Spectral Approach to Process the (Multivariate) High-Order Template Attack against Any Masking Scheme

Maamar Ouladj[1], Sylvain Guilley[2,3,4,*], Philippe Guillot[1] and Farid Mokrane[1]

[1] LAGA, UMR 7539, CNRS, Université de Paris VIII,
2 Rue de la liberté, 93200 Saint Denis, France.
[2] Secure-IC S.A.S., 15 rue Claude Chappe, Cesson-Sévigné, France.
[3] Télécom Paris, Institut Polytechnique de Paris, COMELEC Department, France.
[4] École Normale Supérieure (ENS), 45 rue d'Ulm, Département d'informatique,
CNRS, PSL University, 75005 Paris, France.
[*] ORCID: 0000-0002-5044-3534

E-mails: ouladj.maamar@gmail.com, sylvain.guilley@secure-ic.com,
philippe.guillot@univ-paris8.fr, abdellah.mokrane@gmail.com

Accepted in the Journal of Cryptographic Engineering (JCEN).
DOI: 10.1007/s13389-020-00253-4

Abstract

Cryptographic software is particularly vulnerable to side-channel attacks when programmed in embedded devices. Indeed, the leakage is particularly intense compared to the noise level, making it mandatory for the developer to implement side-channel attack protections. Random masking is a customary option, but in this case, the countermeasure must be high-order, meaning that each sensitive variable is splitted into multiple (at least two) shares. Attacks therefore become computationally challenging.

In this paper, we show that high-order template attacks can be expressed under the form of a convolution. This formulation allows for a considerable speed-up in their computation thanks to fast Fourier transforms. To further speed-up the attack, we also provide an interesting multi-threading implementation of this approach. This strategy naturally applies to template attacks where the leakage of each share is multivariate. We show that this strategy can be adapted to several masking schemes, inherently to the way the splitting is realized. This technique allows us to validate multiple very high-order attacks (order of some tens). In particular, it revealed a non-trivial flaw (hard to detect otherwise) in a multivariate extension of the DSM masking (and subsequently to fix it, and validate its rationale).

Key words: Template attacks, Masking schemes, High-order attacks, Convolution, Fourier transform, Walsh-Hadamard transform, Masking scheme flaw detection.

1 Introduction

Cryptographic devices manage secret keys, which must be protected against extraction. One stealthy attack consists in the analysis of side-channel leakage. As a countermeasure, cryptographic computations can be randomly masked.

The extent of protection bestowed by random masking has been the topic of extensive research. Profiling attacks have proven to be the most efficient, as they model the leakage with great accuracy and subsequently resort to maximizing the key extraction likelihood. Such attacks are known as *template attacks* [18]. When confronted to randomly masked implementations, they can be tailored, as explained for instance in [37]. This paper shows that each share (random split of sensitive variables) can have its leakage profiled, and the resulting high-order template attack recombines all the profiles in order to maximize the key extraction probability.

1.1 Related Works

The seminal paper of Template Attacks (TA) is based on the maximum likelihood principle [18]. Given the importance of masking, several extensions of the TA have been introduced to defeat (or to assess) this countermeasure [37, 30].

In 2014 the Higher-Order Optimal Distinguishers, abridged HOOD in the sequel, have been introduced against the masking countermeasure, whatever its order [9]. It is dubbed optimal in that it consists in computing a maximum likelihood. Nevertheless, its complexity is exponentially linked to the masking order. Thereby, their authors considered only a monovariate distribution (one time sample per share). In practice, side-channel traces are multivariate, because oscilloscopes capture waveforms for each sensitive variable manipulation. Therefore, the HOOD as per [9] is far from being the most suitable distinguisher (for one time sample per share), since we expect that an increase of the number of time samples per share can significantly improve the success rate of the attack [28].

Recently, the optimal side-channel attacks for multivariate leakages and multiple models are introduced [8]. Their inconveniences are twofold. First, this modelling is over-constrained, as there is an additional assumption of a relation between the templates (a coefficient α between each pair of templates' mean). Second, its computation has an exponential complexity according to the masking order.

Subsequently, a Taylor expansion of the maximum likelihood distinguisher has been introduced to reduce its complexity [10]. In this paper the processing is simplified in pre-computations (rounded at a given order, at the expense of the templates precision). The online attack computations are therefore efficient.

More recently a new approach is introduced in [38, §4.4.2]. Its core idea is to combine the leaking shares into one *artificial* trace (inspired from the *normalised* product combination [41] of each share's leakage), for carrying out a template attack efficiently in terms of implementation complexity. Its inconvenient is the fact that this combination could loose some information about the leakage, especially for low-noise devices [42].

The work presented in our paper shows that neither the rounding of attacks nor the shares combination are necessary for computational reasons. Besides, this result holds for any masking scheme.

1.2 Contributions

High-order masking schemes consist in randomly splitting each sensitive variable into several shares, so as to make attacks more difficult (it can be proven that attacks are exponentially complex with the number of shares, in terms of number of traces [17, 40]). Those shares can be linearly combined to carry out computations on them (whilst remaining protected by random masking). Typically, this property also allows to demask at the end of the computation, when all shares need to be aggregated to recover a ciphertext.

The same property is used by an attacker to mount so-called high-order attacks; namely the attacker combines shares so that jointly they contain information on the sensitive variables. The natural combination consists in undoing the sharing, i.e. to conduct a linear combination of the individual shares leakage. For a masking of order d, each tuple of less than or equal to d shares consists in independent variables (i.e., a free family). On the contrary, a tuple of $N = d + 1$ shares has exactly one linear relationship.

In this paper, we leverage this basic noting to conceptualize high-order attacks as a convolution product over the d-dimensional mask space. This convolution stands out in a straightforward manner for Boolean masking. But **it can also be adapted to other masking styles, by the introduction of the relevant variable change**. The linear operation used in the masking scheme impacts the type of convolution, which we capture using the corresponding group law.

This new vision of masking schemes can therefore take advantage of the considerable wealth of *character theory*. In particular, the computation of the side-channel distinguishers can be optimized thanks to Plancherel identity (arising from properties of Fourier transforms).

The contributions of this paper are therefore to:

- Quantify the speed-up in distinguisher computation;
- Show that this speed-up is particularly adapted to the optimal distinguisher (therefore: two advantages for the attacker–minimizing the required number of traces to extract the key [*optimal distinguisher using maximum likelihood*] and minimizing the attack elapsed time and required memory space [*optimal distinguisher computational complexity*]);
- Provide an interesting multi-threading implementation of our approach;
- Apply the attack in the context of multivariate subtraces for the leakage of each share;
- Illustrate quantitatively this new attack paradigm on (very) high-order masking schemes, including **six** styles (Boolean, IPM, DSM, Polynomial DSM, RSM, leakage squeezing);
- Introduce a new multi-share extension of DSM (along with the optimal attack on it).

1.3 Outline

The rest of the paper is structured as follows. Mathematical notions and results useful to present to paper contributions are gathered in Sec. 2. Our innovative algorithmic optimization of the high-order multivariate distinguisher is presented in Sec. 3, on the simple exemple of Boolean Masking. Its adaptation per masking type is the topic of Sec. 4. In particular, Boolean, i.e. additive, multiplicative and affine masking schemes, are addressed in Sec. 4.1 (which covers all these schemes altogether exploiting the fact they are instances of the IPM scheme which generalizes them). DSM and its multi-share extension (MS-DSM) are treated respectively in Sec. 4.2 and 4.3. The polynomial extension of DSM, which (as DSM) allows for both fault detection and side-channel attack protection, is the topic of Sec. 4.4. The Rotating Substitution-box Masking is analyzed in Sec. 4.5. Eventually, leakage squeezing is addressed in Sec. 4.6. Validation of the theory on simulated traces is the topic to Sec. 5. Eventually, conclusions and perspectives are given in Sec. 6. The appendix A explains the flaw of the straightforward extension of DSM to multi-share case, and also provides the subsequent fix. The appendix B provides the success rate curves of the attacks on various masking schemes.

2 Preliminaries

2.1 Linear Algebra and Linear Codes

Definition 1 (complementary subspaces)**.** *Let n be a non-zero integer. Let $\mathbb{C}$ and $\overline{\mathbb{C}}$ be two subspaces (seen as linear codes) of a vector space $\mathbb{F}_2^n$. By definition $\mathbb{C}$ and $\overline{\mathbb{C}}$ are complementary subspaces, if and only if, any element of the vector space $\mathbb{F}_2^n$ can be decomposed in a unique way as a sum of two elements of $\mathbb{C}$ and $\overline{\mathbb{C}}$. We write $\mathbb{F}_2^n = \mathbb{C} \oplus \overline{\mathbb{C}}$.*

Remark 1. *The supplementary subspace of a given subspace is not unique and it is not an orthogonal subspace in the general case. If a linear code $\mathbb{C}$ has an orthogonal supplementary code (a dual code denoted $\mathbb{C}^\perp$), then $\mathbb{C}$ is called a Linear Complementary Dual (LCD) code, and given their dimension, we have that the so-called hull $\mathbb{C} \cap \mathbb{C}^\perp$ of $\mathbb{C}$ is trivial, namely $\mathbb{C} \cap \mathbb{C}^\perp = \{0\}$ [33, 35].*

By the rank-nullity theorem, if the dimension of $\mathbb{C}$ equals $n_0 < n$, then the dimension of any supplementary subspace $\overline{\mathbb{C}}$ is equal $n - n_0$.

Let us consider generating matrices G and $\overline{G}$ of respectively $\mathbb{C}$ and $\overline{\mathbb{C}}$. We denote $\mathbb{C} = \mathrm{span}(G)$ and $\overline{\mathbb{C}} = \mathrm{span}(\overline{G})$. Then every vector $z \in \mathbb{F}_2^n$ can be written in a unique way as $z = z_1 G \oplus z_2 \overline{G}$,where $z_1 \in \mathbb{F}_2^{n_0}$ and $z_2 \in \mathbb{F}_2^{n-n_0}$.

Definition 2 (minimum distance)**.** *The minimum distance $d_{\mathbb{C}}$ of a linear code $\mathbb{C}$ is the smallest Hamming weight of its nonzero codeword.*

Definition 3 (dual distance)**.** *The dual distance $d_{\mathbb{D}}^{\perp}$ of a linear code $\mathbb{D}$ is $d_{\mathbb{D}}^{\perp} = d_{\mathbb{D}^{\perp}}$.*

2.2 Spectral Theory

As will be shown, instead of computing the distinguisher for each subkey k (or equivalently for each sensitive value z) distinctly, only one computation is sufficient, taking advantage of the properties of the convolution product over a finite group. Let us first recall the definition of a convolution product that can be processed efficiently.

Definition 4 ([43])**.** *Let $(\mathbb{S}, +)$ be a group of size 2^n. Let d be a non-zero integer. Let $f^{(0)}, \ldots f^{(d)}$ be $d+1$ functions from $\mathbb{S}$ to $\mathbb{R}$. A dth-order convolution product of $f^{(0)}$, $\ldots$, $f^{(d)}$ is the function denoted by $f^{(0)} \otimes \cdots \otimes f^{(d)}$ and defined from $\mathbb{S}$ to $\mathbb{R}$ by: for every $z \in \mathbb{S}$,*

$$(f^{(0)} \otimes \cdots \otimes f^{(d)})(z) \doteq \sum_{j_1 \in \mathbb{S}} \cdots \sum_{j_d \in \mathbb{S}} f^{(1)}(j_1) \ldots f^{(d)}(j_d) f^{(0)}(z - \sum_{w=1}^{d} j_w),$$

where $-z$ denotes the symmetric element of z relatively to the $+$ law (i.e. in the group $(\mathbb{S}, +)$).

The naïve computation of the convolution product for all possible z needs $\mathcal{O}(2^{dn})$ multiplications. In a view to speed-up the processing of such convolution, let us also denote the so-called *functions product* by $f^{(1)} \bullet f^{(2)}$ where $f^{(1)} \bullet f^{(2)}(z) \doteq f^{(1)}(z) f^{(2)}(z)$. It is the associative coordinate-wise product. In addition, we denote by FFT the (Fast) Fourier Transform over the group $(\mathbb{S}, +)$. An important property of the convolution product with respect to the FFT is that, for every $z \in \mathbb{S}$, we have [43]:

$$(f^{(0)} \otimes \cdots \otimes f^{(d)})(z) = FFT^{-1}\left(FFT(f^{(0)}) \bullet \cdots \bullet FFT(f^{(d)})\right)(z). \quad (1)$$

The advantage of this property is that one can process the dth-order convolution product for all possible values z at once, with an overall complexity of $\mathcal{O}(dn2^n)$ *additions* instead of $\mathcal{O}(2^{dn})$ *multiplications*. Indeed, the computation of the complete spectrum benefits from the Fast Fourier Transform implemented by a butterfly algorithm.

