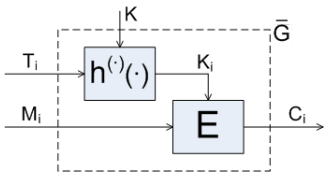

Fig. 4. The tweakable block cipher with serial re-keying. In our application, T_i functions as a counter.

function does not necessarily have to be a hash function. As shown in [1, 2], there is a variety of possible functions which comply with the basic requirement of one-wayness. Our choice of hash function just makes the security analysis somewhat simpler.

6 Side-Channel Security of Serial Re-keying Cipher

The scheme of Sect. 5 can be proven secure using the assumptions (model) introduced in Sect. 4. In particular, assumptions 1-3 still apply. For this scheme, we require that the session keys $K_{r\alpha+1} = h(K\|r)$ for $r = 0, 1, \ldots$ are pre-computed in a secure environment, to prevent side-channel attacks on K. Again, the scheme leaks K_i and M_i, but it can be the case that the adversary can adaptively choose the M_i. In full generality, the side-channel model can be captured by $\tilde{G} : \{0,1\}^n \times \mathbb{F}_{2n,\lambda} \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^\lambda$ defined as:

$$\tilde{G} : (M_i, \Lambda_i; K_i) \mapsto (E_{K_i}(M_i), \Lambda_i(K_i, M_i)).$$

In fact, $\tilde{G}$ also outputs K_{i+1}, but as this is output to the buffer only, it is omitted for simplicity. The model is depicted in Fig. 5. An important observation is that now the session keys are related; this gives the adversary more power. As a consequence also the proof of side-channel security differs. As the security proof is similar to the proof in Sect. 4, we only point out the differences. In particular, the proof of plaintext security is exactly the same as before, and we only need to consider security against key recovery.

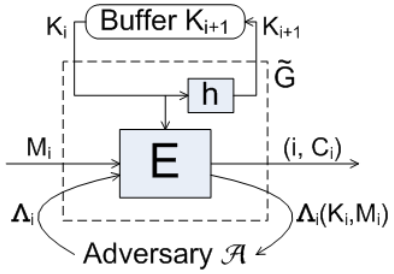

Fig. 5. One execution in the side-channel model with serial re-keying. Adversary $\mathcal{A}$ may be allowed to choose M_i.

6.1 Side-Channel Security against Key Recovery

Similarly to the scheme of Sect. 3, the master key cannot be recovered as it is never used during actual computation (i.e., the session keys K_i have been pre-computed and stored in memory). In particular, the

master key K only leaks via the session keys K_i, which computationally hide K as h is a one-way function. For the security of the session keys, an important observation is that these are related via a polynomial computable function, which gives the adversary more power. However, the sets $\{\{K_{r\alpha+1}, \ldots, K_{(r+1)\alpha}\} \mid r = 0, 1, \ldots\}$ can be considered to be mutually independent, due to the assumed one-wayness of h. Therefore, we only need to consider one such set, which w.l.o.g. is $\{K_1, \ldots, K_\alpha\}$ (for $r = 0$). Without loss of generality, the adversary can query $q = \alpha$ times[11]. The proof follows a similar argument as the proof of Theorem 1, i.e. the maximum success rate of guessing a key K_i ($i = 1, \ldots, \alpha$) is determined. This success rate is denoted by $\mathrm{SR}_{\tilde{G}}^{K_i}$. If an adversary can guess K_i correctly for some $i \leq \alpha$, then he also obtains K_α correctly as $K_\alpha = h^{\alpha-i}(K_i)$. Therefore, $\mathrm{SR}_{\tilde{G}}^{K_i} \leq \mathrm{SR}_{\tilde{G}}^{K_\alpha}$ holds for all i, and it suffices to consider $\mathrm{SR}_{\tilde{G}}^{K_\alpha}$ only. The adversary queries the system α times, each time on a (possibly different) leakage function Λ_i (for $i = 1, \ldots, \alpha$), and he obtains the values $(\Lambda_i(K_i))_{i=1}^{\alpha}$. The adversary wants to guess K_α. Now, $\mathcal{A}$'s output is the key $K_\alpha^* = \arg\max_k Pr(K_\alpha = k \mid (\Lambda_i(K_i))_{i=1}^{\alpha})$. We want to define $d_{\mathbf{l}}$, the most likely values to be K_α. Define

$$e_{\mathbf{l}} = \{(K_i)_{i=1}^{\alpha} \mid (\Lambda_i(K_i))_{i=1}^{\alpha} = \mathbf{l}\}.$$

