On Side-Channel Resistant Block Cipher Usage

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

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

Abstract. Based on re-keying techniques by Abdalla, Bellare, and Borst, 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.

Keywords. Block ciphers, side-channel attacks, re-keying.

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 [14], power consumption [16], and even the electromagnetic radiation [2] emitted by cryptographic operations performed by physical devices. As a result, numerous algorithmic countermeasures [5,22] as well as new logic families [27,13] were developed to counteract such attacks. Unfortunately, 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 [21] 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. [31,30,32], 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

* Work done while visiting Philips Research.

be bound and thus proven secure in a formal model. Examples include pseudo-random generators [21,23] and stream ciphers [10,24]. 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 provably secure manner against side-channel attacks. To our knowledge, there has not been such attempt.

Re-keying Techniques. Re-keying techniques were formally studied in [1] in the black-box model[3]. Abdalla and Bellare consider two types of re-keying: *parallel* 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 ($K_i = f(k_i, 0)$, 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 [4]. Borst [6] considers similar techniques as well as specific instantiations of the constructions based on triple DES. Although [6] has no *formal* security analysis of the proposed constructions, he seems to be the first to observe in the scientific literature that frequently changing the key used to encrypt can be an effective countermeasure against side-channel attacks. Kocher [15] describes a similar technique to that of Borst although the description is not as general. Furthermore, no formal analysis is provided.

Our Contributions. Our contributions are as follows:

Formal Modelling. We prove under reasonable assumptions that the re-keying techniques are secure against side-channel attacks. Our model includes aspects of physical leakage models previously proposed in [31,23,30] and [10,24].

q-limited Adversaries. Recently, Standaert et al. [32] have adapted the definition of q-limited adversary to the side-channel setting extending Vaudenay [38]. We show that our proofs and model naturally fit the q-limited adversary and, in fact, are implied by it, thus making a natural connection between the theoretical leakage resilient cryptography models [10,24] and those inspired by practice [30,32].

Insecurity of Certain Schemes. We show that the re-keying techniques [1,6] naively implemented cannot hope to be proven secure against the side-channel adversary of the Dziembowski and Pietrzak (DzPz) model [10]. A similar statement is true for the scheme suggested in [15].

Implementation Strategy. On the positive side, we show that the constructions in [1,6] can be implemented in a way in which we can bound the amount of information leaked by the implementation. A key observation in this paper is that we can implement the parallel scheme described in [1,6] by storing

[3] We are not aware of any scientific publication previous to [1] 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.

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 [10]. To our knowledge, such implementation strategy seems to be novel, presumably because no-one had previously seen an advantage to it.

Comparison. 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[4].

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). On the other hand, the simplicity and usefulness of the countermeasure makes it highly attractive for implementation in the real world.

Related Work. Block ciphers and their use to build symmetric encryption schemes have been studied by Bellare et al. [4]. Liskov et al. [17] formalizedthe 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 [17]. We have already mentioned the treatment of re-keying techniques by Abdalla and Bellare [1] and Borst [6]. Regarding leakage-proof constructions, we are aware of two parallel lines of research, which originated from the work of Micali and Reyzin [21]: the information theoretic based analysis of Standaert et al. [31,30,32], and the somewhat more theoretical framework of [10,24]. These last works focus on the construction of leakage-resilient stream ciphers. Standaert et al. [31,30] introduced an information theoretical model in which to analyze and compare side-channel countermeasures. Macé et al. [18] have evaluated and compared different logic families using the framework introduced in [30].

2 Preliminaries

Notation. 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. By $\mathbb{F}_{n,m}$ we denote the set of polynomial time computable

[4] Mangard et al. [19,20] showed that in order to minimize the leakage of the masking countermeasure [5,22], it is necessary to guarantee glitch-free XOR gates.

functions $f : \{0,1\}^n \to \{0,1\}^m$.

Symmetric Primitives and Encryption Schemes. 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 *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 $\mathrm{Adv}_E(q,t) = \max_{\mathcal{A};K\in_R\mathcal{K}} \left|Pr(\mathcal{A}^{E_K(\cdot)} = 1) - Pr(\mathcal{A}^{\Pi(\cdot)} = 1)\right|$ 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* [17] 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 [17], 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. Symmetric encryption schemes and their security were studied by Bellare et al. [4]. 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 (usually 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 $\mathcal{D}_K(C) = \mathcal{D}_K(\mathcal{E}_K(M)) = M$.