The two groups of interest for our study are $(\mathbb{F}_2^n, \oplus)$ for which the FFT coincides with the Walsh-Hadamard Transform, and $(\mathbb{F}_{2^n}, +)$ for which it is the Cyclical Fourier Transform.

2.3 Higher-Order Template Attack

This section recalls Boolean Masking and extends state-of-the-art attacks on it to the case the leakage is multivariate leakage. Our approach enabling computational speed-up is presented in Sec. 3. Generalization to other masking schemes is the topic of Sec. 4.

2.3.1 High-Order Boolean Masking

In this section, we introduce the notations. Let us assume an attacker knows the plaintext t and guesses a constant secret key k. They are inputted in a cryptographic algorithm, and their combination gives rise a *sensitive variable* z. For example, $z(t,k) = \texttt{Sbox}(t \oplus k)$, in the case of the attack of a substitution box (in block ciphers such as AES or PRESENT). In general, the function $k \mapsto z(t,k)$ is an injection, for all t.

The sensitive variable z is the target of side-channel attacks. In masking schemes, each z is randomly splitted. Namely, d^{th} higher-order masking consists in computing each sensitive variable $z(t,k)$ separately in $N = d+1$ shares.

For example, in d^{th} order Boolean masking scheme, z is splitted in $(z^{(0)}, \ldots, z^{(d)})$, such that:

$$z = \bigoplus_{w=0}^{d} z^{(w)}. \quad (2)$$

In Eq. (2), the $d+1$ shares are traditionally randomly chosen from d random numbers called masks m_w, with $1 \leq w \leq d$, according to:

- $z^{(1)} = m_1$,
- $z^{(2)} = m_2$,
- $\ldots$
- $z^{(d)} = m_d$,
- $z^{(0)} = z \oplus \bigoplus_{w=1}^{d} z^{(w)}$.

Besides, in Eq. (2), the operation is the bitwise XOR, denoted as "$\oplus$".

In general, such masking adapts to other situations. Typically, one just needs the combination to be a group operation. In the rest of this section, we shall consider that this group is additive. We denote it as $(\mathbb{S},+)$. Notice that multiplicative groups also allow to derive multiplicative masking; therefore the "$+$" sign shall be understood broadly, as the group $\mathbb{S}$ inner operation. Exemples of groups $\mathbb{S}$ are:

1. $(\mathbb{F}_2^n,\oplus)$, used in Boolean masking, or
2. $(\mathbb{F}_{2^n},+)$, such that $+$ is the 2^n-modular addition, which is used in arithmetic masking.

Some cryptographic algorithms make use both of Boolean and arithmetic masking, thus the two masking schemes shall cohabit, and masking conversions shall be implemented [23].

2.3.2 Attack on High-Order Boolean Masking

Template attacks have been introduced as multivariate attacks on unprotected devices in [18]. Their extension to masked implementation has been suggested in [37, 30]. However, template attacks on masking scheme where each share is profiled independently is not clearly described in the state-of-the-art. We discuss it in this paper, which gives the most general attack compared to the existing literature.

During the profiling phase, the adversary can see the leakage measurement X_q as $(d+1)$ disjoint sub-traces denoted by $X_q^{(w)}, 0 \leq w \leq d$. Each sub-trace $X_q^{(w)}$ is a sub-trace of length $D^{(w)}$ and corresponds to the leakage of share $z^{(w)}$. We assume that for each w, $0 \leq w \leq d$, the adversary estimates the multivariate Probability Density Function (PDF) of $X_q^{(w)}$, such that she profiles the device consumption of $D^{(w)}$ samples for the w^{th} share.

Thanks to the profiling result, the adversary can estimate the probability $p(X_q^{(w)}|z^{(w)})$, for each manipulated data $z^{(w)}$. We recall that $z^{(w)}$, for $1 \leq w \leq d$, is equal to mask m_w, and $z^{(0)}$ is equal to $z \oplus m_1 \oplus \cdots \oplus m_d$.

According to [9, Thm 7], when attacking $Q > 0$ traces, the $(d+1)^{th}$-order optimal distinguisher is (for simplicity we assume that all the mask values have the same probability):

$$\mathscr{D}_{opt}^{d} = \underset{k}{\operatorname{argmax}} \prod_{q=1}^{Q} \sum_{m_1,\ldots,m_d \in \mathbb{S}} p(X_q^{(1)}|m_1)\ldots p(X_q^{(d)}|m_d)\, p(X_q^{(0)}|z(t_q,k) \oplus \bigoplus_{w=1}^{d} m_w). \tag{3}$$

This equation is the maximum likelihood estimator. It takes as inputs:

- the known plaintexts $(t_q)_{1 \leq q \leq Q}$ and
- the leakage sub-traces $(X_q^{(0)}, \ldots, X_q^{(d)})_{1 \leq q \leq Q}$,

and returns the most likely key $\hat{k}$ according to the relationship between z, t and k.

In fact, in [9] the HOOD is studied by taking only one sample per share. In our study, we generalize it by assuming $D^{(w)}$ samples for the w^{th} sub-trace, $0 \leq w \leq d$, and considering this sub-trace as a random vector. That means, we generalize the HOOD in same way when one goes from the first-order DPA (one sample per trace to compute the distinguisher) to the (first-order) template attack (D samples per trace but concerning only one sensitive variable z). Here, we migrate from the higher-order DPA to the higher-order template attack.

To compute our distinguisher efficiently, one can see that:

$$\begin{aligned} \mathscr{D}_{opt}^{d} &= \underset{k}{\operatorname{argmax}} \prod_{q=1}^{Q} Sum_q^{\mathsf{BM}}(z(t_q,k)) \\ &= \underset{k}{\operatorname{argmax}} \sum_{q=1}^{Q} \log Sum_q^{\mathsf{BM}}(z(t_q,k)) \end{aligned} \tag{4}$$

where Sum_q^{BM} is a pseudo-Boolean function (meaning a real function defined over Boolean vectors) of 2^n values, equal to:

$$\begin{aligned} Sum_q^{\mathsf{BM}}(z) = & \sum_{m_1,\ldots,m_d \in \mathbb{S}} p(X_q^{(1)}|m_1)\ldots p(X_q^{(d)}|m_d) \\ & p(X_q^{(0)}|z \oplus m_1 \oplus \cdots \oplus m_d) && (5) \\ = & \sum_{m_1 \in \mathbb{S}} \cdots \sum_{m_d \in \mathbb{S}} p(X_q^{(1)}|m_1)\ldots p(X_q^{(d)}|m_d) \\ & p(X_q^{(0)}|z \oplus m_1 \oplus \cdots \oplus m_d) \\ = & \; p(X_q^{(1)}|.) \otimes \cdots \otimes p(X_q^{(d)}|.) \otimes p(X_q^{(0)}|.)(z), && (6) \end{aligned}$$

such that $\otimes$ is the convolution product relatively to the law $\oplus$ in the group $(\mathbb{S},\oplus)$. Clearly, the property that the value of z can be recovered as $z = z^{(0)} - \sum_{w=1}^{d} z^{(w)}$ is central to apparition of the convolution. This property is specific to the Boolean Masking, but extends to all masking schemes, as will be shown in Sec. 4. Indeed, in all masking schemes, there is a relationship between the shares $z^{(0)},\ldots,z^{(d)}$ which allows to recover the sensitive value $z(t,k)$.

3 Our Optimized Higher-Order Template Attack

The former section described the higher-order template attack distinguisher as Eq. (4) and (6). In this section, we show how to implement it in practice.

3.1 Higher-Order Template Attack Setup

The attack setup is depicted in Fig. 1. It consists in several steps:

- First of all, the leakage of each share is profiled (offline); This requires to collect so-called sub-traces $X^{(w)}$ for each share $z^{(w)}$, $0 \leq w \leq d$.

- From this characterization, the probability density functions $p(X^{(w)}|z^{(w)})$ corresponding to the leakage distribution of each share is computed. These template profile functions can be computed efficiently, as shown later in Sec. 3.3.

- Then, the probability density functions are combined by a convolution product. As explained in Sec. 3.2, this computation can implemented very efficiently leveraging the Walsh-Hadamard transform.

In the figure 1, the leakage of each share is represented as a bivariate Gaussian, for the sake of illustration. However, this setup works well for higher multiplicity, and even for different multiplicities $D^{(w)} > 0$ for each share w, $0 \leq w \leq d$. This figure represents the attack on Boolean Masking with $N = d+1$ shares. Adaptation to other masking schemes is the topic of Sec. 4.

3.2 Walsh-Hadamard Transformation to Speed-Up Convolution Computation

The convolution which we exhibit in Sum_q^{BM}, namely in Eq. (6), allows us to compute, for every q, all the 2^n values of $Sum_q^{\mathsf{BM}}(z)$, $z \in \mathbb{F}_2^n$, at once. Indeed, the Fast Walsh-Hadamard relies on a butterfly algorithm of complexity $nd2^n$, whereas the naïve computation has complexity 2^{nd} (direct multiple summation over $\mathbb{F}_2^n \times \ldots \times \mathbb{F}_2^n$, as per Eq. (5)).

The overall attack is described in Alg. 1). It can be noticed that the line 10 assumes that the function $k \mapsto z(t,k)$ is injective.

```
Input  :
    • Templates: for example $(Y_z^{(w)})_{0 \leq z < 2^n}$ and $\Sigma^{(w)}$;
    • Sub-traces: $(X_q^{(0)},\ldots,X_q^{(d)})_{1 \leq q \leq Q}$;
    • Plaintexts: $T = (t_q)_{1 \leq q \leq Q}$,
Output : The most likely key $\mathscr{D}_{opt}^{d} \in \mathbb{F}_2^n$.
1  $Distinguisher \leftarrow \{0,\ldots,0\}$            // A vector of $2^n$ zeros
2  for $q \leftarrow 1$ to $Q$ do
3      $Product \leftarrow \{1,\ldots,1\}$            // A vector of $2^n$ ones
4      for $w \leftarrow 0$ to $d$ do
5          $p(X_q^{(w)}|z)$ is computed from $X_q^{(w)}$ for all $z \in \mathbb{F}_2^n$
             (e.g., using Eq. (8) and template means $Y_z^{(w)}$
             and their common covariance matrix $\Sigma^{(w)}$)
6          $FFT(p(X_q^{(w)}|.))$            // In-place FFT on $2^n$-valued
             vector $p(X_q^{(w)}|.) = \{p(X_q^{(w)}|z), z \in \mathbb{F}_2^n\}$
              // The lines 5 and 6 could be processed in parallel as
             will be shown in Sec. 3.4
7          $Product(.) \leftarrow Product(.) \bullet p(X_q^{(w)}|.)$
8      $Sum_q \leftarrow FFT^{-1}(Product(.))$            // Computation of
         convolution
9      for $k \in \mathbb{F}_2^n$ do  // Pigeonholing spectral components based
         on the plaintext $t_q \in \mathbb{F}_2^n$ value
10         $Distinguisher(k) \leftarrow$
             $Distinguisher(k) + \log Sum_q(z(t_q,k))$
             // Permutation of $Sum_q$ indices
11 return $\mathrm{argmax}_{k \in \mathbb{F}_2^n} Distinguisher(k)$
```

Algorithm 1: H-O Template Attacks with efficient processing.