Clearly, these are all possible key sequences $(K_i)_{i=1}^{\alpha}$ that could end up with leakage sequence $\mathbf{l}$, but our aim is to find the set of most likely key candidates for K_α. As by the one-wayness of h, two different sequences ending with the same α-th component would imply a collision, we observe that all K_α's in $e_{\mathbf{l}}$ are different. In other words, we obtain:

$$d_{\mathbf{l}} = \{K_\alpha \mid \text{there exists } (K_i)_{i=1}^{\alpha} \text{ such that } (K_i)_{i=1}^{\alpha} \in e_{\mathbf{l}}\},$$

and again the probability of a correct guess given $\mathbf{l}$ equals $1/|d_{\mathbf{l}}|$. So the success rate of recovering K_α is, similar to (2):

$$\mathrm{SR}_{\tilde{G}}^{K_\alpha} = \sum_{\mathbf{l};|d_{\mathbf{l}}|\neq 0} Pr((\Lambda_i(K_i))_{i=1}^{\alpha} = \mathbf{l}) \frac{1}{|d_{\mathbf{l}}|}. \tag{5}$$

Theorem 3 (Session key security (serial re-keying)). *Let $\mathcal{A}$ be any Bayesian adversary that can query α tuples $(M_i, \Lambda_i) \in \{0,1\}^n \times \mathbb{F}_{n,\lambda}$ to his oracle $\tilde{G}(\cdot, \cdot; K_i)$. Then, the success rate of recovering K_α, $SR_{\tilde{G}}^{K_\alpha}$ from eqn. (5), is negligible in n for fixed λ and α.*

Proof. For the quantity expressed in (5), we have

$$\begin{aligned}
\mathrm{SR}_{\tilde{G}}^{K_\alpha} &= \sum_{\substack{\mathbf{l}\in\{0,1\}^{\alpha\lambda} \\ |d_{\mathbf{l}}|\neq 0}} Pr((\Lambda_i(K_i))_{i=1}^{\alpha} = \mathbf{l}) \frac{1}{|d_{\mathbf{l}}|} \\
&= \sum_{\substack{\mathbf{l}\in\{0,1\}^{\alpha\lambda} \\ |d_{\mathbf{l}}|\neq 0}} \sum_{k\in\{0,1\}^n} Pr((\Lambda_i(K_i))_{i=1}^{\alpha} = \mathbf{l} \mid K_1 = k) Pr(K_1 = k) \frac{1}{|d_{\mathbf{l}}|} && \{\text{law of total probability}\} \\
&= \sum_{\substack{\mathbf{l}\in\{0,1\}^{\alpha\lambda} \\ |d_{\mathbf{l}}|\neq 0}} \frac{1}{|d_{\mathbf{l}}| 2^n} \sum_{k\in\{0,1\}^n} Pr((\Lambda_i(K_i))_{i=1}^{\alpha} = \mathbf{l} \mid K_1 = k) && \{\text{generalisation}\} \\
&= \sum_{\substack{\mathbf{l}\in\{0,1\}^{\alpha\lambda} \\ |d_{\mathbf{l}}|\neq 0}} \frac{1}{2^n}. && \{\text{definition of } d_{\mathbf{l}}\}
\end{aligned}$$

[11] If $q > \alpha$, the adversary is working with independent sets, which is useless for an adversary focussing on a session key recovery attack. If $q < \alpha$ he simply has a smaller chance to success.

This value is clearly $\leq \frac{1}{2^{n-\alpha\lambda}}$, so is negligible in n (for our α and λ). □

The proof illustrates the importance of smartly choosing α, the depth of the hash before re-initiation. A similar result can be obtained in the parallel re-keying scheme of Sect. 3 when the session keys are used a multiple number of times.

7 Comparison to Alternative Side-Channel Resistant Methodologies

In this section, we study in which situations the parallel re-keying scheme analysized in this paper is competitive in terms of area costs with respect to other alternative countermeasures. Clearly, the parallel-rekeying technique area cost varies according to the number of session keys stored in memory. Thus, it only makes sense to talk about the area cost of this technique for a *specific* number of pre-computed session keys.

7.1 Is an Area-Cost Comparison the Correct Metric?

Before analyzing area costs, one may ask if this is really an appropriate metric to compare different approaches and implementations. Any comparison of approaches should compare the security level as well as the implementation costs (area, performance, etc.). Our starting point is that we assume that *all* approaches compared in this section offer the same security level. Clearly, this is not true[12]. We ignore this fact having argued and proven the security of our schemes in our model in the previous sections of this paper. In other words, given that the countermeasures analyzed in this paper are secure in our model, the other countermeasures can provide *at most* the same level of security. We believe that this is a rather pesimistic comparison from the point of view of the approach proposed in this paper. However, we believe this to be a good choice and in agreement with the common practice in security to take a pesimistic approach (e.g. very powerful adversaries).

