Unknown-Input Attacks in the Parallel Setting: Improving the Security of the CHES 2012 Leakage-Resilient PRF

Marcel Medwed[1], François-Xavier Standaert[2⋆]
Ventzislav Nikov[3], Martin Feldhofer[1].

[1] NXP Semiconductors Austria.
[2] ICTEAM/ELEN/Crypto Group, Universite catholique de Louvain, Belgium.
[3] NXP Semiconductors Leuven, Belgium.

Abstract. In this work we present a leakage-resilient PRF which makes use of parallel block cipher implementations with unknown-inputs. To the best of our knowledge this is the first work to study and exploit unknown-inputs as a form of key-dependent algorithmic noise. It turns out that such noise renders the problem of side-channel key recovery intractable under very little and easily satisfiable assumptions. That is, the construction stays secure even in a noise-free setting and independent of the number of traces and the used power model. The contributions of this paper are as follows. First, we present a PRF construction which offers attractive security properties, even when instantiated with the AES. Second, we study the effect of unknown-input attacks in parallel implementations. We put forward their intractability and explain it by studying the inevitable model errors obtained when building templates in such a scenario. Third, we compare the security of our construction to the CHES 2012 one and show that it is superior in many ways. That is, a standard block cipher can be used, the security holds for all intermediate variables and it can even partially tolerate local EM attacks and some typical implementation mistakes or hardware insufficiencies. Finally, we discuss the performance of a standard-cell implementation.

1 Introduction

Countermeasures against side-channel attacks always imply implementation overheads and rely on physical assumptions. So designing such countermeasures comes with the equally important goals of maximizing security, while minimizing the overheads and relying on physical assumptions that are easy to fulfill by cryptographic engineers. Mainstream masking schemes (i.e. data randomization based on secret sharing) are a typical example of this tradeoff, where security is exponential in the number of shares, performances are quadratic in the number of shares, and implementers need to guarantee that the leakages of the shares are

⋆ Associate researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been funded in part by the ERC project 280141 (acronym CRASH) and by the ARC project NANOSEC.

independent and sufficiently noisy [7,10,15,26]. (Note that the condition of independent leakages is typically hard to guarantee, both in software and hardware implementations [2,8,17,18]). Threshold implementations are a specialization of masking that reduces the independence requirement (by ensuring that glitches do not harm the security of the masked implementations) [5,23], which can also lead to some performance gains with low number of shares [6,22].

At CHES 2012, a quite different tradeoff was introduced. Namely, and starting from the observation that leakage-resilience via re-keying alone is not sufficient to efficiently protect stateless symmetric cryptographic primitives such as block ciphers (later formalized in [3]), Medwed et al. proposed a tweaked construction of an AES-based leakage-resilient PRF, inspired from more formal works such as [1,9,11,28,32], which additionally requires that the AES is implemented in parallel and that its S-boxes have similar leakage models [20]. In this respect, and while the parallel implementation setting is easy to guarantee (and can even be emulated thanks to shuffling [14]), the "similar leakage assumption" turned out to be harder to evaluate. Later results showed that despite not easy to attack, such a solution may not be best suited the AES [4].

In this paper, we aim to improve the tradeoff between security, performance and physical assumptions for the CHES 2012 construction. For this purpose, our main ingredient is to replace the similar leakage assumption by an easier-to-guarantee requirement of unknown plaintexts. Interestingly, this requirement can be easily satisfied by exploiting a leakage-resilient stream cipher in order to generate these plaintexts (we use the efficient construction from [27] for this purpose). As a result, our contributions are as follows. We first describe our new construction of a leakage-resilient PRF based on unknown plaintexts. Second, we analyze its security in front of standard side-channel attacks where the adversary can observe noisy Hamming weight leakages (and compare it with the CHES 2012 proposal). Third, we evaluate the impact of implementation issues such as deviations from the Hamming weight leakages and leakages due to transitions between registers. Finally, we discuss alternative attack paths and put forward the good performances of our new construction. As part of our investigations, we also highlight the interesting security guarantees offered by the combination of unknown cipher inputs and parallel implementations for side-channel resistance, which is of independent interest.

Note that despite our design is inspired by previous constructions of leakage-resilient PRFs, the security guarantees we claim for it are significantly less formal/general. First, the only security we claim is key recovery security, as for the CHES 2012 PRF. This limitation is motivated by the discussions in [21,24], where it was shown that indistinguishability of the inputs is essentially impossible in a physically observable setting (excepted by artificially excluding some leakages of the analysis, which then reduces practical relevance). Second, it is worth emphasizing that our analyzes are only heuristic and based on concrete attacks. In this respect, the goal of our proposal is not to be proven secure under a formal model (in parts because it relies on non-standard implementation assumptions such as parallelism which make formal treatments much more challenging). By contrast, it is an attempt to implement a building block with bounded leakage. In other words, it is more an attempt to instantiate a way to fulfill the basic assumptions of leakage-resilient cryptography than an attempt to formally analyze a leakage-resilient primitive or functionality. For example, our unknown-input PRF could be a candidate for the leak-free block cipher required in [24].

2 Background: the CHES 2012 leakage-resilient PRF

We start with the description of the standard GGM PRF [13], depicted in the left part of Fig. 1, on which the CHES 2012 PRF is based. Let $\mathsf{F}_k(x)$ denote the PRF indexed by k and evaluated on x. Further, let the building blocks $\mathsf{E}_{k^i}(p_j^i)$ denote the application of a block cipher E to a plaintext p_j^i under a key k^i (the figure shows the example of E = AES-128 with $1 \leq i \leq 128$ and $0 \leq j \leq 1$). Let also $x(i)$ denote the i^{th} bit of x. The PRF first initializes $k^0 = k$ and then iterates as follows: $k^{i+1} = \mathsf{E}_{k^i}(p_0^i)$ if $x(i) = 0$ and $k^{i+1} = \mathsf{E}_{k^i}(p_1^i)$ if $x(i) = 1$. Eventually, the $(n+1)^{th}$ intermediate key k^{128} is the PRF output as $\mathsf{F}_k(x)$.

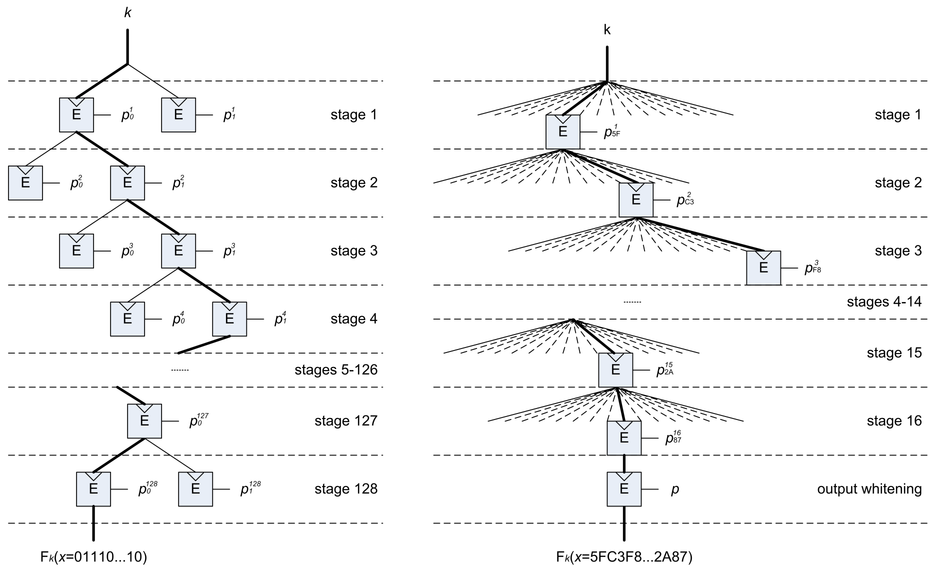

Fig. 1. Leakage-resilient PRFs: straight GGM (left) and efficient alternative (right).