For the sake of the paper to be self-contained, we

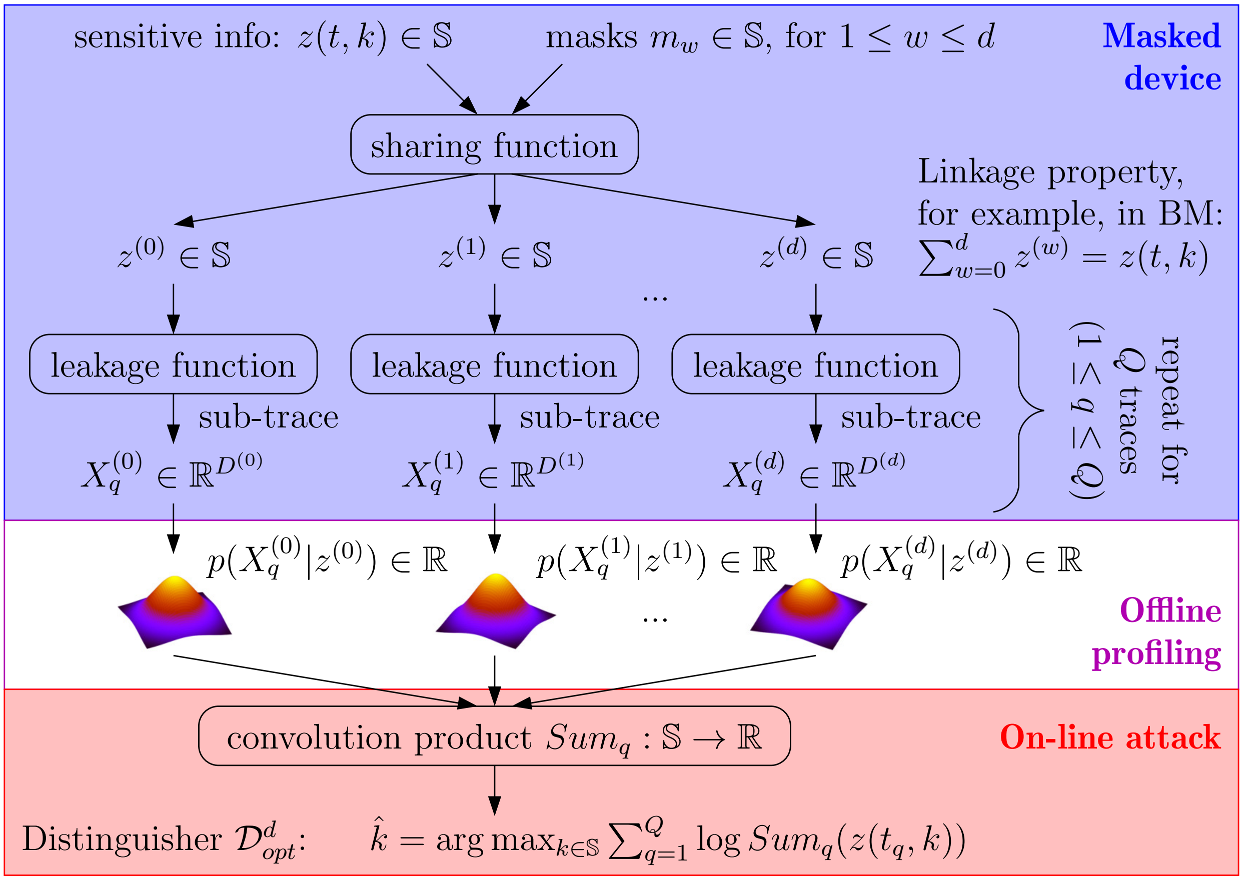

sensitive info: $z(t,k) \in \mathbb{S}$
masks $m_w \in \mathbb{S}$, for $1 \leq w \leq d$
Masked device
sharing function
Linkage property, for example, in BM: $\sum_{w=0}^{d} z^{(w)} = z(t,k)$
$z^{(0)} \in \mathbb{S}$
$z^{(1)} \in \mathbb{S}$
...
$z^{(d)} \in \mathbb{S}$
leakage function
leakage function
leakage function
sub-trace
sub-trace
sub-trace
$X_q^{(0)} \in \mathbb{R}^{D^{(0)}}$
$X_q^{(1)} \in \mathbb{R}^{D^{(1)}}$
$X_q^{(d)} \in \mathbb{R}^{D^{(d)}}$
repeat for Q traces $(1 \leq q \leq Q)$
$p(X_q^{(0)}|z^{(0)}) \in \mathbb{R}$
$p(X_q^{(1)}|z^{(1)}) \in \mathbb{R}$
$p(X_q^{(d)}|z^{(d)}) \in \mathbb{R}$
...
Offline profiling
convolution product $Sum_q : \mathbb{S} \to \mathbb{R}$
On-line attack
Distinguisher $\mathcal{D}_{opt}^{d}$: $\hat{k} = \arg\max_{k \in \mathbb{S}} \sum_{q=1}^{Q} \log Sum_q(z(t_q,k))$

Figure 1: Synoptic of the multivariate high-order attack, applied on d^{th}-order Boolean Masking (BM)

provide here-after the definition of the Walsh-Hadamard transform.

Definition 5 (Walsh-Hadamard Transform [43])**.** *Let $f : \mathbb{F}_2^n \to \mathbb{R}$. The Walsh-Hadamard transform of f is a function $\hat{f} : \mathbb{F}_2^n \to \mathbb{R}$, defined as $\hat{f}(u) = \sum_{z\in\mathbb{F}_2^n}(-1)^{u\cdot z} f(z)$, where "$\cdot$" is the canonical scalar product over $\mathbb{F}_2^n$.*

There exists a butterfly algorithm to compute WHT using only additions (and subtractions), and no multiplications.

The general expression of Eqn. (1) can be formulated in terms of Walsh-Hadamard Transform when $\mathbb{S} = \mathbb{F}_2^n$ as:

Proposition 1 (Convolution Theorem)**.**

$$\forall z, \bigotimes_{w=1}^{n} f^{(w)}(z) = \frac{1}{2^n}\widehat{\prod_{w=1}^{n}\widehat{f^{(w)}}}(z). \tag{7}$$

Proof. This result is well-known, and provided here-after only for the sake of convenience for the reader:

$$\widehat{\prod_{w=1}^{n}\widehat{f^{(w)}}}(z) = \sum_u (-1)^{u\cdot z}\widehat{f^{(1)}}(u)\cdots\widehat{f^{(n)}}(u)$$

$$= \sum_u (-1)^{u\cdot z}\sum_{y_1}(-1)^{y_1\cdot u} f^{(1)}(y_1)\cdots\sum_{y_n}(-1)^{y_n\cdot u} f^{(n)}(y_n)$$

$$= \sum_{y_1,\ldots,y_n} f^{(1)}(y_1)\cdots f^{(n)}(y_n)\sum_u (-1)^{u\cdot(z+y_1+\ldots+y_n)}$$

$$= \sum_{y_1,\ldots,y_n} f^{(1)}(y_1)\cdots f^{(n)}(y_n)\times 2^n \mathbb{1}_{y_1+\ldots+y_n=z}$$

$$= 2^n \sum_{\substack{y_1,\ldots,y_n\\ y_1+\cdots+y_n=z}} f^{(1)}(y_1)\cdots f^{(n)}(y_n) = 2^n\bigotimes_{w=1}^{n} f^{(w)}(z).$$

□

So, the result is a higher-order convolution, which can be computed efficiently (complexity of $dn2^n$ instead of 2^{nd}). Thanks to the formalization in a convolution, one can compute the Sum_q^{BM} of all the possible z at once. So the overall computation complexity of the H-O Template Attack distinguisher (Eq.(3)) is $Qdn2^n$ instead of $Q2^{nd}$. Note that our computed distinguisher is a generalisation of HOOD, in which only $(D^{(w)} = 1)_{w=0,\ldots,d}$ is considered. But our processing is $\frac{2^{nd}}{dn2^n}$ times faster.

Also, let us precise that an optimization of HOOD has been proposed in [10], in which the computationally intensive sums over all masks are simplified in precomputations (rounded at a given order). The online attack computations are therefore efficient. But the work presented in our paper shows that, thanks to Fourier transform, the rounding of attacks is not necessary (there are no longer computational reasons). Hence one major contribution is to show how to compute efficiently and exactly in high-order (possibly multivariate) contexts.

3.3 Efficient Computation of the Template Profile Functions $p(X_q^{(w)}|.)$

First, we notice that the *coalescence* principle [38] cannot be taken advantage of because $\prod_{q=1}^{Q}\sum_{m_1,\ldots,m_d\in\mathbb{S}}$ is different from $\sum_{m_1,\ldots,m_d\in\mathbb{S}}\prod_{q=1}^{Q}$. Intuitively, this means that averaging traces would result in the masking countermeasure cancelling the information.

For each possible value z of the w^{th} share, the adversary should compute $p(X_q^{(w)}|z)$. Thanks to the profiling phase, the adversary estimates this probability. For example, let us assume that the w^{th} random vector follows a Gaussian distribution PDF ($X^{(w)} \sim \mathcal{N}(Y^{(w)}, \Sigma^{(w)})$, as suggested in the seminal TA paper [18]), where $Y^{(w)}$ is the mean of the signal and $\Sigma^{(w)}$ the noise covariance matrix. The construction of the templates requires the accumulation of traces and their covariance, hence has a complexity linear with the number of training traces and quadratic in the subtraces dimensionality $D^{(w)}$, for $0 \leq w \leq d$. In this case

$$p(X_q^{(w)}|z) = \frac{1}{\sqrt{(2\pi)^{D^{(w)}}|\det\Sigma^{(w)}|}} \times \backslash$$
$$\exp\left(-\frac{1}{2}\left(X_q^{(w)} - Y_z^{(w)}\right)^{\mathsf{T}}\Sigma^{(w)^{-1}}\left(X_q^{(w)} - Y_z^{(w)}\right)\right), \tag{8}$$

where T denotes the matrix transposition operator. For the sake of clarity, we recall the dimensions:

- $X_q^{(w)}$ and $Y_z^{(w)}$ are $D^{(w)} \times 1$,
- $\Sigma^{(w)}$ is $D^{(w)} \times D^{(w)}$.

One can remove the factor independent from the key (i.e. from z), and keep only

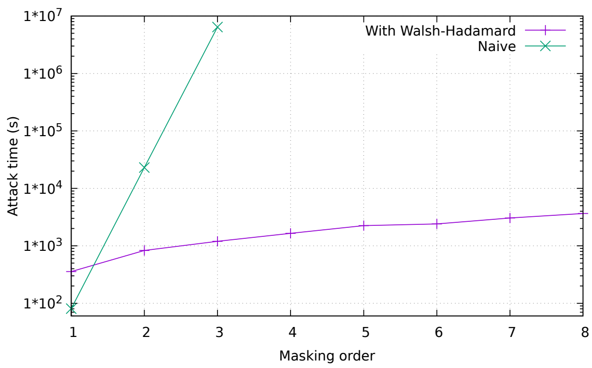


Figure 2: Single attack time (on 5000 samples), as a function of its order (= number of shares)

$$
\begin{aligned}
&p'(X_q^{(w)}|z) \\
&= \exp\left((X_q^{(w)})^\mathsf{T} {\Sigma^{(w)}}^{-1} Y_z^{(w)} - \frac{1}{2} (Y_z^{(w)})^\mathsf{T} {\Sigma^{(w)}}^{-1} Y_z^{(w)} \right) \\
&= \exp\left(((X_q^{(w)})^\mathsf{T} - \frac{1}{2} (Y_z^{(w)})^\mathsf{T}) {\Sigma^{(w)}}^{-1} Y_z^{(w)} \right) \\
&= \exp\left((X_q^{(w)} - \frac{1}{2} Y_z^{(w)})^\mathsf{T} {\Sigma^{(w)}}^{-1} Y_z^{(w)} \right). \qquad (9)
\end{aligned}
$$

It is noteworthy that the vectors ${\Sigma^{(w)}}^{-1} Y_z^{(w)}$ can be precomputed **only once** at the end of the profiling phase. The complexity of inverting $\Sigma^{(w)}$ is about $(D^{(w)})^{2.372}$. The evaluation of the PDF in Eq. (9) has complexity $D^{(w)}$ for every z and w.

3.4 Further Improvement in Performance

The attack time is almost linear as function of the masking order, as shown in Fig 2. Further performance improvement can be obtained by using multi-threaded code. According to the Alg. 1, the bottlenecks of the attack computation are the lines 5, 6 and 7. Their complexities are respectively $\mathcal{O}(2^n \times Q \sum_w D^{(w)})$ (for the multivariate Gaussian noise case), $\mathcal{O}(n2^n \times Q(d+1))$, and $\mathcal{O}(2^n \times Q(d+1))$. Since the processing of the lines 5 and 6 are independent from a share to another, one can dispatch them to $d+1$ threads, such that the wth thread processes the lines 5 and 6 for the corresponding wth share. For the 7th line, one can notice that the product can be computed out-of-order because the multiplication is commutative. This allows for a further improvement by current concurrent product computation using a shared memory between the threads.

If the computer can run $d+1$ threads in parallel, then the time to compute the attack is (almost) independent from the masking order. Clearly, if the computer has less cores than $d+1$, then the full level of simultaneous parallelism cannot be reached. Still, it is possible to devise a load balancing system which allocates a pool of workers (computing the Walsh-Hadamard transforms) to the maximum of the machine computing power capacity. Similarly, the computer memory shall be sufficient to store all structures in RAM, otherwise the operating system will swap RAM from temporary memory on the hard drive, which is efficient. The level of parallelism shall thus be carefully architected to avoid memory swapping.

3.5 Equivalent Multivariate Signal-to-Noise Ratio (SNR)

The presentation of multivariate probability distribution at each share (recall Eq. (8)) shows that the leakage can take advantage of dimensionality reduction. The transformation is that already described in the paper [7, Theorem 8]. This equivalent SNR can thus be deduced exactly with the formula given in [7, Corollary 4], namely:

$$
{Y_z^{(w)}}^\mathsf{T} \Sigma^{-1} Y_z^{(w)}, \text{ for all } w.
$$

4 Type of Fourier Transform per Masking Scheme

In this section, we detail the customizations which shall be performed to adapt the principle of fast multivariate high-order template attack (Sec. 3.2) to several masking types.