Re-keying techniques. In [1], the authors formalized re-keying techniques and deduced symmetric encryption schemes from it. They introduced *parallel* and *serial* re-keying. In parallel re-keying, a map $F : \{0,1\}^k \times \{0,1\}^k \to \{0,1\}^k$ is used to compute the session keys $K_i = F(K,i)$ for $i = 1, 2, \ldots$, where K is the master key. In serial re-keying, the map F can be used to compute each session key from the previous one. One can think of F as a block cipher or a keyed hash function. Borst [6] observed that re-keying techniques form a powerful way to prevent side-channel attacks. In particular, he suggests to use session keys $K_i = F(K, R_i)$ where K is the master key, R_i is some data different for each session, and F is a sufficiently strong one-way function. Kocher [15] suggests a similar technique to Borst's except that the value of K_i is used several times during one session. In particular, his solution guarantees that each session key is not used (to perform a cryptographic operation) or derived more than a fixed (non-zero) number of times. In addition, the functions used to derive the session keys are efficiently invertible. In turn, invertibility of the functions allows re-use of the key for several (non-contiguous) sessions.

3 Re-keying Based Block Cipher and Encryption Scheme

Due to space limitations, we consider parallel re-keying only. The case of serial re-keying is discussed in the full version of this paper. We summarize the parallel

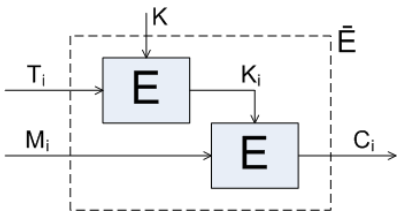

Fig. 1. The tweakable block cipher with parallel re-keying.

re-keying scheme of [1,6] from a tweakable block cipher point of view. This will facilitate the proof of side-channel security. Security in the black-box model has been proven in [1]. 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. 1. In our particular application, the tweak T functions as a counter. Executing the tweakable block cipher $\bar{E}$ in ECB mode results in a secure symmetric cipher [4]. 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 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 [4]):

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.

4 Side-Channel Security Model

We consider a model in which during each execution of $\mathcal{E}_K$ the system might leak information, either about the master key K or about a session key K_i. 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.

Side-Channel Model. We state the assumptions needed to facilitate the side-channel security proof. These were also used in [21,23,10,24]. We stress that these assumptions are reasonable, as argued in the following.

The leakage of information per execution of $\bar{E}$ is 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.

Only accessed memory leaks data. This guarantees that session keys not accessed during an encryption operation do not leak. It is well-known that in practice memory leaks power even when not accessed and that this effect increases as we go down in technology node. However, what this assumption implies is either that the leakage can be ignored or that the memory array has been somehow built so as to leak power but no information about its contents. Notice that *any* cryptographic functionality implemented as a physical circuit must guarantee this. Otherwise, an attacker would only need to wait long enough for the key to leak while stored in memory.

Implementation Details. In addition to these assumptions, we assume that the session keys have been pre-computed, stored in memory 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 [22,8,27,13], namely *simplicity* and *cost efficiency*. For convenience, we will only focus on an adversary eavesdropping *encryption* executions. It suffices to consider messages of block size $m = 1$: anything an adversary can learn from q executions on m blocks, he could have learned from qm executions on 1 block.

Side-Channel Adversary. Given the leakage length λ, we consider a side-channel adversary $\mathcal{A}$ that has t time, and actively eavesdrops the physical data of q executions. Formally, this corresponds to the notion of q-limited adversaries and q-limiting constructions, due to Standaert et al. [32].

Definition 1. *(***[32, Defs. 7 and 8]***)* *An adversary against a block cipher E_K is* q-limited *for a key K, if he can only observe the encryption of q different plaintext values under this key. A block cipher based construction is* q-limiting *for an adversary $\mathcal{A}$, if the number of different encryptions performed under a single key that $\mathcal{A}$ can observe, is limited to q.*

The attack of the adversary is now 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 $\Lambda_i \in \mathbb{F}_{n,\lambda}$ and input M_i. The system reads its i-th key K_i from its memory, and it computes and publishes $C_i = E_{K_i}(M_i)$. Moreover, the value $\Lambda_i(K_i)$ is computed (representing the leakage) and sent to the adversary. The model is visualized in Fig. 2. From now on, the counter i in the output will be implicit. The scheme will be denoted by $\tilde{E} : \{0,1\}^n \times \mathbb{F}_{n,\lambda} \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^\lambda$ defined as:

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

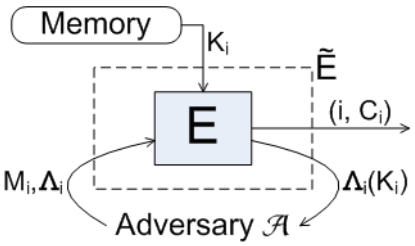

Fig. 2. One execution in the side-channel model with parallel re-keying.