In this basic version, the execution of the PRF guarantees that any side-channel adversary will at most observe the leakage corresponding to two plaintexts per intermediate key (p_0^i and p_1^i). This implies 128 executions of the AES-128 to produce a single 128-bit output. A straightforward solution to trade improved performances for additional leakage is to increase the number of observable plaintexts per intermediate key. If one has N_p such plaintexts per stage, the number of AES-128 executions to produce a 128-bit output is divided by $\log_2(N_p)$. However, as already discussed in [20], such a tradeoff scales badly and very rapidly decreases the side-channel security of an implementation (as it typically allows DPA with N_p observable plaintexts).

To avoid this drawback, an efficient alternative (also proposed in [20]) is illustrated in the right part of Fig. 1. It can be viewed as a GGM construction with $N_p = 256$, but where the same set of 256 carefully chosen plaintexts is re-used in each PRF stage, excepted for the last stage where $N_p = 1$. In terms of efficiency, this proposal reduces the number of stages of a PRF based on the AES-128 to 17 (i.e. 16 plus one final whitening).

The security of this second construction is based on the combination of parallelism with carefully chosen plaintext values, in order to prohibit the application of standard divide-and-conquer strategies. For this purpose, plaintexts of the form $p_j = \{j-1\}^{N_s}$, with $1 \leq j \leq N_p$ and N_p being limited by the S-box input space were considered. Given that all S-boxes leak in parallel, the effect of this measure is that in a DPA attack, the predictions corresponding to the N_s key bytes cannot be distinguished anymore, because these key bytes have to be targeted at the same time. As a result, and even when increasing N_p, not all the N_s key bytes can be highly ranked by the attack. (We will re-detail this effect in Section 4.1, which is reflected by the higher guessing entropy of the targeted key bytes in Fig. 3). In [20], it was even shown that slight differences in the implementation – and therefore in the leakage of the N_s S-boxes – are not easily exploitable. Eventually, if N_s becomes sufficiently large, ordering the N_s recovered subkeys has a cost of $N_s!$, meaning that even after seeing all leakages without noise, the adversary cannot fully recover the key.

Unfortunately, and despite conceptually appealing, this construction has several drawbacks which limit its applicability. First, the security parameter N_s is defined by the number of S-boxes of the underlying block cipher. For some of the currently standardized block ciphers N_s is not large enough (e.g. $N_s = 16$ for the AES-128, which corresponds to an insufficient $N_s! \approx 2^{44}$). Second, if intermediate values other than the first round's S-box outputs are targeted, the leakages might be sufficiently independent such that divide-and-conquer strategies work again. While this generally requires more computational power, recent results on multi-target attack DPA show that it is not out of reach [19]. (This is in fact the reason why attacks on the ciphertext need to be prevented by the whitening step in the CHES 2012 proposal). Finally, the size of the S-box defines the maximum value of N_p and hence the maximum throughput.

3 New leakage-resilient PRF construction

We now present a new construction which improves over the one in [20] in terms of performance and security, at the cost of higher memory requirements. For this purpose, we introduce a pre-computation step in which we generate N_p secret, distinct plaintexts. This step can be seen in Fig. 2. It essentially uses the leakage-resilient PRG from [27] to generate 2^m secret plaintexts $p_0 \dots p_{2^m-1}$ as well as an updated key k'. These secret plaintexts and updated key are then simply used in a tree-based PRF such as in the right side of Fig. 1. The output whitening step stays the same. By design, this new construction has the advantage (compared to the CHES 2012 one) that the plaintexts are secret and of no particular form. This implies that their number is not bounded by the S-box size, allowing for smaller trees of depth $128/m + 1$. From a security point of view, it also comes with interesting implications:

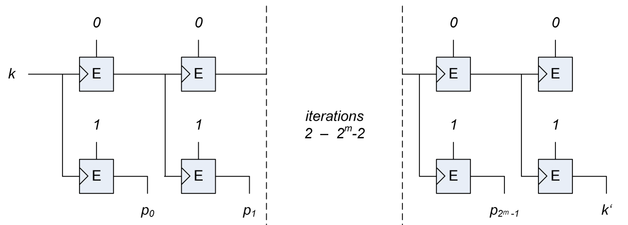

Fig. 2. Leakage-resilient PRG used to generate the 2^m secret plaintexts.

1. Since the plaintexts are unknown, a straight-forward unprofiled DPA is ruled out. Instead, an adversary has to build templates (for instance for the bivariate variable made of the plaintext and S-box output leakages).
2. For a similar reason, there is no straightforward way to verify a key candidate: for this purpose, one would not only need to recover the key but also at least one secret plaintext. In the worst case where the information leakages are not sufficient (i.e. if a successful attack requires additional key/plaintext enumeration [30]) this squares the attack time complexity.
3. As for the CHES 2012 construction using carefully chosen plaintexts, the adversary has no way of separating the leakages from the different subkeys. But contrary to this previous work, this feature now applies to any intermediate variable within the algorithm (not only to the first round leakages).

4 Security analysis w.r.t. basic side-channel attacks

We now detail our security analysis against standard side-channel attacks and use the following notations. First, k denotes a key, k^* denotes a key candidate and k_j the j^{th} byte of a key. Next, $p_{i,j}$ is the j^{th} byte of the i^{th} plaintext out of q ones that are available to an adversary. For p and k, j is assumed to be in the range $1, \ldots, N_s$ where $N_s = 16$ for the AES. Further, t_i is a trace (aka leakage) vector, corresponding to the i^{th} plaintext. A trace may contain several leakage points, denoted by $t_{i,j}$. L denotes the leakage function, e.g. the Hamming weight function in our examples below. Finally, $\mathsf{L}(\mathsf{S}(k_1 \oplus p_{2,1}))$ denotes the leakage of the S-box output corresponding to S-box 1 for the 2nd plaintext. The set of all plaintexts is denoted as P and the set of all traces as T.

In a standard DPA attack, the adversary pursues a divide-and-conquer approach. That is, he first computes the correct subkeys as

$$\tilde{k_j} = \arg\max_{k_j^*} \Pr(k_j^* | p_{1,j} \dots p_{q,j}, t_{1,j} \dots t_{q,j}).$$

Here, $t_{i,j}$ denotes the sample within trace i which only leaks about k_j. Afterwards he combines these subkeys to $\tilde{k}$. The attack is successful if $\tilde{k} = k$.[4] In a parallel hardware scenario, t_i consists of a single leakage point that we approximate as:

$$t_{i,1} = \sum_{j=1}^{N_s} \mathsf{L}(\mathsf{S}(k_j \oplus p_{i,j})). \tag{1}$$

Nevertheless, even in this parallel scenario, an adversary can always target a single key byte at a time by computing:

$$\tilde{k_j} = \arg\max_{k_j^*} \Pr(k_j^* | p_{1,j} \dots p_{q,j}, t_1 \dots t_q).$$