For the Boolean masking, the group law is simply the XOR operation between the $\mathbb{F}_2^n$ elements. Thereby the group is $(\mathbb{F}_2^n, \oplus)$. Subsequently, the corresponding Fourier transform is simply the Walsh-Hadamard Transform. Similarly, for a certain arithmetic masking scheme [27], the group law is the modular addition operation between the $\mathbb{F}_{2^n}$ elements. Subsequently, the corresponding Fourier transform variant is the Cyclical Fourier Transform.

To generalise our approach for other masking schemes, we study the form taken by the term $Sum_q(z)$ defined in Eq. (5), and how it shall be adapted accordingly. This expression will be specialized with the abbreviation of the masking scheme, such as for instance $Sum_q^{\mathsf{IPM}}(z)$ for the IPM masking scheme.

4.1 Type of Fourier Transform for Inner-Product Masking (IPM) Scheme

4.1.1 IPM Description

Let us first focus on the Inner Product Masking (IPM) [22, 3, 1, 19]. This choice is due to the fact that the IPM is a generalisation of several simple masking schemes (Boolean, i.e. additive, multiplicative and affine masking schemes).

Instead of simply splitting every sensitive value as the sum (i.e., $\oplus$) of random shares, IPM consists in decomposing every secret in an inner product between $d+1$ random values and a $d+1$-dimension constant public vector [2]. Formally, the designers choose a constant public vector $L = (l_0, \ldots, l_d) \in \left(\mathbb{F}_{2^n}\backslash\{0\}\right)^{d+1}$. Therefore, all field element l_w is invertible. In practice, and without loss of generality, l_0 is chosen equals 1, for performance reasons. Masking a sensitive value z by IPM means computing $d+1$ shares all in $\mathbb{F}_{2^n}$ and denoted $z^{(0)}, \ldots z^{(d)}$, such that $z = \sum_{w=0}^{d} l_w z^{(w)}$.

Let us consider as in practice $l_0 = 1$ (this condition is not restrictive). Thereby, one has $z^{(0)} = z - \sum_{w=1}^{d} l_w z^{(w)}$ So, the adversary needs to combine the $d+1$ leaking shares to lead a successful H-O attack.

4.1.2 Exhibition of the Convolution Product in Optimal Attack on IPM

According to the above description of the IPM scheme, the optimal adversary has to maximize:

$$\mathscr{D}_{opt}^{d} = \operatorname*{argmax}_{k} \prod_{q=1}^{Q} \sum_{z^{(1)},\ldots,z^{(d)} \in \mathbb{F}_{2^n}} \backslash$$

$$p(X_q^{(1)}|z^{(1)}) \ldots p(X_q^{(d)}|z^{(d)}) p(X_q^{(0)}|z - \sum_{w=1}^{d} l_w z^{(w)}). \tag{10}$$

But,

$$\begin{aligned}
Sum_q^{\mathsf{IPM}}(z) = & \sum_{(z^{(1)},\ldots,z^{(d)}) \in \Pi_{w=1}^{d} \mathbb{F}_{2^n}} p(X_q^{(1)}|z^{(1)}) \ldots \\
& p(X_q^{(d)}|z^{(d)}) p(X_q^{(0)}|z - \sum_{w=1}^{d} l_w z^{(w)}) \\
= & \sum_{(z^{(1)},\ldots,z^{(d)}) \in \Pi_{w=1}^{d} \mathbb{F}_{2^n}} p(X_q^{(1)}|l_1^{-1} l_1 z^{(1)}) \ldots \\
& p(X_q^{(d)}|l_d^{-1} l_d z^{(d)}) p(X_q^{(0)}|z - \sum_{w=1}^{d} l_w z^{(w)}) \\
= & \sum_{(z^{(1)},\ldots,z^{(d)}) \in \Pi_{w=1}^{d} l_w \mathbb{F}_{2^n}} p(X_q^{(1)}|l_1^{-1} z^{(1)}) \ldots \\
& p(X_q^{(d)}|l_d^{-1} z^{(d)}) p(X_q^{(0)}|z - \sum_{w=1}^{d} z^{(w)}) \\
= & \sum_{(z^{(1)},\ldots,z^{(d)}) \in \Pi_{w=1}^{d} l_w \mathbb{F}_{2^n}} p'(X_q^{(1)}|z^{(1)}) \ldots \\
& p'(X_q^{(d)}|z^{(d)}) p(X_q^{(0)}|z - \sum_{w=1}^{d} z^{(w)}),
\end{aligned}$$

where $p'(X_q^{(w)}|z^{(w)}) = p(X_q^{(w)}|l_w^{-1} z^{(w)})$, for all $w = 1, \ldots, d$. In fact, for any $l_w \neq 0$ one has $l_w \mathbb{F}_{2^n} = \mathbb{F}_{2^n}$ because $x \in \mathbb{F}_{2^n} \mapsto l_w x$ is a bijection.

Subsequently, the Sum_q^{IPM} becomes a convolution relatively to the addition law in group $\mathbb{F}_{2^n}$. Thereby, it can be processed efficiently thanks to the (Cyclical) Fourier Transform (CFT):

$$\begin{aligned}
Sum_q^{\mathsf{IPM}}(z) = & \left(p'_k(X_q^{(1)}|.) \otimes \cdots \otimes p'_k(X_q^{(d)}|.) \otimes p_k(X_q^{(0)}|.)\right)(z) \\
= & \; CFT^{-1}\left(CFT(p'_k(X_q^{(1)}|.)) \bullet \ldots \right. \\
& \left. \bullet CFT(p'_k(X_q^{(d)}|.)) \bullet CFT(p_k(X_q^{(0)}|.))\right)(z).
\end{aligned}$$

So the overall computation complexity of the H-O Template Attack distinguisher against the IPM is as expected above (Eq.(3)) $Qdn2^n$ instead of $Q2^{nd}$.

4.2 Type of Fourier Transform for Direct Sum Masking (DSM) Scheme

4.2.1 DSM Description

Since the IPM is a particular case of DSM [39], let us further generalise our approach for DSM. DSM is an error correction code-based masking scheme introduced in [6, 24]. Unlike IPM (which is defined over the field $\mathbb{F}_{2^n}$), DSM is defined over the spacevector $\mathbb{F}_2^n$.

To formally describe DSM, let $\mathbb{C}$ (resp. $\mathbb{D}$) be a linear code of dimension n (resp. n') and G (resp. H) be its generating matrix. The codes $\mathbb{C}$ and $\mathbb{D}$ are chosen to be complementary. This means that square matrix $\begin{pmatrix} G \\ H \end{pmatrix}$ has full rank. Subsequently, we denote J (resp. K) the $(n+n')\times n$ (resp. $(n+n')\times n'$) matrix such that:

$$\begin{pmatrix} G \\ H \end{pmatrix}^{-1} = \begin{pmatrix} J & K \end{pmatrix}. \tag{11}$$

When $\mathbb{C}$ and $\mathbb{D}$ are also dual, in addition to be complementary, then such codes have been studied by Massey [33]. In this case, $J = G^{\mathsf{T}}(GG^{\mathsf{T}})^{-1}$ and $K = H^{\mathsf{T}}(HH^{\mathsf{T}})^{-1}$. But in general, $\mathbb{C}$ and $\mathbb{D}$ are not required to be orthogonal. Using these two complementary codes, DSM consists in:

- first encoding the sensitive value $z \in \mathbb{F}_2^n$ in the n-dimensional code $\mathbb{C}$ by processing $zG \in \mathbb{F}_2^{n+n'}$,
- second encoding a random value $m \in \mathbb{F}_2^{n'}$ in the n'-dimensional code $\mathbb{D}$ by processing $z^{(1)} = mH \in \mathbb{F}_2^{n+n'}$,
- third and finally masking z by processing $z^{(0)} = zG \oplus mH$.

The advantage of this scheme from designer's point of view is the efficient method of de-masking (namely a projection), since $\mathbb{C}$ and $\mathbb{D}$ are complementary: the sensitive variable is recovered by computing $z = z^{(0)}J$.

4.2.2 Attack on DSM

Even if this masking scheme has $d_{\mathbb{D}}^{\perp} - 1$ (i.e. $d_{\mathbb{C}} - 1$ when $\mathbb{D} = \mathbb{C}^{\perp}$) as a masking order (at bit-level), the corresponding Sum_q is simply expressed as:

$$\forall z \in \mathbb{C},\ Sum_q^{\mathsf{DSM}}(z) = \sum_{m \in \mathbb{F}_{2^{n'}}} p(X_q^{(1)}|m)p(X_q^{(0)}|zG \oplus mH). \tag{12}$$

Notice that:

$$Sum_q^{\mathsf{DSM}}(z) = \sum_{z^{(1)} \in \mathbb{F}_2^{n+n'}} 1_{\mathbb{D}}(z^{(1)})p'(X_q^{(1)}|z^{(1)})p(X_q^{(0)}|zG \oplus z^{(1)}), \tag{13}$$

where $p'(X_q^{(1)}|mH) = p(X_q^{(1)}|m)$ The computation according to Eq. (13) is not efficient because its complexity is $\mathcal{O}((n+n')2^{n+n'})$. But, the straightforward processing of the Sum_q^{DSM} (according to Eq. (12)) is not so difficult. It has an overall complexity of $\mathcal{O}(2^{n+n'})$. In fact one cannot compute faster than Eq. (12) because zG and mH are not belonging to the same subspace.

4.3 Multi-share DSM (MS-DSM) Scheme

4.3.1 MS-DSM Description

Rationale for the Straightforward Extension of DSM. In this section, we aim at improving the security of DSM. One option consists in increasing the dual distance of the code $\mathbb{D}$, which comes at the expense of longer codewords (recall that in DSM, there is only one share, namely $z^{(0)}$). Indeed, at bit level:

Lemma 1 (Proposition 1 of [39])**.** *The resistance order of DSM is equal to the dual distance of code $\mathbb{D}$ minus the number one, that is $d_{\mathbb{D}}^{\perp} - 1$.*

At some point, these codewords will have a length which is incompatible with the host processor register size, or with the size of the memories they address.

Therefore, a second option shall be envisioned. It consists in:

- starting from a regular DSM protection,
- but by adding further masks within $\mathbb{D}$.

Say d masks are used. They consist in the encoding of d random values $m_1, \ldots, m_d$ in the same n'-dimension code

$\mathbb{D}$ by processing $(z^{(w)} = m_w H)_{w=1,\dots,d}$. Therefore, the splitting is as follows:

$$(z, m_1, \dots, m_d) \in \mathbb{F}_2^n \times (\mathbb{F}_2^{n'})^d \mapsto (z^{(0)}, z^{(1)}, \dots, z^{(d)}) = (zG \oplus \bigoplus_{w=1}^{d} m_w H, m_1, \dots, m_d). \quad (14)$$

The advantage of DSM in terms of ease of demasking is preserved: as all masks belong to $\mathbb{D}$, so is their sum, hence a single projection on $\mathbb{C}$ allows to remove them all at once. In practice, this means that $z = z^{(0)}J$. Furthermore, the different shares can now be processed independently, which makes mapping on the processor registers easy. Specifically, using DSM, the register bitwidth shall be enough to accommodate $n + n'$ bits, but in multi-share DSM, one can see in Eq. (14) that the masked value is broken in $d+1$ different values.

Security Analysis. This novel MS-DSM scheme processes codewords, which transforms the representation of the shares. It is thus not directly amenable to automatic analysis, as is for instance enabled by [4]. Therefore, its verification poses a threat. We report in this paper that our straightforward extension has a flaw, which is detected by carrying attacks. Namely, the number of shares is increased, and the number of traces to succeed the high-order attack increases, but then saturates at a given order (namely 3 for words represented on 4 bits). This is subsequently analyzed under the prism of code properties: the dual distance of the masking material is indeed bounded to $4 = 3+1$ when the number of shares gets > 3.

A fix is proposed in Appendix A, with an argument on the dual distance. The fix consists in using not the same H for all the masks m_w, $1 \leq w \leq d$, but in using d different codes H_w. Attacks on this revised scheme get strictly more complex (from the traces standpoint—more traces are needed to achieve a given success rate as the number of shares increases). While the attacks increasing complexity with the number of shares is not *per se* a proof for the new schemes, it nonetheless allows to validate its rationale.

Notice that, prior to our results, the study of the actual masking order was only possible using MI curves (slopes, see for instance Fig. 5 from [14]). Now, it is possible to directly conduct the attacks.