It should be clear now that if security is the same for all considered approaches, then our comparison should be based on typical metrics used for this type of (hardware) implementation: area, performance, power consumption, etc. Of these three metrics, the most interesting one is area. In particular, performance is independent of the (side-channel security) approach and essentially dictated by the architecture of the implementation and the technology. Clearly, there is no reason why our design would not be able to use the most efficient technology and/or architecture available. Furthermore, we propose to implement a *simple* AES with additional memory to store session keys. Thus, the performance achieved by our implementation is the same as that of a plain AES implementation and at least as fast as any other implementation using a different (side-channel security) methodology. We ignore power consumption as there are many other variables that could affect it (technology, particular architecture, number of session keys, amount of memory) but we expect that a plain AES implementation will in general have lower power consumption than the alternative implemented using modifed logic styles (see, e.g., [35] for a discussion) Therefore, area is the metric that is left for comparison and it is what we focus on. We end by noticing that an actual implementation [30] comparing a plain AES and AES using WDDL logic [29] confirms our assumptions: performance is reduced by more than a factor of three and power consumption

[12] For example, Mangard et al. [18, 19] have shown attacks on masked implementations and Macé et al. [12] have analyzed from an information theory point of view different logic styles and shown that the amount of information leaked by different logic styles is not the same and it depends heavily on the amount of noise present in the observations made during an attack. Similarly, an attack has been found in [28] against several types of dual rail logic approaches including WDDL [29].

is increased by almost four times.

We consider the following (alternative) side-channel attack countermeasures: algorithmic solutions as introduced by Blömer et al. [6] and later optimized in [7], as well as low level solutions based on dual rail logic [8, 9].

7.2 The Area-Cost of a Masked AES Implementation

In this section, we estimate the cost of using an AES-based encryption only design which requires side-channel resistance. We use the masked implementation of Canright and Batina [27] for our estimates[13] together with Akkar and Giraud [33], who describe the *overall* architecture of a masked AES implementation. Although, [33] focuses on the software implementation of such scheme, the same ideas can be easily implemented in the hardware domain. We compute the estimate this way because we are not aware of any publication describing the costs of a fully masked AES. In order to implement a masked AES, it is sufficient to duplicate all AES transformations (SubBytes, MixColumns, and ShiftRows). It follows that the cost of implementing masked AES is the cost of the masked SubBytes transformation plus twice the size of the unmasked transformations. We consider a standard datapath with 4 S-boxes where only the S-box (and not the inverse S-box is implemented). Clearly the more performance is required the more S-boxes that are needed in parallel and, as a result, the larger the area requirement. Notice also that no overhead is calculated due to the increased complexity of the datapath. Thus, we can see these estimates as good lower bounds. We use the work of Satoh et al. [34] to estimate the cost of a plain (unmasked) AES implementation. We have ignored the cost of the key schedule in all our computations. Table 1 compares a masked implementation with an unmasked implementation. An alternative way to achieve

Component	**unmasked implementation**		**masked implementation**	
	GE	**Source**	**GE**	**Source**
4 S-boxes	1176	[34]	2436	[27]
Data Register	864	[34]	1728	[34, 33]
ShiftRows	160	[34]	320	[34, 33]
MixColumns	273	[34]	546	[34, 33]
AddRoundKey	56	[34]	112	[34, 33]
Total	2529	—	5142	

Table 1. Actual costs of an unmasked AES according to [34] and estimated costs of a masked AES implementation based on [27] and [33]. We ignore the selector, the key scheduled and the overhead of supporting inverse cipher. GE: Gate Equivalents

DPA resistance is to develop dual-rail logic families. The area overhead of different dual-rail logic families has been considered in [8] and ranges from at least twice to as much as 12 times as large, depending on the specific logic family (see [8, 9, 35] for comparisons). Table 2 provides a summary of the area-cost of different AES implementations. Table 2 reports two logic families. We have not included SecLib [36, 9] because it has slightly worse area requirements than WDDL.

[13] The implementation of [27] is the smallest mask implementation of an S-box that we are aware of but it has been found to be flawed in [31]. Canright remarks in [32] that the cost will be significantly higher than predicted. Thus, this is a good "lower bound" on the cost of a masked implementation.