In this case, by just looking at a specific S-box or byte of the key, an adversary neglects the other key bytes and their contribution to the leakage is interpreted as (algorithmic) noise, which eventually averages out if plaintexts are uniformly distributed. As already discussed in [20], for carefully chosen plaintexts, $p_{1,1} = p_{1,j}$ for all $j = 1 \dots N$. Therefore, the equation becomes:

$$\tilde{k_j} = \arg\max_{k_j^*} \Pr(k_j^* | p_{1,1} \dots p_{q,1}, t_1 \dots t_q) \tag{2}$$

and all $\tilde{k_j}$ are the same. That is, since the probability condition is no longer dependent on j, only one joint score vector can be obtained, which contains the information about all the N_s target key bytes at once. For unknown-plaintext attacks, an adversary finally faces the problem of finding:

$$\tilde{k_j} = \arg\max_{k_j^*} \Pr(k_j^* | (t_{1,1}, t_{1,2}) \dots (t_{q,1}, t_{q,2})), \tag{3}$$

where he has no direct access to plaintext information, and therefore must extract this information from the traces as well (reflected by the second sample of the traces in the equation). We assume that this information is separately available and that the traces take the form:

$$(t_{i,1}, t_{i,2}) = \left(\sum_{j=1}^{N_s} \mathsf{L}(p_{i,j}), \sum_{j=1}^{N_s} \mathsf{L}(\mathsf{S}(k_j \oplus p_{i,j})) \right). \tag{4}$$

This has the following important implications on the attack:

[4] If for some or all values of j $\tilde{k_j} \neq k_j$, one may still find k using key enumeration techniques in the combination step, given that the bias is sufficiently high [30].

1. The adversary cannot apply a divide-and-conquer brute-force attack anymore. As in the case of carefully chosen plaintexts, also here the probability's condition becomes independent of j, which results in only a single score vector containing the information for all the N_s subkeys.
2. Successful attacks have to be bi-variate ones, in which a second-order moment of the leakage distribution is exploited. This makes them more sensitive to noise. Furthermore, as for the CHES 2012 construction as well, $N_s - 1$ contributors for each leakage point represent key-dependent algorithmic noise, and cannot be averaged out like in the case of masking (as noted in [3]). [5]

In the following we present three experiments. In the first one we recap the security of the CHES 2012 scheme in order to allow for a later comparison. We do so by estimating the guessing entropy and the subkey rank distribution as a function of N_s after seeing all possible traces. In the second experiment, we do the same for our improved proposal. This allows us to highlight the security improvement. In a third experiment we look at the model errors which are the reason for the security improvement. All experiments are carried out based on template attacks as this represents the most powerful side-channel adversary. We used discrete histograms (instead of continuous distributions) for our templates since the leakage function (aka power model) used in our experiments is also discrete and no noise is added. Hence, the number of bins is determined automatically and the histograms capture all the available information. Finally, we evaluate our metrics for increasing number of traces (with bounded number of plaintexts in the case of the CHES 2012 construction).

4.1 Security based on carefully chosen plaintexts

For the CHES 2012 scheme, the plaintexts are known, the target function is the AES S-box and the assumed power model is the Hamming weight model. Thus, the leakages are in the form of Equation (1). Knowing this, we can generate a template $\mathcal{D}^i$ for each of the subkey candidates, assuming the plaintext to be zero. In our simulations we look at N_s parallel AES S-boxes and the leaking variables are 8-bit valued. Therefore, each template is a histogram with $8 \cdot N_s + 1$ bins, starting at bin $\mathcal{D}^i(0)$ which indicates the probability that for a subkey $k = i$, the leakage sample has a value of 0. The templates are built according to Algorithm 1.

During the attack phase, the templates have been permuted according to the plaintext byte, that is, the probability for a certain leakage given a certain plaintext was calculated as $\Pr(t_1 | p_{1,j}, k_j^*) = \mathcal{D}^{k_j^* \oplus p_{1,j}}(t_1)$.

The result of the known plaintext attack can be seen in Fig. 3. The left plot represents a scenario where the plaintexts were not carefully chosen and therefore, the S-boxes leak independently. This just serves as a reference for the right plot, where the actual CHES 2012 scheme with carefully chosen plaintexts was analyzed. The y-axes represent the average key rank of k_1 in $\log_2$-scale. A

[5] Note, that even if an implementation unintentionally compresses the distribution to a uni-variate one with an informative first-order moment, exploitations do not automatically become easier as discussed in Section 5.2.

random guess would result in an average key rank of 128 and thus a 7 in $\log_2$-scale indicates that no information was retrieved via the side channel. Zero on the other hand indicates that the correct key was ranked first and thus it has been recovered with certainty. The x-axis shows the number of required traces to reach a certain average key rank, again in $\log_2$-scale. The different curves represent different numbers of parallel S-boxes ranging from 1 to 32 in powers of two. Each curve has been averaged over 10k attacks. On the right side we can observe a stagnation of the average rank at approximately $(N_s + 1)/2$ for $N_s \leq 8$ (in $\log_2$ this results in 0, 0.6, 1.3, and 2.2). As the adversary targets all subkeys at the same time, this is what one would expect intuitively. However, for 16 and 32 S-boxes, the average rank becomes higher, namely $\log_2(11.2) = 3.5$ (instead of 3.1) and $\log_2(27.1) = 4.8$ (instead of 4.0). This may look surprising, since due to the higher probability of collisions (i.e. repetitions within the N_s subkey values for large values of N_s) the rank could be expected to be below $(N_s + 1)/2$. However, as the number of S-boxes increases, the key-dependent algorithmic noise also increases and starts to dominate, implying that incorrect keys start to be ranked amongst the most likely ones in this case.

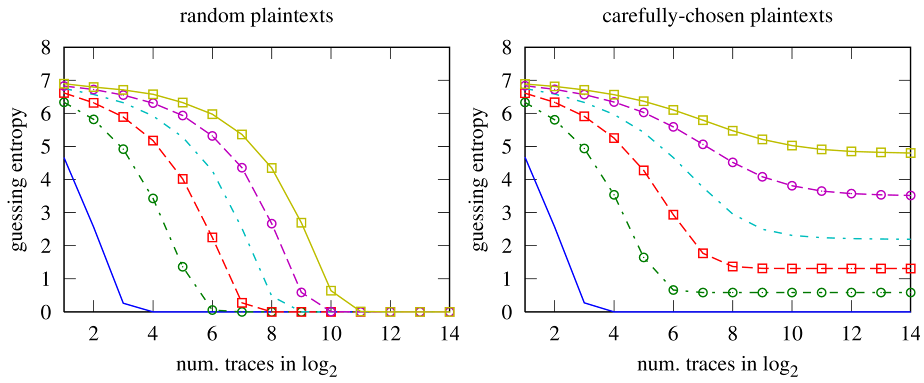

Fig. 3. Average guessing entropy after attacks with known plaintexts for $N_s = 1$ (blue,s/), 2 (green,dd/c), 4 (red,d/s), 8 (cyan,dd/), 16 (magenta,d/c), and 32 (yellow,s/s) with {s=solid,d=dashed,dd=dotted dashed}/{c=circle,s=square}.