4.3.2 Exhibition of the Convolution Product in MS-DSM

From adversary point of view, our approach is useful for carrying out the Maximum Likelihood-based (optimal) distinguisher. Against such scheme, the $Sum_q^{\mathsf{MS-DSM}}$ writes:

$$\begin{aligned} & Sum_q^{\mathsf{MS-DSM}}(z) \\ = & \sum_{(m_1,\dots,m_d) \in \Pi_{w=1}^{d} \mathbb{F}_2^{n'}} p(X_q^{(1)}|m_1) \dots p(X_q^{(d)}|m_d) \\ & p(X_q^{(0)}|zG \oplus m_1 H \cdots \oplus m_d H) \\ = & \sum_{(m_1,\dots,m_d) \in \Pi_{w=1}^{d} \mathbb{F}_2^{n'}} p(X_q^{(1)}|m_1) \dots p(X_q^{(d)}|m_d) \\ & p_z(X_q^{(0)}|(0 \oplus m_1 \cdots \oplus m_d) H) \\ = & \sum_{(m_1,\dots,m_d) \in \Pi_{w=1}^{d} \mathbb{F}_2^{n'}} p(X_q^{(1)}|m_1) \dots p(X_q^{(d)}|m_d) \\ & p'_z(X_q^{(0)}|0 \oplus m_1 \cdots \oplus m_d) \\ = & \; p(X_q^{(1)}|.) \otimes \cdots \otimes p(X_q^{(d)}|.) \otimes p'_z(X_q^{(0)}|.)(0), \end{aligned}$$

where, $p'_z(X_q^{(0)}|m) = p_z(X_q^{(0)}|mH) = p(X_q^{(0)}|zG \oplus mH)$, for $m \in \mathbb{F}_2^{n'}$.

Remark 2. *For the version of Eq.* (19)*, the former convolution still applies, albeit with:* $p'(X|mH_w) = p(X|m)$ *for* $1 \leq w \leq d$*, and* $p_z(X_q^{(0)}|m_1 H_1 \oplus \dots \oplus m_d H_d) = p(X_q^{(0)}|zG \oplus m_1 H_1 \oplus \dots \oplus m_d H_d)$.

Subsequently, the sum $Sum_q^{\mathsf{MS-DSM}}$ becomes a convolution relatively to the $\oplus$ law in the group $(\mathbb{F}_2^{n'}, \oplus)$. Thereby, it can be processed efficiently for each $zG \in \mathbb{C}$, thanks to the Walsh-Hadamard Transform, as shown below:

$$\begin{aligned} & Sum_q^{\mathsf{MS-DSM}}(z) = WHT^{-1} \\ & \left(WHT(p(X_q^{(1)}|.)) \bullet \cdots \bullet WHT(p(X_q^{(d)}|.)) \bullet WHT(p'_z(X_q^{(0)}|.)) \right)(0) \\ = & \frac{1}{2^{\frac{n'}{2}}} \sum_{m \in \mathbb{F}_2^{n'}} \\ & \left(WHT(p(X_q^{(1)}|.)) \bullet \cdots \bullet WHT(p(X_q^{(d)}|.)) \bullet WHT(p'_z(X_q^{(0)}|.)) \right)(m). \end{aligned}$$

The sum expresses that the value of the Walsh-Hadamard Transform is 0 is the average of the function. One can process the product function

$$WHT(p(X_q^{(1)}|.)) \bullet \cdots \bullet WHT(p(X_q^{(d)}|.))$$

only once (for each q), since it is independent from z. Also, one can ignore the factor $\frac{1}{2^{n'/2}}$ since it is independent from k. Its overall complexity is about $Qdn2^n$.

So the overall computation complexity of the H-O Template Attack distinguisher against the so-called multi-share DSM is slightly different to that expected above (Eq. (3)). It is about $Qdn'2^{n'} + Qn'2^{n+n'}$ instead of $Qdn2^n$. It is noticeable that this complexity is still better than that of the naïve computation which is about $Q2^{n'd+n}$, especially when $d > 2$ (which is not uncommon).

4.4 Type of Fourier Transform for the Polynomial DSM (PDSM) Scheme

4.4.1 PDSM Description

A DSM scheme written over the space $\mathbb{F}_2[\alpha]/\langle P(\alpha)\rangle$, such that the degree of the irreducible polynomial P equals $n+n'$ (instead of its isomorphic space $\mathbb{F}_2^{n+n'}$), is called Polynomial Direct Sum Masking (PDSM) [16]. The PDSM is more adapted for the AES masking as both are built upon the same space $\mathbb{F}_2[\alpha]/\langle P(\alpha)\rangle$ where P is the following irreducible polynomial: $P(\alpha) = \alpha^8 + \alpha^4 + \alpha^3 + \alpha + 1$.

4.4.2 Optimal Attack on PDSM

A similar extension of the $(1+1)$-shares PDSM to the so-called $(d+1)$-shares PDSM is possible and it is of the same interest of that of DSM.

Thereby, a similar spectral approach is possible to compute the optimal distinguisher, thanks to the corresponding FFT of the additive finite group of $\mathbb{F}_2[\alpha]/\langle P(\alpha)\rangle$.

4.5 Type of Fourier Transform for the Rotating S-boxes Masking (RSM) Scheme

4.5.1 RSM Scheme Description

For a security/complexity compromise in the masking of the AES S-boxes, authors of [34] introduce the Rotating S-boxes Masking (RSM) [26, 44]. This scheme is the Boolean masking, where the masks are chosen uniformly from a code $\mathbb{C}$ [24] of length $n = 8$ in $\mathbb{F}_2$, either:

- $\mathbb{C}_0 = \{\texttt{0x00}\}$ (no masking),
- $\mathbb{C}_1 = \{\texttt{0x00}, \texttt{0xff}\}$,
- $\mathbb{C}_2$, a non-linear code of length 8, size 12 and its generating matrix denoted G, or
- $\mathbb{C}_3$, is a linear code of length 8, and dimension 4 (see paper [15] for more details).

The last case is interesting since there are sixteen masks (*i.e.* sixteen masked S-boxes' tables) [24, 11, 21].

4.5.2 Exhibition of the Convolution Product in Optimal Attack on RSM

For computing efficiently the Sum_q^{RSM} against this scheme, let us denote by G the generating matrix of $\mathbb{C}_3$. Let us also denote by $\overline{G}$ the generating matrix of a complementary subspace $\overline{\mathbb{C}}_3$ of $\mathbb{C}_3$.

Subsequently, each $z \in \mathbb{F}_2^8$ can be written as a sum $z_1 G \oplus z_2\overline{G}$, where $z_1, z_2 \in \mathbb{F}_2^4$, and the $Sum_q^{\mathsf{RSM}}(z)$ becomes:

$$\begin{aligned}
& \qquad Sum_q^{\mathsf{RSM}}(z) \\
= & \sum_{m^{(1)},\ldots,m^{(d)}\in\mathbb{C}_3} p(X_q^{(1)}|m^{(1)})\ldots p(X_q^{(d)}|m^{(d)}) \\
& \qquad p(X_q^{(0)}|z\oplus m^{(1)}\cdots\oplus m^{(d)}) \\
= & \sum_{m_1,\ldots,m_d\in\mathbb{F}_2^4} p(X_q^{(1)}|m_1G)\ldots p(X_q^{(d)}|m_dG) \\
& \qquad p(X_q^{(0)}|z_2\overline{G}\oplus(z_1\oplus m_1\cdots\oplus m_d)G) \\
= & \sum_{m_1,\ldots,m_d\in\mathbb{F}_2^4} p'(X_q^{(1)}|m_1)\ldots p'(X_q^{(d)}|m_d) \\
& \qquad p'_{z_2}(X_q^{(0)}|z_1\oplus m_1\cdots\oplus m_d) \\
= & \; p'(X_q^{(1)}|.)\otimes\cdots\otimes p'(X_q^{(d)}|.)\otimes p'_{z_2}(X_q^{(0)}|.)(z_1),
\end{aligned}$$

where,

$$\begin{cases} \left(p'(X_q^{(w)}|m_w) = p(X_q^{(w)}|m_wG)\right)_{w=1,\ldots d} \quad \text{and} \\ p'_{z_2}(X_q^{(0)}|z_1\oplus m) = p(X_q^{(0)}|z_2\overline{G}\oplus(z_1\oplus m)G). \end{cases} \tag{15}$$

Consequently, the Sum_q^{RSM} is also a convolution relatively to the $\oplus$ law in the group $(\mathbb{F}_2^4, \oplus)$. Thereby, for each $z_2 \in \mathbb{F}_2^4$ one can efficiently process the sixteen $Sum_{(q,z_2)}^{\mathsf{RSM}}(.) = Sum_q^{\mathsf{RSM}}(z_2\overline{G} \oplus .)$, thanks to the Walsh-Hadamard Transform (WHT) as follows:

$$\begin{aligned}
&Sum_q^{\mathsf{RSM}}(z) = Sum_q^{\mathsf{RSM}}(z_1 G \oplus z_2 \overline{G}) = Sum_{(q,z_2)}^{\mathsf{RSM}}(z_1) \\
&= WHT^{-1}\Big(WHT(p'(X_q^{(1)}|.)) \bullet \cdots \bullet \\
&\qquad WHT(p'(X_q^{(d)}|.)) \bullet WHT(p'_{z_2}(X_q^{(0)}|.))\Big)(z_1).
\end{aligned}$$

That means, one needs to compute only sixteen (*i.e.* $2^{n/2}$) functions $Sum_{(q,z_2)}^{\mathsf{RSM}}(.)$ with an overall complexity of $\mathcal{O}(\frac{n}{2}d2^{n/2})$ for each one. Finally, the overall complexity of the attack is about $\mathcal{O}(\frac{n}{2}Qd2^n)$, *i.e.* 2× faster then that against the perfect masking.

4.6 Type of Fourier Transform for the Leakage Squeezing Masking (LSM) Scheme

4.6.1 LSM Description

In this scheme, the shares are like for perfect masking (Boolean masking with all possible values for the masks) except some bijective functions $(F_w)_{w=1,\ldots,d}$ are applied to the masks [12, 25]. That leads to a better mixing of bits [13, 24, 20].

4.6.2 Exhibition of the Convolution Product in Optimal Attack on LSM

Since F_w is bijective for each w, so Sum_q^{LSM} becomes as follows:

$$\begin{aligned}
& Sum_q^{\mathsf{LSM}}(z) \\
= & \sum_{m^{(1)},\ldots,m^{(d)} \in G} p(X_q^{(1)}|m^{(1)}) \ldots p(X_q^{(d)}|m^{(d)}) \\
& p(X_q^{(0)}|z \oplus F_1(m^{(1)}) \cdots \oplus F_d(m^{(d)})) \\
= & \sum_{m^{(1)},\ldots,m^{(d)} \in G} p'(X_q^{(1)}|F_1(m^{(1)})) \ldots p'(X_q^{(d)}|F_d(m^{(d)})) \\
& p(X_q^{(0)}|z \oplus F_1(m^{(1)}) \cdots \oplus F_d(m^{(d)})) \\
= & \sum_{F_1(m^{(1)}),\ldots,F_d(m^{(d)}) \in G} p'(X_q^{(1)}|F_1(m^{(1)})) \ldots \\
& p'(X_q^{(d)}|F_d(m^{(d)}))p(X_q^{(0)}|z \oplus F_1(m^{(1)}) \cdots \oplus F_d(m^{(d)})) \\
= & \sum_{m^{(1)},\ldots,m^{(d)} \in G} p'(X_q^{(1)}|m^{(1)}) \ldots p'(X_q^{(d)}|m^{(d)}) \\
& p(X_q^{(0)}|z \oplus m^{(1)} \cdots \oplus m^{(d)}) \\
= & p'(X_q^{(1)}|.) \otimes \cdots \otimes p'(X_q^{(d)}|.) \otimes p(X_q^{(0)}|.)(z),
\end{aligned}$$

where $\left(p'(X_q^{(w)}|F_w(m^{(w)})) = p(X_q^{(w)}|m^{(w)})\right)_{w=1,\ldots,d}$.

Thereby,

$$\begin{aligned}
& Sum_q^{\mathsf{LSM}}(z) = WHT^{-1} \\
& \Big(WHT(p'(X_q^{(1)}|.)) \bullet \cdots \bullet WHT(p'(X_q^{(d)}|.)) \bullet WHT(p(X_q^{(0)}|.))\Big)(z).
\end{aligned}$$

4.7 Synthesis about the Six Masking Schemes

By running through all six representative masking schemes, it appears that HOOD attack against all of them can benefit from an acceleration. The adaptation of the reference attack (that on Boolean Masking, namely Eq. (6)) assumes the definition of a commutative group $\mathbb{S}$ with additive law, and a transform on the precharacterized templates (as per defined in Sec. 3.3). The table 1 summarizes different tweaks of the reference attack to adapt to other masking schemes. The notations are those borrowed from this section.