Approach	**Source**	% overhead logic (GE)	% overhead Flip-Flops (GE)	**Total Area Cost (GE)**	**% Total overhead**
plain	[34]	0%(1665)	0%(864)	2529	0%
masked	Table 1	—	—	5142	103 %
dual-rail	[8]	108% (3464)	228% (2851)	6315	150 %
WDDL	[29]	100% (3330)	300% (3456)	6786	168 %

Table 2. Estimated costs of a 4-Sbox AES implementation with different technologies. Key scheduled and the overhead of supporting inverse cipher are not considered. GE: Gate Equivalents.

7.3 Comparison

Notice that implementing the parallel re-keying scheme, we need to implement a simple unmasked AES implementation and add storage for the session keys K_i. In our hypothetical implementation, we just need to store the session keys (no need to erase them) together with a small counter, whose area cost we ignore (probably a 6-bit counter will do for our applications). We parametrize the cost of non-volatile memory via the number of bits needed to store s session keys of size 128-bits. This allows one to choose the cost function that best fits the user application and technology used.
Table 1 compares a masked implementation with an unmasked implementation. It is easy to see that for an implementation with 4 S-boxes we have about 100% overhead with a masked implementation. This means we have the area equivalent to a whole AES module for use as session key storage. For example, assume that in our particular standard cell library, 1 non-volatile memory cell (this could be ROM, Flash, etc.) occupies an area equivalent to one NAND equivalent gate. Here we are being conservative as a simple ROM cell requires only one transistor, whereas a NAND gate requires four. Then, we would have storage for $(5142 - 2529)/(128) = 20$ session keys. This clearly shows that the scheme analyzed in this paper is competitive with the masked methodologies and even more so with modified logic styles such as those proposed in [8, 29, 36].
It should be clear from the previous discussion that the parallel re-keying technique can be implemented in a manner that is competitive with alternative side-channel countermeasures in terms of area. As we mentioned in the previous section, performance will certainly be better and power consumption (probably) as well. Finally, other advantages include simpler designs, possibility to use standard design flows. This comparison seems to indicate that in practice side-channel attacks could be solved by countermeasures designed at the protocol level rather than at the low level implementation level as extensively studied until now.

8 Conclusions

We considered the re-keying techniques described in [1, 2], and proved them secure in the side-channel model following the models introduced in [14, 15]. A drawback of the scheme is that it only offers possibility for a limited number of encryptions, but as shown in Sect. 7, it compares favorably against alternatives like masking schemes or modified logic styles.

We end by pointing out that given that side-channel countermeasures have to be put in place to guarantee security of an implementation, it is fair to ask (as does Borst [2]) what is the cheapest (in whatever metric) way to deploy such countermeasures: changing the hardware seems very costly, changes in software can be costly as they usually result in large code sizes, changing the encryption algorithm requires years to deploy (the algorithms have to be thoroughly studied by experts and the whole infrastructure needs to be updated to support the new algorithm). The re-keying techniques presented in [1, 2] can be seen as countermeasures at the protocol level. As such, they do not require a change of algorithms or hardware or

infrastructure. They involve changes in how keys are managed using standard (well accepted) algorithms. Thus, they appear to be very appealing from a practical point of view [2].

References

[1] Abdalla, M., Bellare, M.: Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-keying Techniques. In Okamoto, T., ed.: Advances in Cryptology — ASIACRYPT '00. Volume 1976 of LNCS., Springer (December 3-7, 2000) 546–559

[2] Borst, J.: Block Ciphers: Design, Analysis, and Side-channel Analysis. PhD thesis, Departement Elektrotechniek — ESAT/COSIC, Katholieke Universiteit Leuven (September 2001) Available at `http://www.cosic.esat.kuleuven.be/publications/`.

[3] Kocher, P.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Koblitz, N., ed.: Advances in Cryptology — CRYPTO '96. Volume 1109 of LNCS., Springer (August 18-22, 1996) 104–113

[4] Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In Wiener, M., ed.: Advances in Cryptology — CRYPTO '99. Volume 1666 of LNCS., Springer (August 15-19, 1999) 388–397

[5] Agrawal, D., Archambeault, B., Rao, J., Rohatgi, P.: The EM Side–Channel(s). In Kaliski Jr., B., Ç. Koç, Paar, C., eds.: Cryptographic Hardware and Embedded Systems — CHES '02. Volume 2523 of LNCS., Springer (August 13-15, 2003) 29–45

[6] Blömer, J., Guajardo, J., Krummel, V.: Provably Secure Masking of AES. In Handschuh, H., Anwar Hasan, M., eds.: Selected Areas in Cryptography — SAC '04. Volume 3357 of LNCS., Springer (August 9-10, 2004) 69–83