Next to the average guessing entropy, it is also insightful to look at the rank distribution after seeing all possible leakages. This is done by analyzing the device's leakage distribution that we denote as $\mathcal{D}$. For instance, given two S-boxes and two subkeys k_1 and k_2, the exact leakage distribution of such device can be computed as $\mathcal{D} = conv(\mathcal{D}^{k_1}, \mathcal{D}^{k_2})$ (using convolutions reduces the complexity of computing $\mathcal{D}$ for an 8-bit S-box from $2^{8 \cdot N_s}$ for the naive approach to $(8 \cdot N_s + 1)^2$). The outcome of this experiment for 1000 random keys can be seen in Fig. 4. The plots show the PMF (in solid blue) and the CDF (in dotted dashed green) for the rank distribution after seeing all possible traces for $N_s = 16$ with carefully chosen plaintexts. The x-axis corresponds to the key ranks and the y-axis corresponds to the probabilities. This figure confirms the previous observations with additional intuitions. First for the left plot, since the median is at rank ≈ 8 for each subkey in this case, an adversary would have a success rate of 0.5 to find the subkey within the ≈ 8 most likely candidates. Next, in the middle plot, we show the distribution of the minimum rank within the 16 subkeys k_j. It can be clearly seen that the subkey ranked first is almost surely one of the correct ones. This is an important observation and will allow us to construct an advanced attack in Section 6.1. As for the distribution of the maximum rank within the 16 subkeys k_j in the right plot, it can be seen that below rank 16, the success rate is almost zero since this can only happen (but is not given) if two subkeys are equal. Finally, in order to have a success rate of 0.5 to see the worst ranked k_j (and therefore also seeing all other correct subkeys), the adversary would need to look at the first 37 most likely candidates.

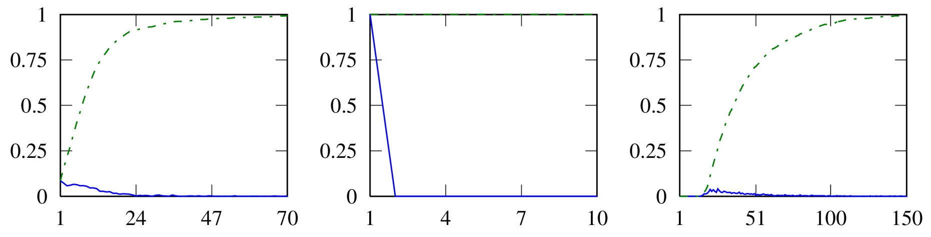

Fig. 4. Rank distributions for carefully-chosen plaintexts with $N_s = 16$. Average rank of subkey k_1 (left), average minimum rank amongst all k_j's (middle) and average maximum rank amongst all k_j (right).

4.2 Security based on unknown plaintexts

For the unknown plaintext scenario we targeted leakages in the form of Equation (4) and generated the templates as two-dimensional histograms. Each dimension has $8 \cdot N_s + 1$ bins, starting at bin $\mathcal{D}^i(0,0)$ which indicates the probability that for key $k = i$ both leakage samples have a value of 0. The templates are built according to Algorithm 2.

The left side of Fig. 5 again shows a reference result for independent noise. Since, in the unknown plaintext scenario, we cannot decouple the noise by simply randomizing the plaintexts, we had to use a trick. Namely, we only fixed k_1 and randomly drew q different values for each k_j with $j \in 2, \ldots, N_s$. It can be seen that a recovery for $N_s = 1$ S-boxes requires around 2^8 traces, whereas for $N_s = 16$ around 2^{27} traces can be expected.

The right side of the figure represents the unknown-plaintext scenario (where the subkeys are constant over all traces within one instance of the experiment). It can be seen that key-dependent noise leads to a stagnation of the correct subkey's rank. This is similar to the carefully-chosen plaintext case and expected. However, the important difference compared with the previous experiment is that the stagnation does not take place at $y \approx \log_2((N_s + 1)/2)$ but much earlier. In order to get the full picture, we again look at the rank distributions in Fig. 6.

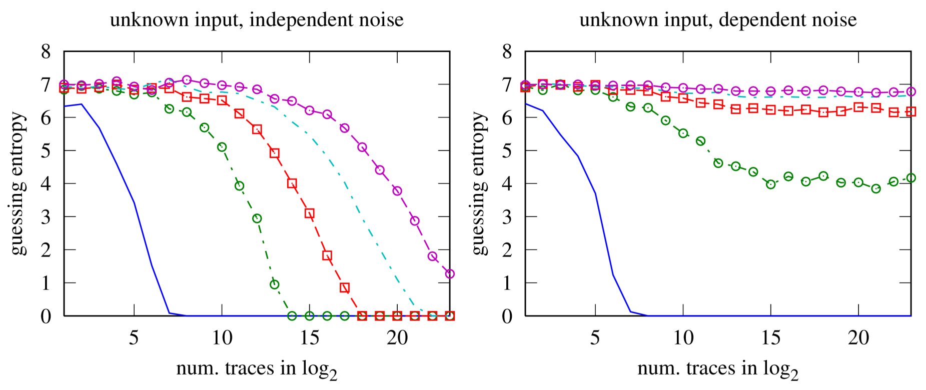

Fig. 5. Average guessing entropy after attacks with unknown plaintexts for $N_s =$ (blue,s/), 2 (green,dd/c), 4 (red,d/s), 8 (cyan,dd/), 16 (magenta,d/c).

First, we observe in the left plot that the subkey ranks (from 40 000 experiments) look close to uniformly distributed (which would be reflected by a straight line), with a median rank at ≈ 102 (instead of 128 for the uniform distribution). For the minimum rank distribution (middle plot), the median rank is at ≈ 6, which has to be compared to a value of 10 that would be obtained for a uniform distribution with $N_s = 16$. As for the median of the maximum rank, it moved to ≈ 240 (whereas it would be at 245 for a uniform distribution with $N_s = 16$). In our experiments, the lowest maximum rank value found was 110. This essentially means that with a search complexity of $\binom{110}{16} \cdot 16! \approx 2^{107}$, the correct key is found with probability $\approx 1/40000 \approx 2^{-15}$. In fact, already for $N_s > 4$ and even when seeing all possible leakages in a noise-free Hamming weight scenario, the guessing entropy is close to 7 and the rank distribution close to uniform.

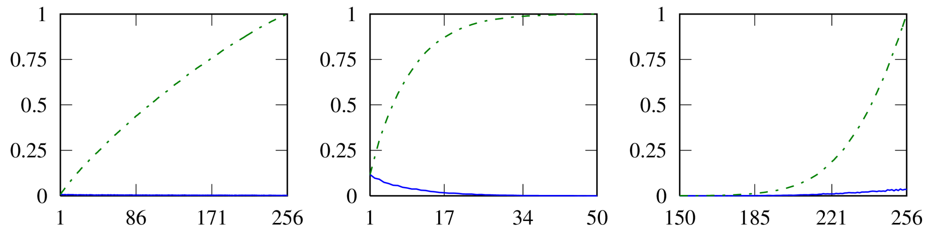

Fig. 6. Rank distributions for unknown plaintexts with $N_s = 16$. Average rank of subkey k_1 (left), average minimum rank amongst all k_j's (middle) and average maximum rank amongst all k_j (right)

4.3 Explaining the results: analysis of model errors

Both for the carefully-chosen plaintext scenario as well as for the unknown-input scenario, we are not able to perfectly model the leakage distribution without knowing the key. This is because, we have no means of marginalizing the distributions for the not-targeted subkeys as explained by Equations (2) and (3). In other words, due to the key-dependent algorithmic-noise, we inevitably build somewhat incorrect templates. In order to further explain our results, we now investigate how significant our model errors become with large N_s. For the carefully-chosen plaintext scenario, we saw that for small values of N_s (≤ 8), the average rank was at an optimum of $(N_s + 1)/2$. This suggests that the model errors are still tolerable, as illustrated in the left and middle plots of Fig. 7. These figures show the statistical distance between the true leakage distribution $\mathcal{D}$ and the models $\mathcal{D}^{k_j^*}$ for $N_s = 4$ and $N_s = 8$. For measuring the distance we computed one line of the mutual information matrix as defined in [10], corresponding to one used key. This metric was chosen because it directly reflects what will happen in a template attack. Namely, the key candidate $\mathcal{D}^{k_j^*}$ which is closest to $\mathcal{D}$ (i.e. has the highest value for the metric) will eventually be rated first (if enough measurements are exploited). The distances between $\mathcal{D}$ and $\mathcal{D}^{k_j}$ are marked by a red x for the N_s correct subkeys. They are indeed maximum in the left and middle plots, for $N_s = 4$ and $N_s = 8$. By contrast, for $N_s = 16$ in the rightmost plot, only seven of the 16 correct subkeys are ranked first. Although these plots only show the effect for a specific set of subkeys, it already confirms that the average rank has to be higher than $(N_s + 1)/2$.