5 Experiments

5.1 Experimental Test Plan

The validation of the rewriting of the multivariate high-order optimal distinguisher is carried out on synthetic traces. As previously illustrated in Fig. 1, each trace consists in $N = d + 1$ sub-traces. Each sub-trace is multivariate; in our experiment, all are made up of the same number $D^{(w)} = D$ $(\forall w \in \{0, \ldots, d\})$ of samples. The sub-trace waveforms are represented in Fig. 3. It represents an arch of the sine curve, as can be observed in real measurements where the probe and the target circuit under

Table 1: Tweaks from the optimal attack to adapt from Boolean Masking to the other masking schemes studied in this Sec. 4

Masking scheme	**Group $\mathbb{S}$**	**Mathematical transformation on the pre--characterized templates**
IPM	$\mathbb{F}_{2^n}$	$p'(X_q^{(w)} \| z^{(w)}) = p(X_q^{(w)} \| l_w^{-1} z^{(w)})$
DSM	$\mathbb{F}_2^{n+n'}$	$p'(X_q^1 \| mH) = p(X_q^1 \| m)$
MS-DSM	$\mathbb{F}_2^{n'}$	$p'(X \| mH_w) = p(X \| m)$ and $p_z(X_q^{(0)} \| \bigoplus_{w=1}^d m_w H_w) = p(X_q^{(0)} \| zG \oplus \bigoplus_{w=1}^d m_w H_w)$
PDSM	$\mathbb{F}_2[\alpha]/\langle P(\alpha)\rangle$	Direct usage of the templates
RSM	$\mathbb{F}_2^4$	See transformation in Eq. (15)
LSM	$\mathbb{F}_2^n$	$p'(X_q^{(w)} \| F_w(m^{(w)})) = p(X_q^{(w)} \| m^{(w)})$

attack are not matched in terms of impedance. Therefore, the captured side-channel waveform is low-pass filtered, meaning that the actual "impulse" signal corresponding to the manipulation of the shares upon clock rising edge is "smoothed". On purpose, we generated traces according to the pattern in Fig. 3 for $D = 10$, and we pruned them for lower values of D. This allowed us to simulate attacks of with reduced dimensionality ($D = 9, 8, \ldots, 1$) so that they are consistently comparable, i.e., common samples have same noise. This envelop is scaled with the Hamming weight $w_H(z^{(w)})$ (living in $\{0, \ldots, n\}$) of the shared variable $z^{(w)}$.

The traces are artificially noised. Each sample in each sub-trace is added an independent identically distributed normal noise, of zero mean and of σ^2 variance. This models typical measurement noise from oscilloscopes [32]. At the point of maximal amplitude in the envelop, the signal takes value 1 (in arbitrary units). In perfect masking schemes, the sharing is fully entropic, thereby $Z^{(w)}$ is uniformly distributed in $\mathbb{S}$. In this article, the size of $\mathbb{S}$ is $|\mathbb{S}| = 2^n$. This means that the signal $w_H(Z^{(w)})$ garnered by the attacker in each sub-trace follows a binomial distribution, of parameters n and $p = 1/2$. Thus the signal variance is $Var(w_H(Z^{(w)})) = n/4$. As a result, the noise $\mathcal{N}(0, \sigma^2)$ being independent from the signal, the SNR is equal to:

$$\mathrm{SNR} = \frac{1}{4\sigma^2}.$$

The corresponding multivariate SNR is derived in Sec. 3.5.

The attack proper consists in the computation of

$$\arg\max_{k \in \mathbb{S}} \sum_{q=1}^{Q} \log Sum_q^{\mathscr{M}}(z(t_q, k)),$$

as defined in Eq. (4) and illustrated at the bottom of Fig. 3. In this equation, the calligraphic $\mathscr{M}$ represents any of the six masking schemes discussed in previous Sec. 4. We leverage the fact that the expression can be evaluated incrementally to compute the attack success SR versus the number of side-channel traces q. Namely, the algorithm consists in the repetition of $R = 100$ attacks, and the estimation of the success rate by averaging the number of correct key k^* retrieval as a function of q. It is given in Alg. 2.

```
Input : The value of the correct key $k^* \in \mathbb{S}$, and the
        side-channel information, namely:
    • Sub-traces $(X[r]_q^{(w)})_{1\leq q\leq Q, 0\leq w\leq d, 1\leq r\leq R}$ and
    • corresponding plaintexts $(t[r]_q)_{1\leq q\leq Q, 1\leq r\leq R}$.
Output : Success rate curve $(\mathsf{SR}(q))_{1\leq q\leq Q}$.
$\mathsf{SR} \leftarrow \{0,\ldots,0\}$                    // A vector of $Q$ zeros
for $r \leftarrow 1$ to 100 do
    $\mathscr{D}^{\mathscr{M}} \leftarrow \{0,\ldots,0\}$              // A vector of $2^n$ zeros
    for $q \leftarrow 1$ to $Q$ do
        for $k \leftarrow \mathbb{S}$ do
            $\mathscr{D}^{\mathscr{M}}(k) \leftarrow \mathscr{D}^{\mathscr{M}}(k) + \log Sum_q^{\mathscr{M}}(z(t[r]_q, k))$
        if $k^* = \arg\max_{k\in\mathbb{S}} \mathscr{D}^{\mathscr{M}}(k)$ then
            $\mathsf{SR}(q) \leftarrow \mathsf{SR}(q) + 1$         // Incrementation
return $\mathsf{SR}/100$          // Vectorial scaling by factor $1/100$
```

Algorithm 2: Estimation of the attack success rate SR on masking scheme $\mathscr{M}$ versus the number of consumed traces q.

In practice, for the sake of implementation simplicity, Alg. 2 has been coded in `python` scripting language. Still, this implementation leverages efficient computational packages, such as `numpy`. The Walsh-Hadamard transform is computed using the open source library `fwht` [31]. Notice that the absolute time for the attacks shall not be considered; only the relative time between two setups can be appreciated in a "portable" manner across different implementations of Alg. 2.

5.2 Results of High-Order Attacks on Boolean Masking

The success rate of attacks on windows of size $D = 1,2,\ldots,10$ are shown in Fig. 8, for several masking orders ($N = 1,\ldots,8$) in the case of Boolean masking. The duration of the success rate computation (excluding the time to generate the traces) is provided in the figures as well; they result from the execution of 100 independent attacks (recall Alg. 2), as executed on a server[1] running GNU/Linux Debian 4.9.189-3+deb9u1. The durations are the sum of the times for attacks on traces with dimensionality $D = 1,2,\ldots,10$.

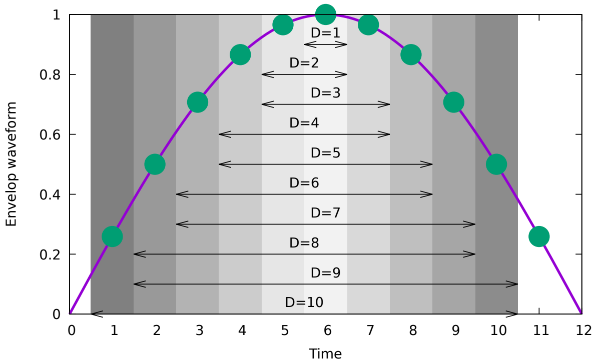


Figure 3: Envelop of the $D = 1,2,\ldots,10$ considered waveforms

A summary of all those graphs is given in Fig. 4. Curves interrupt (for increasing values of N) when there is not enough simulation data; however, one can safely extrapolate them. This beam of curves reveals several aspects of masking schemes security:

- whatever the masking scheme and whatever the leakage dimensionality, the number of traces to recover the key with a given probability is exponential in the masking order;
- the dimensionality plays the role of an "SNR" booster, in that if it increases, then the number of traces required to extract is reduced (though remaining exponential). This is consistent with the fact that more information is available when the dimensionality D increases, despite the added noise on the extra samples.

Each of the graphs in Fig. 8 clearly demonstrate the usefulness to shift from monovariate HOOD ($D = 1$) to multivariate HOOD ($D > 1$). Indeed, the multivariate traces allow for attacks with fewer traces. Please notice that for $D = 1$, our new attack is the same as the HOOD, except that it is computed 32^d faster.

The computation gain of our Walsh-Hadamard approach is contrasted to that of the naïve approach in Fig. 2. The figures 8 and 2 demonstrate the usefulness of the

[1] 32-core Intel(R) Xeon(R) CPU E7- 8837 @ 2.67GHz, with 256 GB of RAM.

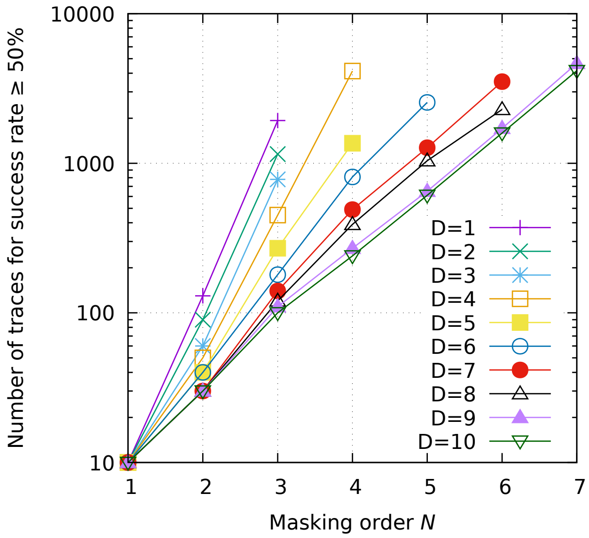


Figure 4: Number of traces necessary for SR $\geq 50\%$ on $n = 8$ bits (when $\sigma = 1$), as a function of the masking order $N = d + 1$, for several sub-trace dimensionalities D

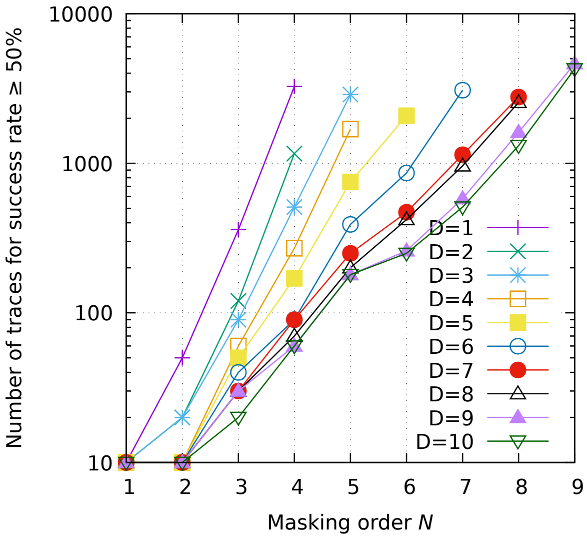


Figure 5: Number of traces necessary for SR $\geq 50\%$ on $n = 4$ bits (when $\sigma = 1$), as a function of the masking order $N = d + 1$, for several sub-trace dimensionalities D

Walsh-Hadamard transform, since the complexity is reduced from 2^{nd} to $nd2^n$. This is a decrease from exponential to linear with respect to side-channel order d.

We next analyze the case where z is on $n = 4$ bits (using PRESENT S-box). The number of traces to recover the key with probability 50% is given in Fig. 5. The success rate of attacks is represented in Fig. 9. One can see that, compared to the case of AES ($n = 8$ bits), the attacks require a fewer number of traces.

5.3 Results of Multi-Shares DSM applied to PRESENT

In this section, we evaluate the security of introduced "Multi-Share DSM" countermeasure applied on PRESENT ($n = 4$ bit). For the corresponding basic DSM, the selected codes $\mathbb{C}$ and $\mathbb{D}$ are generated by matrices G and H given below:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

and

$$H = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Results on multi-share DSM are shown in Fig. 10 for $\sigma = 1$. This figure shows that the incorrect construction (see Appendix A.1) stops making attacks more complex, in terms of number of traces to recover the key, when order increases (namely, there is no further improvement starting from order $d = 2$). Indeed, it is proven in the appendix that the security order is no more than $d^{\perp}_{\mathbb{D}} - 1 = 3 - 1$ (that is when $N = d + 1 = 3$). The behavior is reminiscent of the one already noted by Battistello et al. [5]: the attack countermeasure becomes no longer useful at a given noise level σ when the order increases. Such behaviour could not have been seen by using the regular attack method, since the order is too high.