It is clear that the above described model justifies 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 [10,24]). In the next sections, we will study in what sense the master key K and the session keys K_i are secure. This is done in Sect. 5, and follows the ideas of Standaert et al. [31,23,30].

Key Recovery Attack. Standaert et al. [31,30] introduced the notion of q-query key recovery adversaries, where the adversary can observe q executions of the block cipher in order to recover the corresponding key K. 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, Standaert et al. [31,30] 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 [30, Sect. 7.3]. For the security metric, they introduce two variants. Following [31,23], 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[5] $K^* = \arg\max_k Pr(K = k \mid (\Lambda_i(K))_{i=1}^q)$. Now, the '$q$-query success rate' of recovering K is defined to be:

$$\mathrm{SR}_{\tilde{E}}^{K} = \sum_{\mathbf{l};|d_{\mathbf{l}}|\neq 0} Pr((\Lambda_i(K))_{i=1}^q = \mathbf{l})\frac{1}{|d_{\mathbf{l}}|}, \text{ with } d_{\mathbf{l}} = \{k \mid (\Lambda_i(k))_{i=1}^q = \mathbf{l}\}. \quad (1)$$

Indeed, if the leakage function equals $\mathbf{l}$, the Bayesian adversary considers all key candidates that satisfy $(\Lambda_i(k))_{i=1}^q = \mathbf{l}$, hence collects the most likely key candidates, and will then take a value from this set. Intuitively, the success rate of recovering K is the expectation of the success taken over all possible leakage values $\mathbf{l}$.

q-query Success Rate in Practice. Given the importance of the number of queries in our model, it would be nice if it was possible to provide an idea

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

regarding the number of queries that a successful attack may require. Fortunately, several previous works [13,32] have done this and we limit ourselves to reproducing some of these numbers here, as shown in Table 1. We observe that the number of queries necessary for a successful attack varies considerably depending on the attack type, the implementation (software vs. hardware) and the platform.

Algorithm	**Device**	**Ref.**	Year	q	**Success Rate**
AES	PIC 8-bit controller	[25]	2009	≈ 2	1
	Atmel 8-bit controller (template attack)	[29]	2008	≈ 10	1
	Atmel 8-bit controller (correlation attack)	[29]	2008	≈ 100	1
DES	64-bit ASIC	[28]	2008	≈ 100	2^{-12}

Table 1. Number of queries q for successful attacks and corresponding attack success rates in different devices. Source [32].

5 Side-Channel Security of Parallel Re-keying

As the session keys are pre-computed outcomes of an encryption function, the master key cannot be recovered due to the security of the cipher. In particular, this security 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. In particular, (1) now reduces to:

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

We only need to prove that this success rate is sufficiently small, as the mutual information between the key and its leakage is $\leq \lambda$ by definition of Λ_i.

Theorem 1 (Session key security (parallel re-keying)). *Let $\mathcal{A}$ be any Bayesian adversary that can query a single tuple $(M_i, \Lambda_i) \in \{0,1\}^n \times \mathbb{F}_{n,\lambda}$ to his*

oracle $\tilde{E}(\cdot,\cdot;K_i)$. Then, the success rate of recovering K_i, $SR_{\tilde{E}}^{K_i}$ from eqn. (1), is negligible in n for fixed λ.

Proof. For the quantity expressed in (1), we have

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

where the second equality is by definition of d_l, and the third since $K_i \in_R \{0,1\}^n$. 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 [31]. 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. Now, $\mathrm{SR}_{\tilde{E}}^{K_i}$ from the proof of Thm. 1 equals $\frac{|\{l\in\{0,1\}^\lambda \mid |d_l|\neq 0\}|}{2^n}$. But as $|d_l| \neq 0$ only holds for $l \in \{0,\ldots,n\}$, it follows that $\mathrm{SR}_{\tilde{E}}^{K_i} = \frac{n+1}{2^n}$ (which is exactly the same result as in [31, Sect. 3.1]).