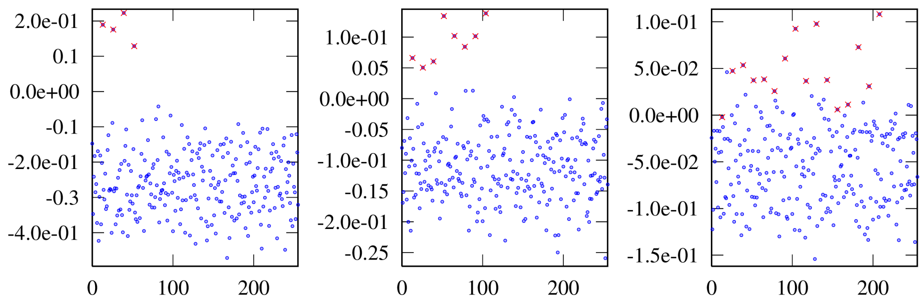

Fig. 7. Distance between $\mathcal{D}$ and $\mathcal{D}^{k_j^*}$ for carefully-chosen plaintexts. The device holds the subkeys k_j marked by the red x. As the distance is measured by the entries of the mutual information matrix, a higher value on the y-axis indicates a smaller distance. From left to right the scenarios for $N_s = 4, 8$, and 16 are depicted.

In the unknown plaintext scenario, we additionally need to estimate a second-order moment of a bi-variate distribution. From studies on masking, we know that such distributions are much more susceptible to noise [29]. Furthermore, in our case the relation between the leakage samples is not straightforward, as for affine or multiplicative masking [12]. Both circumstances suggest that the key dependent noise should cause more severe model errors and indeed, this is what can be observed in Fig. 8. Be aware that this time, the leftmost plot depicts the case for $N_s = 2$ and even there already none of the correct subkeys is ranked first. As we move to higher values for N_s, it can also be seen that the distances themselves become much smaller. As a consequence, measurement noise (remember that until now all experiments were performed without noise) and the inability to calculate the templates will make attacks even harder, as will be discussed in Section 5.3.

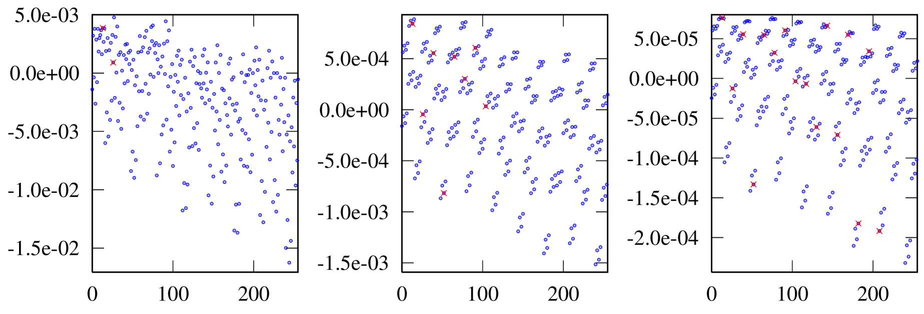

Fig. 8. Distance between $\mathcal{D}$ and $\mathcal{D}^{k_j^*}$ for unknown plaintexts. From left to right the scenarios for $N_s = 2, 8$, and 16 are depicted.

5 Implementation and attack issues

The previous evaluations of our new construction assumed bi-variate noise-free Hamming weight leakages and perfectly calculated templates. In this section we want to address the violation of these assumptions in a real world implementation and attack scenario. We start by analyzing the deviation from Hamming weight leakages, then discuss the case of transition-based leakages (aka Hamming distance model), and finally look at the impact of a more realistic (bounded) template estimation phase.

5.1 Deviations from the Hamming weight leakage function

In this section we show that the previous experiments based on Hamming weight leakages are appropriate and sufficient to argue about the security of our construction, even if such leakages are not accurately met in a real world application. We do so by exploring different power models. In particular, we choose power models with low and high resolution and with low and high non-linearity. As for the resolution, we choose the leakage functions to be the Hamming weight function (hw), the Hamming weight function plus quadratic terms (quad), and as the identity function (id). As for the non-linear leakage function we chose the Hamming weight function preceded by an AES S-box (nlhw). In addition, we target two kinds of S-boxes, the AES S-box (AES) and an identity function S-box (ID8). The latter one would correspond to directly attacking the key addition layer of the AES. In Fig. 9 we compare the rank distributions for these various scenarios and $N_s = 2, 4$ and 16. For $N_s = 2$ it can be seen that the non-linearity of the target function is of higher importance than the one of the leakage function. The non-linearity of the leakage function only helps significantly for linear target functions (ID8), while for (AES) the impact is minor. Most importantly, we see that as soon as N_s increases, the impact of all these combinations of leakage functions and targets vanishes, confirming our claims.

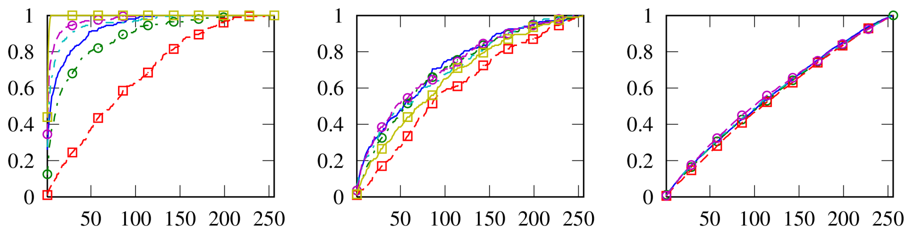

Fig. 9. Comparison of different combinations of target functions and power models: AES+id (yellow,s/s), AES+quad (magenta,d/c), AES+nlhw (cyan,dd/), AES+hw (blue,s/), ID8+nlhw (green,dd/c), ID8+hw (red,d/s). From left to right the scenarios for $N_s = 2, 4$, and 16 are depicted.

5.2 Distance-based leakages

In practice, a cryptographic implementations can be flawed because an adversary sees leakages which are not covered by the theoretical analysis. This can be due to glitches, early propagation, or most deadly for Boolean masking, unintentional distance-based leakage [2,8]. That is, a secret shared as $(s \oplus m, m)$ leaks via $HD(s \oplus m, m)$. Such leakage can occur if a register holding the first share is overwritten with the second share. Another scenario, where the adversary might get an advantage is if he can perform a normalized product combining before summing up the leakage points. This can be the case for a weakly shuffled software implementation which handles the key addition and the S-box operations together. Interestingly, we can show experimentally that none of these implementation issues represent a threat in the unknown-input case. From Fig. 10 (which contains the rank distributions of our construction in the context of uni-variate Hamming weight leakages corresponding to the XOR between the two intermediate values of our previous bi-variate distributions) we see that this Hamming distance case already performs badly for small values of N_s, whereas the normalized product combining still gives a slight advantage due to the reduced noise impact. Again, the higher the value of N_s becomes, the more forgiving the scheme becomes w.r.t. implementation weaknesses.