After fixing the multi-share DSM (the matrices H_w are all different, for $1 \leq w \leq d$), we get new results which attest that the scheme is now more secure (no flaw), as shown in Fig. 11. The number of traces to extract the

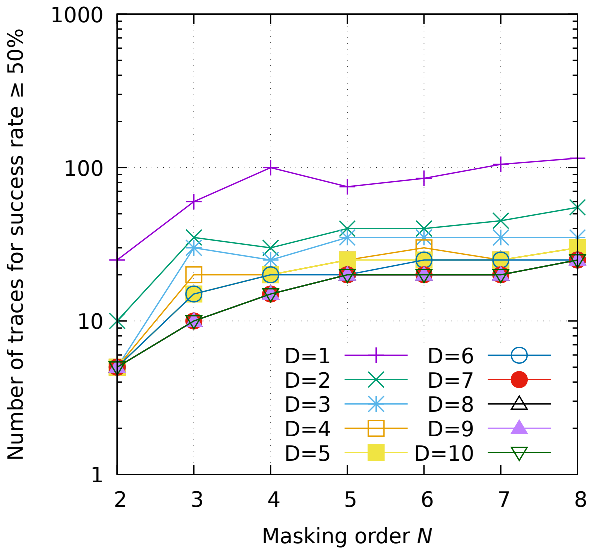


Figure 6: Number of traces necessary for SR $\geq 50\%$ for incorrect MS-DSM on $n = 4$ bits (when $\sigma = 1$), as a function of the masking order $N = d+1$, for several sub-trace dimensionalities D

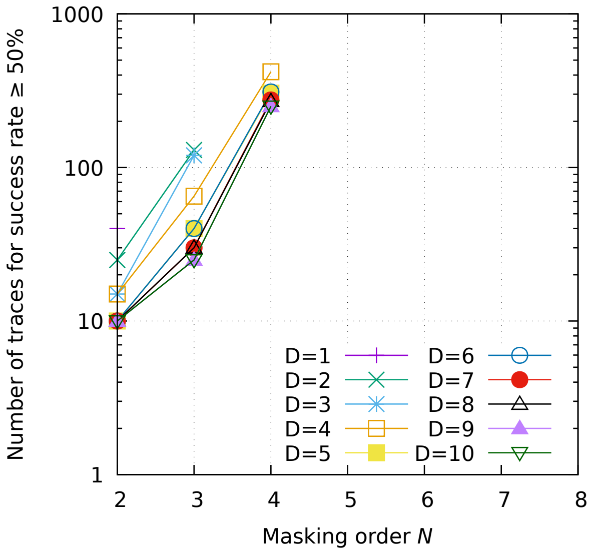


Figure 7: Number of traces necessary for SR $\geq 50\%$ for correct MS-DSM on $n = 4$ bits (when $\sigma = 1$), as a function of the masking order $N = d+1$, for several sub-trace dimensionalities D

key with success probability $\geq 50\%$ is provided in Fig. 6 (resp. Fig. 7) for the incorrect (resp. correct) versions. In the incorrect version, the number of traces required to break the secret key stops increasing after order $N \approx 4$, whereas it is exponential (as in Fig. 4 and 5) for the corrected version.

5.4 Discussion

The simulations described in this section attest that the optimized rewriting of the high-order multivariate template attack can be used as a "drop-in" replacement for the naïve HOOD. This explains why our experiments have been carried out only on synthetic traces: this paper is concerned with the *analysis part*, not the *acquisition part* of side-channel attacks. Now, the high-order multivariate template attacks described in this paper apply on provided these conditions are met:

- the shares are manipulated at non-overlapping moments in time;
- each share is leaked independently,
- the noise is additive and independent from the signal.

Those conditions could be seen as limitations, but in practice, they are always asserted. Indeed, in order to avoid combination of shares by the processor to appear out of the blue, the manipulation of each share is carefully separated from that of other shares.

Our efficient computation method allowed us to rapidly test several situations, which uncovered many interesting results. For example, we characterized that:

- The number of traces required to extract the secret key with a given success rate increases exponentially with the attack order;
- This number of traces decreases with the sub-traces dimensionality;
- High-order masking schemes can be validated by simply carrying attacks on them. Flaws can thus be identified.

6 Conclusion and Perspectives

6.1 Conclusion

Template attacks can be performed in the presence of masking. In this respect, the leakage from the different shares shall be profiled. The result is a collection of $d+1$ distributions, characterized for each share value. For each share value, the profiling is a waveform (termed a sub-trace), and the attack naturally apply to this situation, even though the sub-traces have different sizes. We show in this paper that the attack therefore consists in the convolution product of the $d+1$ distributions, independently to the masking scheme. The exact nature of the convolution depends on the type of masking (i.e., the random sharing method), and we show how to compute this convolution for six representative masking schemes.

The template attack online computation can be greatly speeded-up by computing the convolution using spectral techniques (Plancherel equality). Furthermore, to still speed-up the attack, we provide an interesting multi-threading implementation of our approach. In this respect, we show very high order attacks on six flavors of masking schemes. In particular, we introduce a new masking scheme (multi-share extension of DSM), and show based on high-order attacks a 3rd order flaw. We then fix the scheme and validate it by attacks: the number of traces to succeed the attacks do increase as the number of shares increases.

6.2 Perspectives

The properties of the masking schemes highlighted in this paper could benefit to other studies related to side-channel analysis. For instance, a technique to formally verify masking schemes is presented in [4]. It consists in a symbolic verification that all tuples of shares which are supposed to be independent of the sensitive variable indeed are. The methodology bases itself upon simplification rules which are specific to Boolean masking, in that it replaces expressions such as $X_1 \oplus E$ where E is an expression which depends on random variables $(X_i)_{i\in I}$, where $1 \notin I$, as R (which is randomly distributed). The method of Barthe [4] can be transported to other masking schemes by leveraging the transformations we put forward in this paper.

Acknowledgements

The authors acknowledge technical and scientific inputs from Nicolas Bruneau and Olivier Rioul.

Edge computing is challenging to protect against side-channel attacks, since it can be the target of both network-level (such as Platypus [29]) and physical-level side-channel attacks (such as those exploiting power or electromagnetic leaks). This research is therefore motivated to better evaluate the robustness of high-order masking schemes. It has been partly financed by the BRAINE ("Big data pRocessing and Artificial Intelligence at the Network Edge") H2020 ECSEL European Project, under Grant Agreement N° 876967.

References

[1] Josep Balasch, Sebastian Faust, and Benedikt Gierlichs. Inner product masking revisited. In Oswald and Fischlin [36], pages 486–510.

[2] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating Inner Product Masking. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 724–754. Springer, 2017.

[3] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, and Ingrid Verbauwhede. Theory and Practice of a Leakage Resilient Masking Scheme. 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 758–775. Springer, 2012.

[4] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, and Pierre-Yves Strub. Verified Proofs of Higher-Order

Masking. In Oswald and Fischlin [36], pages 457–485.

[5] Alberto Battistello, Jean-Sébastien Coron, Emmanuel Prouff, and Rina Zeitoun. Horizontal Side-Channel Attacks and Countermeasures on the ISW Masking Scheme. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, volume 9813 of *Lecture Notes in Computer Science*, pages 23–39. Springer, 2016.

[6] Julien Bringer, Claude Carlet, Hervé Chabanne, Sylvain Guilley, and Houssem Maghrebi. Orthogonal Direct Sum Masking – A Smartcard Friendly Computation Paradigm in a Code, with Builtin Protection against Side-Channel and Fault Attacks. In *WISTP*, volume 8501 of *LNCS*, pages 40–56. Springer, June 2014. Heraklion, Greece.

[7] Nicolas Bruneau, Sylvain Guilley, Annelie Heuser, Damien Marion, and Olivier Rioul. Less is More - Dimensionality Reduction from a Theoretical Perspective. In Tim Güneysu and Helena Handschuh, editors, *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, volume 9293 of *Lecture Notes in Computer Science*, pages 22–41. Springer, 2015.

[8] Nicolas Bruneau, Sylvain Guilley, Annelie Heuser, Damien Marion, and Olivier Rioul. Optimal side-channel attacks for multivariate leakages and multiple models. *J. Cryptographic Engineering*, 7(4):331–341, 2017.

[9] Nicolas Bruneau, Sylvain Guilley, Annelie Heuser, and Olivier Rioul. Masks Will Fall Off – Higher-Order Optimal Distinguishers. 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 344–365. Springer, 2014.

[10] Nicolas Bruneau, Sylvain Guilley, Annelie Heuser, Olivier Rioul, François-Xavier Standaert, and Yannick Teglia. Taylor Expansion of Maximum Likelihood Attacks for Masked and Shuffled Implementations. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 573–601, 2016.

[11] Claude Carlet. Correlation-Immune Boolean Functions for Leakage Squeezing and Rotating S-Box Masking against Side Channel Attacks. In Benedikt Gierlichs, Sylvain Guilley, and Debdeep Mukhopadhyay, editors, *SPACE*, volume 8204 of *Lecture Notes in Computer Science*, pages 70–74. Springer, 2013.

[12] Claude Carlet, Jean-Luc Danger, Sylvain Guilley, and Houssem Maghrebi. Leakage Squeezing of Order Two. In Steven D. Galbraith and Mridul Nandi, editors, *Progress in Cryptology - INDOCRYPT 2012, 13th International Conference on Cryptology in India, Kolkata, India, December 9-12, 2012. Proceedings*, volume 7668 of *Lecture Notes in Computer Science*, pages 120–139. Springer, 2012.

[13] Claude Carlet, Jean-Luc Danger, Sylvain Guilley, and Houssem Maghrebi. Leakage squeezing: Optimal implementation and security evaluation. *J. Mathematical Cryptology*, 8(3):249–295, 2014.

[14] Claude Carlet, Jean-Luc Danger, Sylvain Guilley, Houssem Maghrebi, and Emmanuel Prouff. Achieving side-channel high-order correlation immunity with leakage squeezing. *J. Cryptographic Engineering*, 4(2):107–121, 2014.

[15] Claude Carlet and Sylvain Guilley. Side-Channel Indistinguishability. In *HASP*, pages 9:1–9:8, New York, NY, USA, June 23-24 2013. ACM.

[16] Tavernier Cedric, Claude Carlet, Sylvain Guilley, and Abderrahman Daif. Polynomial direct sum masking to protect against both SCA and FIA. *Journal of Cryptographic Engineering*, 08 2018.

[17] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 1999.

[18] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski, Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.

[19] Wei Cheng, Sylvain Guilley, Claude Carlet, Sihem Mesnager, and Jean-Luc Danger. Optimizing Inner Product Masking Scheme by a Coding Theory Approach. *IEEE Trans. Inf. Forensics Secur.*, 16:220–235, 2021.

[20] Jean-Luc Danger and Sylvain Guilley. Protection des modules de cryptographie contre les attaques en observation d'ordre élevé sur les implémentations à base de masquage, 20 Janvier 2009. Brevet Français FR09/50341, assigné à l'Institut TELECOM.

[21] Alexander DeTrano, Naghmeh Karimi, Ramesh Karri, Xiaofei Guo, Claude Carlet, and Sylvain Guilley. Exploiting Small Leakages in Masks to Turn a Second-Order Attack into a First-Order Attack and Improved Rotating Substitution Box Masking with Linear Code Cosets. *The Scientific World Journal*, page 10, 2015. DOI: 10.1155/2015/743618.

[22] Stefan Dziembowski and Sebastian Faust. Leakage-resilient cryptography from the inner-product extractor. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 702–721. Springer, 2011.

[23] Louis Goubin. A Sound Method for Switching between Boolean and Arithmetic Masking. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *CHES*, volume 2162 of *Lecture Notes in Computer Science*, pages 3–15. Springer, 2001.

[24] Sylvain Guilley, Annelie Heuser, and Olivier Rioul. Codes for Side-Channel Attacks and Protections. In Said El Hajji, Abderrahmane Nitaj, and El Mamoun Souidi, editors, *Codes, Cryptology and Information Security - Second International Conference, C2SI 2017, Rabat, Morocco, April 10-12, 2017, Proceedings - In Honor of Claude Carlet*, volume 10194 of *Lecture Notes in Computer Science*, pages 35–55. Springer, 2017.

[25] Sandip Karmakar and Dipanwita Roy Chowdhury. Leakage Squeezing Using Cellular Automata. In Jarkko Kari, Martin Kutrib, and Andreas Malcher, editors, *Automata*, volume 8155 of *Lecture Notes in Computer Science*, pages 98–109. Springer, 2013.

[26] Sebastian Kutzner and Axel Poschmann. On the Security of RSM - Presenting 5 First- and Second-Order Attacks. In Emmanuel Prouff, editor, *Constructive Side-Channel Analysis and Secure Design - 5th International Workshop, COSADE 2014, Paris, France, April 13-15, 2014. Revised Selected Papers*, volume 8622 of *Lecture Notes in Computer Science*, pages 299–312. Springer, 2014.

[27] Kerstin Lemke, Kai Schramm, and Christof Paar. DPA on n-Bit Sized Boolean and Arithmetic Operations and Its Application to IDEA, RC6, and the HMAC-Construction. In *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 205–219. Springer, August 11-13 2004. Cambridge, MA, USA.

[28] Kerstin Lemke-Rust and Christof Paar. Gaussian Mixture Models for Higher-Order Side Channel Analysis. In Pascal Paillier and Ingrid Verbauwhede, editors, *CHES*, volume 4727 of *LNCS*, pages 14–27. Springer, 2007.

[29] Moritz Lipp, Andreas Kogler, David Oswald, Michael Schwarz, Catherine Easdon, Claudio Canella, and Daniel Gruss. With Great Power

Comes Great Leakage: Software-based Power Side-Channel Attacks on x86. 2020.

[30] Victor Lomné, Emmanuel Prouff, Matthieu Rivain, Thomas Roche, and Adrian Thillard. How to Estimate the Success Rate of Higher-Order Side-Channel Attacks. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, volume 8731 of *Lecture Notes in Computer Science*, pages 35–54. Springer, 2014.