[7] Oswald, E., Mangard, S., Pramstaller, N., Rijmen, V.: A Side-Channel Analysis Resistant Description of the AES S-Box. In Gilbert, H., Handschuh, H., eds.: Fast Software Encryption — FSE '05. Volume 3557 of LNCS., Springer (February 21-23, 2005) 413–423

[8] Sokolov, D., Murphy, J., Bystrov, A., Yakovlev, A.: Design and Analysis of Dual-Rail Circuits for Security Applications. IEEE Trans. Computers **54**(4) (2005) 449–460

[9] Guilley, S., Sauvage, L., Hoogvorst, P., Pacalet, R., Bertoni, G., Chaudhuri, S.: Security Evaluation of WDDL and SecLib Countermeasures against Power Attacks. IEEE Transactions on Computers **57**(11) (2008)

[10] Micali, S., Reyzin, L.: Physically Observable Cryptography. In Naor, M., ed.: Theory of Cryptography. Volume 2951 of LNCS., Springer (February 19-21, 2004) 278–296

[11] Standaert, F., Peeters, E., Archambeau, C., Quisquater, J.: Towards Security Limits in Side-Channel Attacks (With an Application to Block Ciphers). In Goubin, L., Matsui, M., eds.: Cryptographic Hardware and Embedded Systems — CHES '06. Volume 4249 of LNCS., Springer (October 10-13, 2006) 30–45 Extended version available at `http://eprint.iacr.org/2007/222`.

[12] Macé, F., Standaert, F., Quisquater, J.: Information Theoretic Evaluation of Side-Channel Resistant Logic Styles. In Paillier, P., Verbauwhede, I., eds.: Cryptographic Hardware and Embedded Systems — CHES '07. Volume 4727 of LNCS., Springer (September 10-13, 2007) 427–442

[13] Standaert, F., Malkin, T., Yung, M.: A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In Joux, A., ed.: Advances in Cryptology — EUROCRYPT '09. Volume 5479 of LNCS., Springer (April 26-30, 2009) 443–461 Extended version available at `http://eprint.iacr.org/2006/139`.

[14] Petit, C., Standaert, F., Pereira, O., Malkin, T., Yung, M.: A Block Cipher Based PRNG Secure Against Side-Channel Key Recovery. In Abe, M., Gligor, V., eds.: ACM Symposium on Information, Computer and Communications Security — ASIACCS '08, ACM (March 18-20, 2008) 56–65

[15] Dziembowski, S., Pietrzak, K.: Leakage-Resilient Cryptography. In: FOCS '08. Proceedings of the 49nd IEEE symposium on FOCS, IEEE Computer Society (October 25-28, 2008) 293–302

[16] Pietrzak, K.: A Leakage-Resilient Mode of Operation. In Joux, A., ed.: Advances in Cryptology — EUROCRYPT '09. Volume 5479 of LNCS., Springer (April 26-30, 2009)

[17] Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A Concrete Security Treatment of Symmetric Encryption. In: FOCS '97. Proceedings of the 38nd IEEE symposium on FOCS, IEEE Computer Society (October 20-22, 1997) 394–403 Extended version available at `http://www-cse.ucsd.edu/users/mihir/papers/sym-enc.pdf`.

[18] Mangard, S., Pramstaller, N., Oswald, E.: Successfully Attacking Masked AES Hardware Implementations. In Rao, J., Sunar, B., eds.: Cryptographic Hardware and Embedded Systems — CHES '05. Volume 3659 of LNCS., Springer (August 29 - September 1, 2005) 157–171

[19] Mangard, S., Schramm, K.: Pinpointing the Side-Channel Leakage of Masked AES Hardware Implementations. In Goubin, L., Matsui, M., eds.: Cryptographic Hardware and Embedded Systems — CHES '06. Volume 4249 of LNCS., Springer (October 10-13, 2006) 76–90

[20] Girault, M., Poupard, G., Stern, J.: On the Fly Authentication and Signature Schemes Based on Groups of Unknown Order. J. Cryptology **19**(4) (2006) 463–487