5.3 Bounded template estimation

Besides the previously studied key-dependent algorithmic noise, another standard source of errors for templates is poor estimation. Usually, one exhaustively

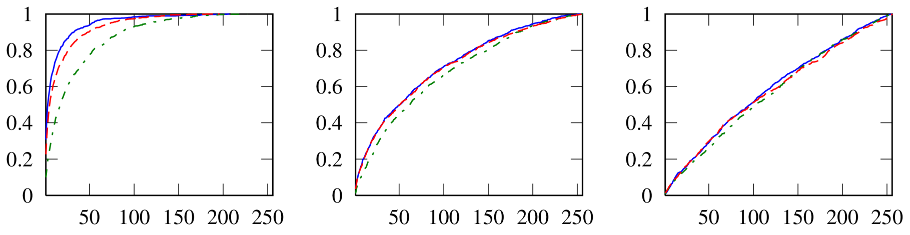

Fig. 10. Comparison of the rank distribution when attacking a standard bi-variate (red,d), a Hamming distance-based (green,dd) and a normalized product combining (blue,s) based leakage distribution. From left to right the scenarios for $N_s = 2, 4,$ and 16 are depicted.

acquires traces for all inputs. In practice, this is not possible as the number of inputs grows exponentially with N_s, but usually good enough if the number of traces is sufficiently large.[6] In Fig. 11 we can see that this is not the case for unknown-inputs. We compare the rank distribution for an attack with dependent noise to an attack with independent noise but with insufficiently sampled templates. For the left plot with $N_s = 2$, 2^{26} traces for template building yield a smaller error than key dependent noise, but still do not allow to recover the key with certainty as in the left plot of Fig. 5 where the templates where calculated. Using only 2^{22} traces already leads to a larger model error than dependent noise. Finally, for $N_s = 4$ the calculated templates for dependent noise already perform best.

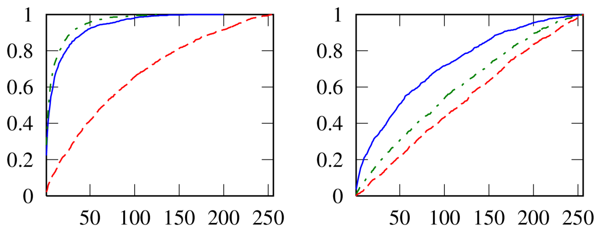

Fig. 11. Rank distribution for calculated templates with dependent noise (blue,s) and estimated templates with independent noise. For the dotted dashed green line the templates were estimated using 2^{26} traces, for the dashed red one using 2^{22} traces. $N_s = 2$ in the left plot and $N_s = 4$ in the right plot.

6 Alternative attack paths

In this last section, we finally mention two alternative attack paths that could be considered against our construction. These are iterative DPA attacks (as

[6] One could overcome this insufficiency by building the templates for the S-boxes independently and afterwards combine them like we did in our simulations. However, the errors for the $\mathcal{D}^i$s will multiply when calculating the overall template and therefore the overall error will grow exponentially with N_s.

they represent the strongest attack against the CHES 2012 construction) and attacks to recover the plaintexts (as our security is based on their secrecy). While we leave their detailed analysis as a scope for further research, we provide concrete arguments showing that they have limited chances of success for realistic adversaries. Finally, we also discuss localized EM attacks.

6.1 Iterative DPA attacks and key verification

In [20], an iterative attack was described which allows to recover the 16 subkeys up to their order (the best result one can hope for) by successively removing the dependent noise in an iterative DPA. In this attack the authors exploited the fact that the first ranked key was always one of the correct ones and thus could be used to model the key dependent noise in the next iteration. Thus virtually, the parameter N_s was reduced by one in each iteration. The complexity of the iterative DPA is $2^8 \cdot q \cdot N_s = 2^{20}$ for AES (2^8 key candidates and $N_s = 16$, thus 16 iterations) while assuming that in a noise-free case $q = N_p = 2^8$ traces. Afterwards, the enumeration costs are $16! \approx 2^{44}$. Key verification during enumeration is straightforward since the plaintexts are known.

In the unknown-input case we could follow a similar strategy. In order to model the algorithmic noise, induced by already guessed subkeys, we would need to construct the templates freshly in every iteration. On top of that we cannot just take the first subkey candidate but need to exhaust the lists up to a certain threshold.[7] To estimate the effort of this, we multiply the medians of the ranks for the best ranked k_j for $N_s = 1 \dots 16$. The result is that with a probability of 2^{-16} we recover the correct key set after $\approx 2^{37}$ iterations. Each iteration comprises 16 template building and attack operations which in turn has a complexity of $\approx 2^{28}$(at least 2^{20} traces and 2^8 keys) each. Thus, investing around $2^{37+4+28} = 2^{69}$ one can recover the subkey bytes up to permutation. Ordering them costs again $16! \approx 2^{44}$. Note, that unlike for the carefully-chosen input case, the result of the iterative DPA is not conclusive and therefore has to be multiplied by the ordering effort. In fact, it would be even less complex to directly exhaust for the subkeys rather than the subkey set. Based on the medians for the actual ranks of k_j for $j = N_s = 1 \dots 16$ this would result in a complexity of 2^{90}. Finally, after going through all this effort, one still has no means of verifying whether the correct key was found as one needs at least one secret plaintext to verify the key based on a known answer. As recovering a plaintext is as hard as recovering a key with the assumed unbounded data complexity and both need to be jointly verified, the effort squares.

6.2 Attacks on the plaintexts

Attacks on the key are restricted to N_p traces in practice. As the plaintexts need to be precomputed, N_p will take values between 2^4 and 2^{16}. As an adversary

[7] Be aware that key enumeration algorithms do not work here since the lists are dependent and thus no full key sorting according to probabilities is possible.

cannot launch a meaningful attack on the key with this restriction, he might instead target the plaintexts. This can be done by randomizing the PRF input for all iterations of the tree except for the last one. Hence, in the last iteration the key will be randomized, but the plaintext will be fixed. This in turn switches the role of the key and the plaintext in the attack and leads to a virtually unbounded data complexity. Recovering sufficient plaintexts following this strategy, a standard DPA on the key could be mounted.

Our previous analysis shows, that even with unlimited data complexity, one is far from recovering a key or a plaintext. Thus, for the standard DPA, the plaintext bytes for building the hypotheses are uncertain and have to be guessed. Let us first assume, that only one subkey byte is targeted. The adversary then needs to pick the plaintext byte for each trace from a set. Without side-channel information, this set would have a size of 256. From Fig. 6 we know that with a 50% probability, the plaintext byte is contained in a set of 240 entries after an unbounded attack. Thus, overall in an attack where r plaintexts are used, $2^8 \cdot 240^r$ hypotheses have to be built. Even then, the probability that the correct plaintext bytes are contained is only 2^{-r}. Therefore, this seems to be a rather futile attack path.

6.3 Localized EM attacks

We analyze the impact of localized EM attacks by reducing N_s. The simulation is performed for the generation of the secret plaintexts (two traces per key) and the PRF evaluation. For the latter, we look at attacks on the key (16 traces) and on the secret plaintexts (unlimited traces). Both scenarios are studied for the Hamming weight and for the ID leakage function.[8] Even though the same information can be extracted from Fig. 3, 5, and 9, we present a, for our purpose, more representative cross-section of these.