On The Insecurity of Certain Schemes. We consider the initial scheme of Borst [6, Sect. 6.6.1]. Here the session keys K_i are derived from a master key MK as $K_i = F(MK, R_i)$ for some random value R_i. As noticed already by Borst, this scheme is still susceptible to a side-channel attack as the value MK is used to derive session keys K_i. To counteract this threat, Borst also suggests to construct a key with intermediate keys IK_i derived by using the master key and a random value, and session keys derived in turn from the IK_i's and additional random values. Observe that the DzPz adversary (used in our model) 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 MK 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 an adversary can always obtain the whole key in $|K|/\lambda$ runs of the [1,6] schemes if the session keys have to be computed during each session by processing the master secret key MK. A similar attack is possible on the intermediate keys of [15] as intermediate keys are re-used more than once. Observe, however, that the scheme in [15] also compromises the master keys due to the choice of invertible functions to derive session keys. In particular, should a session key be compromised, all keys are compromised (including keys used in the past). We end by pointing out that we have only shown that the schemes described are not secure in the particular model that we consider. It is possible (and in fact shown in the full version of this paper [11]) that under relaxed assumptions on the power of adversaries, the scheme of Borst can provide side-channel security guarantees.

6 Comparison to Alternative Side-Channel Resistant Methodologies

In this section, we study in which situations the parallel re-keying scheme analyzed in this paper is competitive in terms of area costs with respect to alternative countermeasures. Clearly, the parallel re-keying 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. Observe that an area cost analysis implicitly considers memory as a cost measure. In particular, memory limited devices have limited memory because additional memory requires additional area, which in turn, implies that the final chip would increase in monetary costs.

Area-Cost as the Right Metric for Comparison. 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[6]. 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 pessimistic comparison from the point of view of the approach proposed in this paper. However, this is in agreement with the common practice in cryptography to take a conservative 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 and we focus on it. 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

[6] Mangard et al. [19,20] have shown attacks on masked implementations and Macé et al. [18] 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 [33] against several types of dual rail logic approaches including WDDL [35].

memory) but we expect that a plain AES implementation will in general have much lower power consumption than the alternatives implemented using modifed logic styles (see, e.g., [34] for a discussion). We end by noticing that an actual implementation [36] comparing a plain AES and AES using WDDL logic [35] confirms our assumptions: performance is reduced by more than a factor of three and power consumption is increased by almost four times. We consider the following (alternative) side-channel attack countermeasures: algorithmic solutions as introduced by Blömer et al. [5] and later optimized in [22], as well as low level solutions based on dual rail logic [27,13].

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 [8] for our estimates[7] together with the one of Akkar and Giraud [3], who describe the *overall* architecture of a masked AES implementation. Although [3] focuses on the software implementation of such scheme, the same ideas can be easily implemented in the hardware domain. We compute the estimates to be able to fairly compare available methodologies[8]. 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 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. [26] to estimate the cost of a plain (unmasked) AES implementation. We have ignored the cost of the key scheduling in all our computations. Table 2 compares a masked implementation with an unmasked implementation.

An alternative way to achieve DPA resistance is to develop dual-rail logic families. The area overhead of different dual-rail logic families has been considered in [27] and ranges from at least twice to as much as 12 times as large, depending on the specific logic family (see [27,13,34] for comparisons). Table 3 provides a summary of the area-cost of different AES implementations. Table 3

[7] The implementation of [8] is the smallest mask implementation of an S-box that we are aware of, but it has been found to be flawed in [9]. Canright remarks in [7] that the cost will be significantly higher than predicted. Thus, this is a good "lower bound" on the cost of a masked implementation.

[8] We are only aware of [37] reporting a whole masked AES processor. We do not use their numbers because it would make it hard to compare all other approaches. In particular, this implementation is considerably larger (unmasked) than the best reported in [26]. However, the overhead due to masking is quite close to 100% in agreement with our estimates. Observe that the larger area requirements would only make our implementation strategy more attractive, as we would be able to store more keys.