[21] Liskov, M., Rivest, R., Wagner, D.: Tweakable Block Ciphers. In Yung, M., ed.: Advances in Cryptology — CRYPTO '02. Volume 2442 of LNCS., Springer (August 18-22, 2002) 31–46
[22] Schroeppel, R.: The Hasty Pudding Cipher. AES candidate submitted to NIST (1997) http://www.cs.arizona.edu/~rcs/hpc.
[23] Rogaway, P.: Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In Lee, P., ed.: Advances in Cryptology — ASIACRYPT '04. Volume 3329 of LNCS., Springer (December 5-9, 2004) 16–31
[24] Wang, P., Feng, D., Wu, W.: On the Security of Tweakable Modes of Operation: TBC and TAE. In Zhou, J., Lopez, J., Deng, R., Bao, F., eds.: Information Security — ISC '05. Volume 3650 of LNCS., Springer (September 20-23, 2005) 274–287
[25] Chakraborty, D., Sarkar, P.: A General Construction of Tweakable Block Ciphers and Different Modes of Operations. In Lipmaa, H., Yung, M., Lin, D., eds.: Information Security and Cryptology — INSCRYPT '06. Volume 4318 of LNCS., Springer (November 29 - December 1, 2006) 88–102
[26] Bellare, M., Kilian, J., Rogaway, P.: The Security of the Cipher Block Chaining Message Authentication Code. J. Comput. Syst. Sci. **61**(3) (2000) 362–399 Extended abstract in Advances in Cryptology — Crypto 94, volumen 839 of LNCS, Y. Desmedt (Ed.), Springer-Verlag, 1994. Full paper available at http://www.cs.ucsd.edu/~mihir/papers/cbc.html.
[27] Canright, D., Batina, L.: A Very Compact "Perfectly Masked" S-Box for AES. In Bellovin, S., Gennaro, R., Keromytis, A., Yung, M., eds.: Applied Cryptography and Network Security — ACNS '08. Volume 5037 of LNCS., Springer (June 3-6, 2008) 446–459 Corrected extended version available in [31].
[28] Suzuki, D., Saeki, M.: Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style. In Goubin, L., Matsui, M., eds.: Cryptographic Hardware and Embedded Systems - CHES 2006. Volume 4249 of LNCS., Springer (2006) 255–269
[29] Tiri, K., Verbauwhede, I.: A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In: Design, Automation and Test in Europe Conference and Exposition — DATE 2004, IEEE Computer Society (16-20 February 2004) 246–251
[30] Tiri, K., Verbauwhede, I.: A digital design flow for secure integrated circuits. IEEE Trans. on CAD of Integrated Circuits and Systems **25**(7) (2006) 1197–1208
[31] Canright, D., Batina, L.: A Very Compact "Perfectly Masked" S-Box for AES (corrected). Cryptology ePrint Archive, Report 2009/011 (2009) http://eprint.iacr.org/2009/011.
[32] Canright, D.: Avoid Mask Re-use in Masked Galois Multipliers. Cryptology ePrint Archive, Report 2009/012 (2009) http://eprint.iacr.org/2009/012.
[33] Akkar, M., Giraud, C.: An Implementation of DES and AES, Secure against Some Attacks. In Koç, Ç.., Naccache, D., Paar, C., eds.: Cryptographic Hardware and Embedded Systems — CHES '01. Volume 2162 of LNCS., Springer (May 14-16, 2001) 309–318
[34] Satoh, A., Morioka, S., Takano, K., Munetoh, S.: A Compact Rijndael Hardware Architecture with S-Box Optimization. In Boyd, C., ed.: Advances in Cryptology - ASIACRYPT '01. Volume 2248 of LNCS., Springer (December 9-13, 2001) 239–254
[35] Tillich, S., Großschädl, J.: Power Analysis Resistant AES Implementation with Instruction Set Extensions. In Paillier, P., Verbauwhede, I., eds.: Cryptographic Hardware and Embedded Systems — CHES '07. Volume 4727 of LNCS., Springer (September 10-13, 2007) 303–319
[36] Guilley, S., Flament, F., Hoogvorst, P., Pacalet, R., Mathieu, Y.: Secured CAD Back-End Flow for Power-Analysis-Resistant Cryptoprocessors. IEEE Design & Test of Computers **24**(6) (2007) 546–555

A Black-box security proof for $(\mathcal{E}, \mathcal{D})$

Before proving the security of the symmetric encryption scheme of Sect. 3, we introduce the security definitions of tweakable block ciphers and symmetric encryption schemes. Then, we will prove that $\bar{E}$ is a secure tweakable block cipher in Prop. 1. This proof is based on a standard hybrid argument and is similar to the proof of [21, Thm. 1]. Theorem 4 then shows that this implies that the encryption scheme of Sect. 3 is secure.