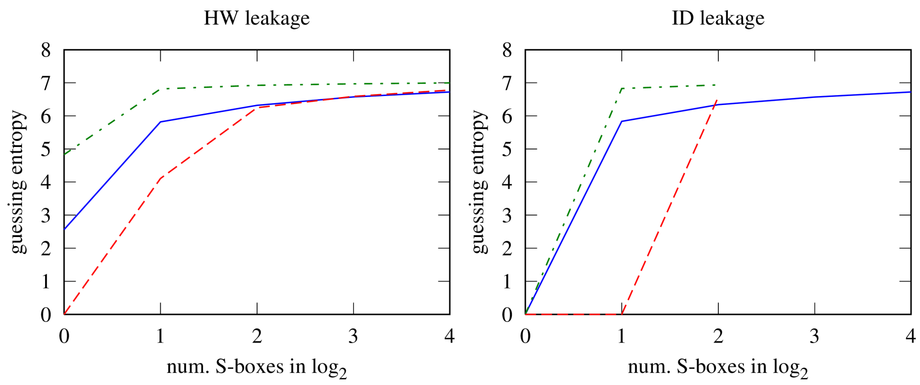

Fig. 12. Average guessing entropy after attacks with two known plaintexts (blue,s/), 16 unknown plaintexts (green,dd/c), unlimited unknown plaintexts (red,d/s).

[8] As before, the ID leakage experiments were only carried out for up to 4 parallel S-boxes due to the prohibitive simulation complexity for larger values of N_s.

In Fig. 12 it can be seen that attacks on the key during the PRF evaluation are the least informative ones. Even for ID leakage, two parallel S-boxes are sufficient to raise the guessing entropy to a value close to seven. For recovering the secret plaintexts (unlimited traces), the situation is less clear, but remember from Section 6.2 that the complexity of such an attack grows exponentially with the number of plaintexts that need to be recovered. As a result two parallel S-boxes are sufficient in the HW case and for the ID case (notably uncommon in practice) three to four are required. Finally, attacking the plaintext generation seems the be the most promising strategy. Yet, already with $N_s = 2$ both scenarios lead to a considerable guessing entropy close to six. This is a quite positive result as we cannot do better than touching a key twice and therefore anyway need to protect this part sufficiently. Note that, even if we require $N_s = 4$, this is a much stronger result than in [20] where $N_s \geq 24$ was suggested.

Now this leads us to the question how local EM attacks can be, that is, how few S-boxes can be targeted at once with an EM micro probe. Unfortunately, the only work studying such distinguishability so far aimed at a 90nm FPGA implementation of a block cipher with 32 4-bit S-boxes [4]. From Fig. 5 in [4] one can see that the area in which leakage is observed covers approximately $1mm^2$. An AES implementation which suits our purposes in $40nm$ technology on the other hand can be expected to cover only $10,7kGE * 0.71\mu m^2/GE \approx 7600\mu m^2$. Given that even in [4], the S-boxes could not be fully separated, we expect such an attack to be difficult. However, we leave this question to future research.

7 Implementation figures

In this section we would like to discuss an implementation of the primitive and its performance. The AES coprocessor has been implemented using a 32-bit datapath with 4 S-boxes. All 32-bit operations (SubBytes, AddKey, MixCols) are performed in one cycle per column. ShiftRows and the key schedule are 128-bit operations and have been separated in order to minimize the power consumption. This results in a cost of 6 cycles per round and 66 in total. In addition the coprocessor features an IO register which allows to e.g. transfer data between the IO and the data or the key register within 4 cycles. The latter one has been implemented for a fast ciphertext to key transfer during the PRF evaluation. The total area of the coprocessor including the SFR interface accounts for 10.7 kGE.

Also, thanks to the IO register, loading data from the CPU to the coprocessor (32 cycles) can be done while the coprocessor is busy and the delay caused by the CPU between encryptions can be kept low. In total it takes therefore 2951 cycles to pre-compute the secret plaintexts and 2775 cycles for a PRF evaluation including a fault-protected final transformation. As a comparison, a fault protected AES takes four data operations (load key, load/unload data, compare) and two cipher operations, thus a total of 250 cycles. Hence, even though, one PRF evaluation takes 34 AES calls, in our architecture it only takes 11 times longer than a fault-protected AES. Since symmetric cryptographic operations are usually not the dominating part in an application, the overhead decreases with every abstraction layer. That is, when looking at the C function API level during a mutual authentication based on ISO-9798-4 (MAC based authentication), the overhead already decreases to a factor of 4. On an OS level or even transaction level (including communication overheads), the factor would decrease further.

Security wise, we addressed the need for parallelism against localized EM attacks in a hybrid way. On the one hand side, we implement a 32-bit datapath, that is, $N_s = 4$. On the other hand, we implement a four-fold shuffling, virtually resulting in $N_s = 16$. Thanks to this design, even if an adversary would be able to exploit localization, we still have time randomization as a backup. Furthermore, attacks on shuffling itself become more unlikely with increasing noise [31], which we take care of by widening the data path.

8 Conclusions

In this work we presented a leakage-resilient PRF which makes use of parallel block cipher implementations with unknown-inputs. To the best of our knowledge this is the first work to study and exploit this form of key-dependent algorithmic noise. It turns out that it renders the problem of side-channel key recovery intractable, even in a noise-free setting and independent of the number of traces and the used power model.

Thanks to this security improvement over the CHES 2012 construction, standardized algorithms like the AES can be used in our construction. Moreover, our analysis suggests that even localized EM attacks can be tolerated to some degree. That is, even if an EM probe would only catch the signal of 8 or 4 S-boxes, the attack would not suddenly become trivial. On top of that, we showed that opposed to the previous construction, the strong side-channel resistance holds throughout the entire algorithm and not only for the first round's S-box layer.

We also showed that these results hold even if actual implementations show leakage behaviors that significantly deviate from our experimental conditions. In fact, the security of our construction essentially relies on secret inputs and nothing else. Yet, and as usual, it will additionally benefit from any concrete limitation of the quality of the templates, e.g. due to a bounded number of traces for profiling and/or electrical noise.

Finally, and from a performance point of view, our new construction allows to use larger values for N_p than the size of the S-box. In practice, it will be quite application specific whether large values for N_p pay off (depending on the memory available for pre-computations). However, at least for block ciphers which use small S-boxes, like e.g. PRESENT, the construction should lead to a significant performance increase over the CHES 2012 one.

References

1. Michel Abdalla, Sonia Belaïd, and Pierre-Alain Fouque. Leakage-resilient symmetric encryption via re-keying. In Guido Bertoni and Jean-Sébastien Coron, editors,

Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings, volume 8086 of *Lecture Notes in Computer Science*, pages 471–488. Springer, 2013.