Component	unmasked implementation		masked implementation	
	GE	**Source**	**GE**	**Source**
4 S-boxes	1176	[26]	2436	[8]
Data Register	864	[26]	1728	[26,3]
ShiftRows	160	[26]	320	[26,3]
MixColumns	273	[26]	546	[26,3]
AddRoundKey	56	[26]	112	[26,3]
Total	2529	—	5142	

Table 2. Actual costs of an unmasked AES according to [26] and estimated costs of a masked AES implementation based on [8] and [3]. We ignore the selector, the key scheduling and the overhead of supporting inverse cipher. GE: Gate Equivalents.

reports two logic families. We have not included SecLib [12,13] because it has slightly worse area requirements than WDDL.

Approach	Source	% overhead logic (GE)	% overhead Flip-Flops (GE)	Total Area Cost (GE)	% Total overhead
plain	[26]	0%(1665)	0%(864)	2529	0%
masked	Table 2	—	—	5142	103 %
dual-rail	[27]	108% (3464)	228% (2851)	6315	150 %
WDDL	[35]	100% (3330)	300% (3456)	6786	168 %

Table 3. Estimated costs of a 4-Sbox AES implementation with different technologies. Key scheduling and the overhead of supporting inverse cipher are not considered. GE: Gate Equivalents.

Comparison with Parallel Re-keying Technique. 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 2 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 [27,35,12]. 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 and the 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.

7 Conclusions

We considered the re-keying techniques described in [1,6], and proved them secure in the side-channel model following the models introduced in [23,10]. A drawback of the scheme is that it only offers possibility for a limited number of encryptions, but as shown in Sect. 6, 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 [6]) 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 [1,6] 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 [6].

Acknowledgments. This work has been funded in part by the European Community's Sixth Framework Programme under grant number 034238, SPEED project - Signal Processing in the Encrypted Domain, in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II. The second author is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

1. Abdalla, M., Bellare, M.: Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-keying Techniques. In: ASIACRYPT '00. LNCS, vol. 1976, pp. 546–559. Springer (2000)

2. Agrawal, D., Archambeault, B., Rao, J., Rohatgi, P.: The EM Side–Channel(s). In: CHES '02. LNCS, vol. 2523, pp. 29–45. Springer (2003)
3. Akkar, M., Giraud, C.: An Implementation of DES and AES, Secure against Some Attacks. In: CHES '01. LNCS, vol. 2162, pp. 309–318. Springer (2001)
4. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A Concrete Security Treatment of Symmetric Encryption. In: FOCS '97. pp. 394–403. Proceedings of the 38nd IEEE symposium on FOCS, IEEE Computer Society (1997)
5. Blömer, J., Guajardo, J., Krummel, V.: Provably Secure Masking of AES. In: SAC '04. LNCS, vol. 3357, pp. 69–83. Springer (2004)
6. Borst, J.: Block Ciphers: Design, Analysis, and Side-channel Analysis. Ph.D. thesis, Katholieke Universiteit Leuven (September 2001)
7. Canright, D.: Avoid Mask Re-use in Masked Galois Multipliers. Cryptology ePrint Archive, Report 2009/012 (2009)
8. Canright, D., Batina, L.: A Very Compact "Perfectly Masked" S-Box for AES. In: ACNS '08. LNCS, vol. 5037, pp. 446–459. Springer (2008), corrected extended version available in [9]
9. Canright, D., Batina, L.: A Very Compact "Perfectly Masked" S-Box for AES (corrected). Cryptology ePrint Archive, Report 2009/011 (2009)
10. Dziembowski, S., Pietrzak, K.: Leakage-Resilient Cryptography. In: FOCS '08. pp. 293–302. Proceedings of the 49nd IEEE symposium on FOCS, IEEE Computer Society (2008)
11. Guajardo, J., Mennink, B.: Towards side-channel resistant block cipher usage or can we encrypt without side-channel countermeasures? Cryptology ePrint Archive, Report 2010/015 (2010)
12. 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), 546–555 (2007)
13. 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)
14. Kocher, P.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: CRYPTO '96. LNCS, vol. 1109, pp. 104–113. Springer (1996)
15. Kocher, P.: Leak-Resistant Cryptographic Indexed Key Update (Filed July 2, 1999), patent No.: US 6539092 B1. Date of Patent: March 25, 2003
16. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: CRYPTO '99. LNCS, vol. 1666, pp. 388–397. Springer (1999)
17. Liskov, M., Rivest, R., Wagner, D.: Tweakable Block Ciphers. In: CRYPTO '02. LNCS, vol. 2442, pp. 31–46. Springer (2002)
18. Macé, F., Standaert, F., Quisquater, J.: Information Theoretic Evaluation of Side-Channel Resistant Logic Styles. In: CHES '07. LNCS, vol. 4727, pp. 427–442. Springer (2007)
19. Mangard, S., Pramstaller, N., Oswald, E.: Successfully Attacking Masked AES Hardware Implementations. In: CHES '05. LNCS, vol. 3659, pp. 157–171. Springer (2005)
20. Mangard, S., Schramm, K.: Pinpointing the Side-Channel Leakage of Masked AES Hardware Implementations. In: CHES '06. LNCS, vol. 4249, pp. 76–90. Springer (2006)
21. Micali, S., Reyzin, L.: Physically Observable Cryptography. In: TCC '04. LNCS, vol. 2951, pp. 278–296. Springer (2004)