The security of a tweakable block cipher is defined similarly as for block ciphers [21]. Let $\bar{E}$ be a tweakable block cipher. We define $\bar{\Pi}(\cdot,\cdot)$ to be a family of permutations, such that for each $T \in \mathcal{T}$, $\bar{\Pi}(T,\cdot)$ is a random permutation on $\mathcal{M}$. Now, the security of $\bar{E}$ is quantified by:

$$\mathrm{Adv}_{\bar{E}}(q,t) = \max_{\mathcal{A}, K \in_R \mathcal{K}} \left| Pr(\mathcal{A}^{\bar{E}_K(\cdot,\cdot)} = 1) - Pr(\mathcal{A}^{\bar{\Pi}(\cdot,\cdot)} = 1) \right|, \tag{6}$$

and $\bar{E}$ is said to be secure if $\mathrm{Adv}_{\bar{E}}$ is negligible in the length of the key.
The security of symmetric encryption schemes has been studied by Bellare et al. [17]. We will adopt the 'left-or-right indistinguishability under chosen plaintext attacks' (lor-CPA). Formally, we have an encryption oracle that for a fixed value $b \in \{0,1\}$, on input (M_0, M_1) encrypts M_b. The security statement says that it is hard for an adversary to find out the value b. We define the function $\mathrm{lor} : \mathcal{M}^2 \times \{0,1\} \to \mathcal{M}$, that on input (M_0, M_1, b) with $|M_0| = |M_1|$ outputs M_b. The security of the symmetric encryption scheme is defined by

$$\mathrm{Adv}^{cpa}_{\mathcal{SE}}(k,t,q_e,\mu_e) = \max_{\mathcal{A}} \left| Pr(\mathcal{A}^{\mathcal{E}_K(\mathrm{lor}(\cdot,\cdot,1))} = 1, K \in_R \mathcal{K}_e(k)) - Pr(\mathcal{A}^{\mathcal{E}_K(\mathrm{lor}(\cdot,\cdot,0))} = 1, K \in_R \mathcal{K}_e(k)) \right|,$$

where the adversary has maximum computation time t, making at most q_e oracle queries, totally at most $2\mu_e$ bits. The scheme is called secure if the above quantity is negligible in k.
We are now ready to prove the security of the scheme of Sect. 3.

Proposition 1. *Let E be a secure block cipher with security function Adv_E. Then $\bar{E} : (K,T,M) \mapsto E_{E_K(T)}(M)$ is a secure tweakable block cipher, with security function*

$$\mathrm{Adv}_{\bar{E}}(q,t) \leq \mathrm{Adv}_E(q,t) + \max \sum_i \mathrm{Adv}_E(k_i,t). \tag{7}$$

with the maximum taken over all possible choices $(k_i)_{i=1}^{n_q}$ with $\sum_{i=1}^{n_q} k_i = q$, for some $1 \leq n_q \leq q$.

Proof. Let $\mathrm{Adv}_E(q,t)$ be the security function of E and let $\mathcal{A}$ be an adversary for $\bar{E}$. Let $K \in_R \mathcal{K}$ and $\bar{\Pi}$ a random permutation family, so for each $T \in \{0,1\}^n$, $\bar{\Pi}(T,\cdot)$ is a random permutation on $\{0,1\}^n$. Our target is to limit

$$\left| Pr(\mathcal{A}^{E_{E_K(\cdot)}(\cdot)} = 1) - Pr(\mathcal{A}^{\bar{\Pi}(\cdot,\cdot)} = 1) \right|, \tag{8}$$

which will be done using a hybrid argument.
(i). Let Π be a random permutation on $\mathcal{M}$. Then, if $\mathcal{A}$ can distinguish between $E_{E_K(\cdot)}(\cdot)$ and $E_{\Pi(\cdot)}(\cdot)$, he can distinguish between E_K and Π. So

$$\Xi_0 := \left| Pr(\mathcal{A}^{E_{E_K(\cdot)}(\cdot)} = 1) - Pr(\mathcal{A}^{E_{\Pi(\cdot)}(\cdot)} = 1) \right| \leq \mathrm{Adv}_E(q,t). \tag{9}$$