2. Josep Balasch, Benedikt Gierlichs, Vincent Grosso, Oscar Reparaz, and François-Xavier Standaert. On the cost of lazy engineering for masked software implementations. In Joye and Moradi [16], pages 64–81.
3. Sonia Belaïd, Vincent Grosso, and François-Xavier Standaert. Masking and leakage-resilient primitives: One, the other(s) or both? *Cryptography and Communications*, 7(1):163–184, 2015.
4. Sonia Belaïd, Fabrizio De Santis, Johann Heyszl, Stefan Mangard, Marcel Medwed, Jörn-Marc Schmidt, François-Xavier Standaert, and Stefan Tillich. Towards fresh re-keying with leakage-resilient PRFs: Cipher design principles and analysis. *J. Cryptographic Engineering*, 4(3):157–171, 2014.
5. Begül Bilgin, Benedikt Gierlichs, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Higher-order threshold implementations. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 326–343. Springer, 2014.
6. Begül Bilgin, Benedikt Gierlichs, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. A more efficient AES threshold implementation. In David Pointcheval and Damien Vergnaud, editors, *Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings*, volume 8469 of *Lecture Notes in Computer Science*, pages 267–284. Springer, 2014.
7. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 1999.
8. Jean-Sébastien Coron, Christophe Giraud, Emmanuel Prouff, Soline Renner, Matthieu Rivain, and Praveen Kumar Vadnala. Conversion of security proofs from one leakage model to another: A new issue. In Werner Schindler and Sorin A. Huss, editors, *Constructive Side-Channel Analysis and Secure Design - Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings*, volume 7275 of *Lecture Notes in Computer Science*, pages 69–81. Springer, 2012.
9. Yevgeniy Dodis and Krzysztof Pietrzak. Leakage-resilient pseudorandom functions and side-channel attacks on feistel networks. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 21–40. Springer, 2010.
10. Alexandre Duc, Sebastian Faust, and Franois-Xavier Standaert. Making masking security proofs concrete or how to evaluate the security of any leaking device (extended version). Cryptology ePrint Archive, Report 2015/119, 2015. `http://eprint.iacr.org/`.
11. Sebastian Faust, Krzysztof Pietrzak, and Joachim Schipper. Practical leakage-resilient symmetric cryptography. In Prouff and Schaumont [25], pages 213–232.
12. Guillaume Fumaroli, Ange Martinelli, Emmanuel Prouff, and Matthieu Rivain. Affine masking against higher-order side channel analysis. In *Proceedings of the*

17th International Conference on Selected Areas in Cryptography, SAC'10, pages 262–280, Berlin, Heidelberg, 2011. Springer-Verlag.

13. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, August 1986.
14. Vincent Grosso, Romain Poussier, François-Xavier Standaert, and Lubos Gaspar. Combining leakage-resilient prfs and shuffling - towards bounded security for small embedded devices. In Joye and Moradi [16], pages 122–136.
15. Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481. Springer, 2003.
16. Marc Joye and Amir Moradi, editors. *Smart Card Research and Advanced Applications - 13th International Conference, CARDIS 2014, Paris, France, November 5-7, 2014. Revised Selected Papers*, volume 8968 of *Lecture Notes in Computer Science*. Springer, 2015.
17. Stefan Mangard, Thomas Popp, and Berndt M. Gammel. Side-channel leakage of masked CMOS gates. In Alfred Menezes, editor, *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings*, volume 3376 of *Lecture Notes in Computer Science*, pages 351–365. Springer, 2005.
18. Stefan Mangard, Norbert Pramstaller, and Elisabeth Oswald. Successfully attacking masked AES hardware implementations. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, volume 3659 of *Lecture Notes in Computer Science*, pages 157–171. Springer, 2005.
19. Luke Mather, Elisabeth Oswald, and Carolyn Whitnall. Multi-target DPA attacks: Pushing DPA beyond the limits of a desktop computer. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 243–261. Springer, 2014.
20. Marcel Medwed, François-Xavier Standaert, and Antoine Joux. Towards super-exponential side-channel security with efficient leakage-resilient PRFs. In Prouff and Schaumont [25], pages 193–212.
21. Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Moni Naor, editor, *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, volume 2951 of *Lecture Notes in Computer Science*, pages 278–296. Springer, 2004.
22. Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the limits: A very compact and a threshold implementation of AES. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 69–88. Springer, 2011.
23. Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure hardware implementation of nonlinear functions in the presence of glitches. *J. Cryptology*, 24(2):292–321, 2011.
24. Olivier Pereira, François-Xavier Standaert, and Srinivas Vivek. Leakage-resilient authentication and encryption from symmetric cryptographic primitives. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd*

ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015, pages 96–108. ACM, 2015.
25. Emmanuel Prouff and Patrick Schaumont, editors. *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *Lecture Notes in Computer Science*. Springer, 2012.
26. Matthieu Rivain and Emmanuel Prouff. Provably secure higher-order masking of AES. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*, pages 413–427. Springer, 2010.
27. François-Xavier Standaert, Olivier Pereira, and Yu Yu. Leakage-resilient symmetric cryptography under empirically verifiable assumptions. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 335–352. Springer, 2013.
28. François-Xavier Standaert, Olivier Pereira, Yu Yu, Jean-Jacques Quisquater, Moti Yung, and Elisabeth Oswald. Leakage resilient cryptography in practice. In Ahmad-Reza Sadeghi and David Naccache, editors, *Towards Hardware-Intrinsic Security - Foundations and Practice*, Information Security and Cryptography, pages 99–134. Springer, 2010.
29. François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The world is not enough: Another look on second-order DPA. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 112–129. Springer, 2010.
30. Nicolas Veyrat-Charvillon, Benoît Gérard, Mathieu Renauld, and François-Xavier Standaert. An optimal key enumeration algorithm and its application to side-channel attacks. In Lars R. Knudsen and Huapeng Wu, editors, *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers*, volume 7707 of *Lecture Notes in Computer Science*, pages 390–406. Springer, 2012.
31. Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 740–757. Springer, 2012.
32. Yu Yu and François-Xavier Standaert. Practical leakage-resilient pseudorandom objects with minimum public randomness. In Ed Dawson, editor, *Topics in Cryptology - CT-RSA 2013 - The Cryptographers' Track at the RSA Conference 2013, San Francisco,CA, USA, February 25-March 1, 2013. Proceedings*, volume 7779 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2013.

A Template building algorithms

In the algorithms below $conv(\cdot,\cdot)$ refers to the discrete convolution function.

Algorithm 1 Template construction for known inputs

```
Require: $M = 256$, $N_s$
Ensure: Templates
  //Build template for each key
  for $k = 0 \ldots M-1$ do
    $\mathcal{D}^k = 0$
    $\mathcal{D}^k(\mathsf{L}(\mathsf{S}(k))) = 1$
  end for
  //Calculate the algorithmic noise contribution of an S-box as the marginal distribu-
  tion
  $H = 0$
  for $k = 0 \ldots M-1$ do
    $H += \mathcal{D}^k/M$
  end for
  //Perform $N_s - 1$ convolutions with the marginal distribution
  for $k = 0 \ldots M-1$ do
    for $i = 1 \ldots N_s - 1$ do
      $\mathcal{D}^k = conv(\mathcal{D}^k, H)$
    end for
  end for
```

Algorithm 2 Template construction for unknown-inputs

```
Require: $M = 256$, $N_s$
Ensure: Templates
  //Build template for each key
  for $k = 0 \ldots M-1$ do
    for $p = 0 \ldots M-1$ do
      $\mathcal{D}^k(\mathsf{L}(p), \mathsf{L}(\mathsf{S}(p \oplus k))) += 1/M$
    end for
  end for
  //Calculate the algorithmic noise contribution of an S-box as the marginal distribu-
  tion
  $H = 0$
  for $k = 0 \ldots M-1$ do
    $H += \mathcal{D}^k/M$
  end for
  //Perform $N_s - 1$ convolutions with the marginal distribution
  for $k = 0 \ldots M-1$ do
    for $i = 1 \ldots N_s - 1$ do
      $\mathcal{D}^k = conv(\mathcal{D}^k, H)$
    end for
  end for
```