22. Oswald, E., Mangard, S., Pramstaller, N., Rijmen, V.: A Side-Channel Analysis Resistant Description of the AES S-Box. In: FSE '05. LNCS, vol. 3557, pp. 413–423. Springer (2005)
23. Petit, C., Standaert, F., Pereira, O., Malkin, T., Yung, M.: A Block Cipher Based PRNG Secure Against Side-Channel Key Recovery. In: ASIACCS '08. pp. 56–65. ACM (2008)
24. Pietrzak, K.: A Leakage-Resilient Mode of Operation. In: EUROCRYPT '09. LNCS, vol. 5479. Springer (2009)
25. Renauld, M., Standaert, F.X., Veyrat-Charvillon, N.: Algebraic Side-Channel Attacks on the AES: Why Time also Matters in DPA. In: CHES 2009. LNCS, vol. 5747, pp. 97–111. Springer (2009)
26. Satoh, A., Morioka, S., Takano, K., Munetoh, S.: A Compact Rijndael Hardware Architecture with S-Box Optimization. In: ASIACRYPT '01. LNCS, vol. 2248, pp. 239–254. Springer (2001)
27. Sokolov, D., Murphy, J., Bystrov, A., Yakovlev, A.: Design and Analysis of Dual-Rail Circuits for Security Applications. IEEE Trans. Computers 54(4), 449–460 (2005)
28. Standaert, F.X., Bulens, P., de Meulenaer, G., Veyrat-Charvillon, N.: Improving the Rules of the DPA Contest. Cryptology ePrint Archive, Report 2008/517 (2008)
29. Standaert, F.X., Gierlichs, B., Verbauwhede, I.: Partition vs. Comparison Side-Channel Distinguishers: An Empirical Evaluation of Statistical Tests for Univariate Side-Channel Attacks against Two Unprotected CMOS Devices. In: ICISC 2008. LNCS, vol. 5461, pp. 253–267. Springer (2008)
30. Standaert, F.X., Malkin, T., Yung, M.: A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In: EUROCRYPT '09. LNCS, vol. 5479, pp. 443–461. Springer (2009)
31. Standaert, F.X., Peeters, E., Archambeau, C., Quisquater, J.: Towards Security Limits in Side-Channel Attacks (With an Application to Block Ciphers). In: CHES '06. LNCS, vol. 4249, pp. 30–45. Springer (2006)
32. Standaert, F.X., Pereira, O., Yu, Y., Quisquater, J.J., Yung, M., Oswald, E.: Leakage resilient cryptography in practice. Cryptology ePrint Archive, Report 2009/341 (2009)
33. Suzuki, D., Saeki, M.: Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style. In: CHES '06. LNCS, vol. 4249, pp. 255–269. Springer (2006)
34. Tillich, S., Großschädl, J.: Power Analysis Resistant AES Implementation with Instruction Set Extensions. In: CHES '07. LNCS, vol. 4727, pp. 303–319. Springer (2007)
35. Tiri, K., Verbauwhede, I.: A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation. In: DATE '04. pp. 246–251. IEEE Computer Society (2004)
36. Tiri, K., Verbauwhede, I.: A digital design flow for secure integrated circuits. IEEE Trans. on CAD of Integrated Circuits and Systems 25(7), 1197–1208 (2006)
37. Trichina, E., Korkishko, T.: Secure AES Hardware Module for Resource Constrained Devices. In: ESAS '04. LNCS, vol. 3313, pp. 215–230. Springer (2005)
38. Vaudenay, S.: Decorrelation: A Theory for Block Cipher Security. J. Cryptology 16(4), 249–286 (2003)