(ii). We define a new oracle $Z(\cdot,\cdot)$. It maintains a database *db* on tuples (T,M,C), whose initial state is $\{\}$. For a query (T_i, M_i), it operates as follows: if $(T_i, *, *) \notin db$, then just take $C_i \in_R \{0,1\}^n$. Otherwise, take $C_i \in_R \{0,1\}^n$ such that $(T_i, *, C_i) \notin db$. Output C_i and add (T_i, M_i, C_i) to *db*. In other words: 'If T_i was not queried before, just take a new random C_i. Otherwise, simulate the permutation indexed by T_i.' Next, we want to bound $\Xi_1 := \left| Pr(\mathcal{A}^{E_{\Pi(\cdot)}(\cdot)} = 1) - Pr(\mathcal{A}^{Z(\cdot,\cdot)} = 1) \right|$. Notice that provided $T_i \neq T_j$, the oracle outcomes C_i, C_j are mutually independent for both $E_{\Pi(\cdot)}(\cdot)$ (note that different tweak values imply different keys) and $Z(\cdot,\cdot)$ (by construction). Let $(T_i)_{i=1}^q$ be a q-tuple of tweak values queried by $\mathcal{A}$. By independence in case of different tweak values, we can w.l.o.g. assume that

$$(T_1, \ldots, T_q) = (\underbrace{T_{(1)}, \ldots, T_{(1)}}_{k_1}, \ldots, \underbrace{T_{(n_q)}, \ldots, T_{(n_q)}}_{k_{n_q}}),$$

for some n_q, $(k_i)_{i=1}^{n_q}$ and *different* values $(T_{(i)})_{i=1}^{n_q}$. Again by independence, the advantage of the adversary in distinguishing $E_{\Pi(\cdot)}(\cdot)$ from $Z(\cdot,\cdot)$ is at most the sum of the advantages in the 'sessions' $i = 1, \ldots, n_q$. But in one such session, the same tweak is always used, which means that the advantage is exactly

the advantage of distinguishing $E_{\Pi(T_{(i)})}$ from a random permutation $\Pi_{T_{(i)}}$ indexed by $T_{(i)}$. Hence, its advantage is upper bounded by $\mathrm{Adv}_E(k_i, t)$. Maximized over all possible tweak value tuples, we conclude:

$$\Xi_1 \leq \max\left\{\sum_{i=1}^{n_q} \mathrm{Adv}_E(k_i, t) \mid 1 \leq n_q \leq q;\ (k_i)_{i=1}^{n_q} \in \mathbb{N};\ \sum_{i=1}^{n_q} k_i = q\right\}. \tag{10}$$

(iii). By construction, for each occurred tweak value, Z initiates a random permutation. So in fact, Z operates *exactly* as $\bar{\Pi}(\cdot,\cdot)$. Hence:

$$\Xi_2 := \left|Pr(\mathcal{A}^{Z(\cdot,\cdot)} = 1) - Pr(\mathcal{A}^{\bar{\Pi}(\cdot,\cdot)} = 1)\right| = 0. \tag{11}$$

Now, using the hybrid argument, we have (8) is $\leq \Xi_0 + \Xi_1 + \Xi_2$, due to which (9-11) imply the upper bound in (7). As this holds for any adversary and any K, this completes the proof. □

If Adv_E is at least linear in q, i.e. $\mathrm{Adv}_E(q,\cdot) \in \Omega(q)$, the max-term can be upper bounded by $\mathrm{Adv}_E(q,t)$. However if the adversary is restricted to taking different tweak values only, in particular if T_i functions as a counter, it is clear from the proof that the upper bound reduces to $\mathrm{Adv}_E(q,t) + q\mathrm{Adv}_E(1,t)$ (as long as $q \leq 2^n$). Not surprisingly, the results are in agreement with [1].

Theorem 4. *Let $\bar{E}$ be a secure tweakable block cipher with security function $\mathrm{Adv}_{\bar{E}}$. Then, $(\mathcal{E}, \mathcal{D})$ as described above is a secure symmetric encryption scheme. More specifically, for any t, q_e, and $\mu_e = \min\{qn, n2^n\}$:*

$$\mathrm{Adv}^{cpa}_{\mathcal{E}}(n, t, q_e, \mu_e) \leq 2\mathrm{Adv}_{\bar{E}}(q, t). \tag{12}$$

The formal proof is similar to the security proofs in [17] and it is omitted for brevity. The intuition is to use an adversary for $\mathcal{E}$ as a distinguisher between $\bar{E}$ and a random permutation, who has by definition advantage function $\mathrm{Adv}_{\bar{E}}$. Clearly $\mathrm{Adv}^{cpa}_{\mathcal{E}}$ is negligible in n as $\mathrm{Adv}_{\bar{E}}$ is negligible in n by Prop. 1 (recall that $\mathcal{K} = \mathcal{M} = \{0,1\}^n$). Following the discussion after Prop. 1, we can bound the security of $(\mathcal{E}, \mathcal{D})$ in terms of the basic primitive E as $\mathrm{Adv}^{cpa}_{\mathcal{E}}(n, t, q_e, \mu_e) \leq 2\mathrm{Adv}_E(q, t) + 2q\mathrm{Adv}_E(1, t)$.