[31] Ding Luo. `fwht`: Fast Walsh Hadamard Transform in Python, June 11 2015. https://github.com/dingluo/fwht.

[32] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, December 2006. ISBN 0-387-30857-1, http://www.dpabook.org/.

[33] James L. Massey. Linear codes with complementary duals. *Discrete Mathematics*, 106-107:337–342, 1992.

[34] Maxime Nassar, Youssef Souissi, Sylvain Guilley, and Jean-Luc Danger. RSM: A small and fast countermeasure for AES, secure against 1st and 2nd-order zero-offset SCAs. In Wolfgang Rosenstiel and Lothar Thiele, editors, *2012 Design, Automation & Test in Europe Conference & Exhibition, DATE 2012, Dresden, Germany, March 12-16, 2012*, pages 1173–1178. IEEE, 2012.

[35] Xuan Thuy Ngo, Shivam Bhasin, Jean-Luc Danger, Sylvain Guilley, and Zakaria Najm. Linear complementary dual code improvement to strengthen encoded circuit against hardware Trojan horses. In *IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2015, Washington, DC, USA, 5-7 May, 2015*, pages 82–87. IEEE, 2015.

[36] Elisabeth Oswald and Marc Fischlin, editors. *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*. Springer, 2015.

[37] Elisabeth Oswald and Stefan Mangard. Template Attacks on Masking — Resistance Is Futile. In Masayuki Abe, editor, *CT-RSA*, volume 4377 of *Lecture Notes in Computer Science*, pages 243–256. Springer, 2007.

[38] Maamar Ouladj, Nadia El Mrabet, Sylvain Guilley, Philippe Guillot, and Gilles Millérioux. On the power of template attacks in highly multivariate context. *J. Cryptogr. Eng.*, 10(4):337–354, 2020.

[39] Romain Poussier, Qian Guo, François-Xavier Standaert, Claude Carlet, and Sylvain Guilley. Connecting and Improving Direct Sum Masking and Inner Product Masking. In Thomas Eisenbarth and Yannick Teglia, editors, *Smart Card Research and Advanced Applications - 16th International Conference, CARDIS 2017, Lugano, Switzerland, November 13-15, 2017, Revised Selected Papers*, volume 10728 of *Lecture Notes in Computer Science*, pages 123–141. Springer, 2017.

[40] Emmanuel Prouff and Matthieu Rivain. Masking against Side-Channel Attacks: A Formal Security Proof. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 142–159. Springer, 2013.

[41] Emmanuel Prouff, Matthieu Rivain, and Régis Bevan. Statistical Analysis of Second Order Differential Power Analysis. *IEEE Trans. Computers*, 58(6):799–811, 2009.

[42] 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 *ASIACRYPT*, volume 6477 of *LNCS*, pages 112–129. Springer, December 5-9

2010. Singapore. http://www.dice.ucl.ac.be/~fstandae/PUBLIS/88.pdf.

[43] Audrey Terras. *Fourier Analysis on Finite Groups and Applications*. London Mathematical Society Student Texts. Cambridge University Press, 1999.

[44] Noritaka Yamashita, Kazuhiko Minematsu, Toshihiko Okamura, and Yukiyasu Tsunoo. A smaller and faster variant of RSM. In *DATE*, pages 1–6. IEEE, 2014.

A Multi-Share DSM Scheme

In this appendix, we explain how to extend DSM to a multi-share case. This appendix is divided into two subsections: how not to extend (Sec. A.1) and how to it (Sec. A.2). Both methods are supported by arguments based on coding theory.

A.1 Incorrect Multi-Share DSM Scheme

A.1.1 Recall About DSM

The DSM leverages two complementary codes $\mathbb{C}$ and $\mathbb{D}$ of generator matrix G and H. The unique share is denoted by $z^{(0)} = xG + mH$. As per Lemma 1, we are interested in the dual distance $d^{\perp}_{\mathbb{D}}$, because the bit-level security order is equal to $d^{\perp}_{\mathbb{D}} - 1$.

A.1.2 Extension of DSM: a First Attempt

The first generalization is that given in Eq. (14). Actually, although the sharing is now written with $(d+1)$ shares, the representation can still be viewed as DSM, using:

$$\begin{cases} \mathbf{G} &= \begin{pmatrix} G & 0_{n\times(n+n')} & 0_{n\times(n+n')} & \cdots & 0_{n\times(n+n')} \end{pmatrix} \\ \mathbf{H} &= \begin{pmatrix} H & I_{n'} & 0_{n'\times n'} & \cdots & 0_{n\times n'} \\ H & 0_{n'\times n'} & I_{n'} & \cdots & 0_{n\times n'} \\ \vdots & 0_{n'\times n'} & 0_{n'\times n'} & \ddots & \\ H & 0_{n'\times n'} & 0_{n\times n'} & \cdots & I_{n'} \end{pmatrix} \end{cases} \tag{16}$$

where $\mathbf{G}$ is used in lieu of G and $\mathbf{H}$ is used in lieu of H. It is easy to see that the dual of spacevector generated by $\mathbf{H}$ is generated by:

$$\begin{pmatrix} I_{n+n'} & H^{\mathsf{T}} & \cdots & H^{\mathsf{T}} \end{pmatrix}. \tag{17}$$

This means that all rows of Eq. (17) are orthogonal with all rows of $\mathbf{H}$ as defined in Eq. (16). The security of this new masking scheme is thus equal to $d^{\perp}_{\mathrm{span}(\mathbf{H})} - 1$. It is equal to minimum distance of the code spawn by the matrix represented in Eq. (17) (minus the number one).

It could be believed that the minimum distance of the matrix represented in Eq. (17) grows when the number of H^{T} blocks increases. This is not true, since the minimum distance can saturate. This happens when there exists a linear combination of lines in H^{T} that is equal to the null vector. Typically, we used

$$H = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Clearly, the sum of the four first column is equal to the all-zero column, hence after transposition, the sum of the four first lines of H^{T} is equal to zero.

So, at most, the minimum dual distance of the code generated by $\mathbf{H}$ is 5, and the maximum security level is 4. This explains why the attacks do not become more difficult as d increases (in Fig. 10).

A.2 Correct Multi-Share DSM Scheme

A.2.1 Proposed Correction, Inspired from IPM

The limitation identified in the first attempt to make DSM multi-share came from the fact all masks $m_1, \ldots, m_d$ where processed by the same code $\mathbb{D}$. One wishes to replace Eq. (17) by:

$$\begin{pmatrix} I_{n+n'} & H_1^{\mathsf{T}} & \cdots & H_d^{\mathsf{T}} \end{pmatrix}. \tag{18}$$

It is then possible have a growth of the minimum distance of the spacevector spawn of this matrix linearly with d. Thus, instead of Eq. (16), the correct multi-share DSM scheme therefore employs:

$$\begin{cases} \mathbf{G} &= \begin{pmatrix} G & 0_{n\times(n+n')} & 0_{n\times(n+n')} & \cdots & 0_{n\times(n+n')} \end{pmatrix} \\ \mathbf{H} &= \begin{pmatrix} H_1 & I_{n'} & 0_{n'\times n'} & \cdots & 0_{n\times n'} \\ H_2 & 0_{n'\times n'} & I_{n'} & \cdots & 0_{n\times n'} \\ \vdots & 0_{n'\times n'} & 0_{n'\times n'} & \ddots & \\ H_d & 0_{n'\times n'} & 0_{n\times n'} & \cdots & I_{n'} \end{pmatrix} \end{cases} \tag{19}$$

where $\mathbf{G}$ is used in lieu of G and $\mathbf{H}$ is used in lieu of H.

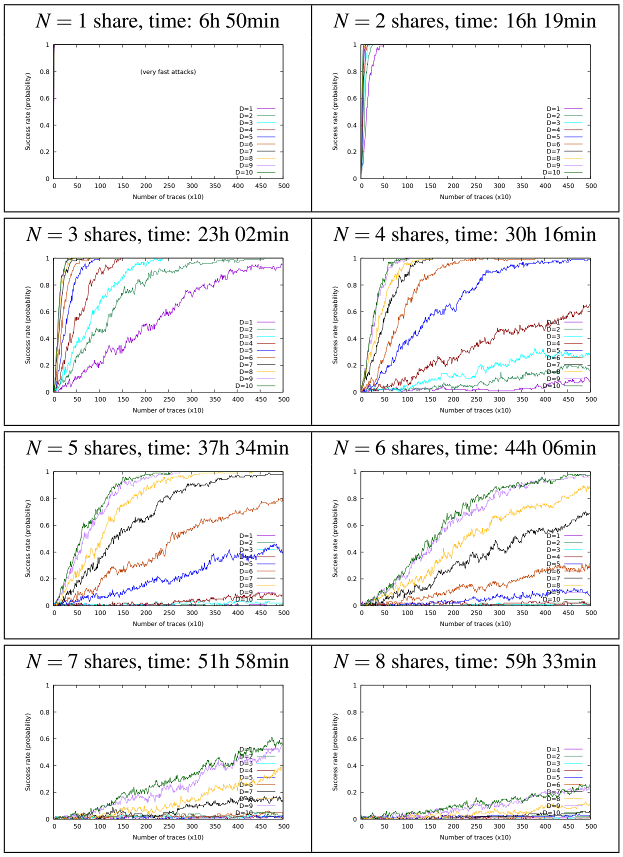
A.2.2 Validation

This new scheme works in practice, as attested by the attacks requiring more and more traces to reach a given success rate, (as show in Fig. 11).

B Success Rate of Attacks

This appendix provides the success rate curves of the multivariate high-order template attacks on various masking schemes.

Namely, Fig. 4 (resp. Fig. 5, Fig. 6, Fig. 7) have been plotted by extracting the minimum number of traces required to reach success rate $\geq 50\%$ from Fig. 8 (resp. Fig. 9, Fig. 10, Fig. 11).



$N = 1$ share, time: 6h 50min

$N = 2$ shares, time: 16h 19min

$N = 3$ shares, time: 23h 02min

$N = 4$ shares, time: 30h 16min

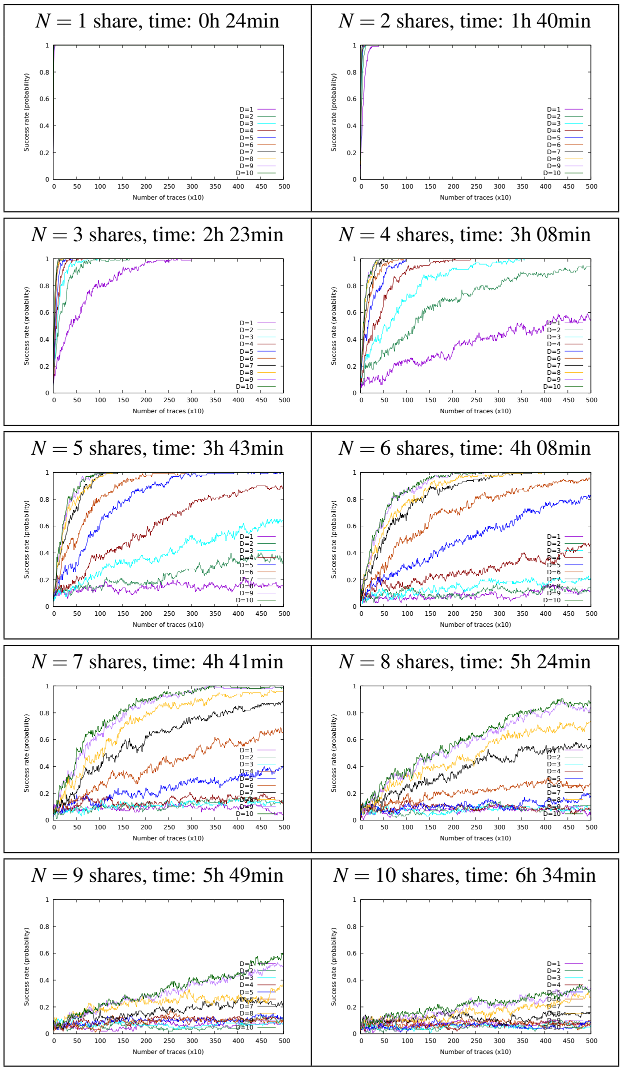
$N = 5$ shares, time: 37h 34min

$N = 6$ shares, time: 44h 06min

$N = 7$ shares, time: 51h 58min

$N = 8$ shares, time: 59h 33min

Figure 8: Attacks on $D = 1, \ldots, 10$ samples on Boolean Masking of AES ($n = 8$ bit) for noise $\sigma = 1.0$, and various number of shares (for 100 attacks)



$N = 1$ share, time: 0h 24min

$N = 2$ shares, time: 1h 40min

$N = 3$ shares, time: 2h 23min

$N = 4$ shares, time: 3h 08min

$N = 5$ shares, time: 3h 43min

$N = 6$ shares, time: 4h 08min

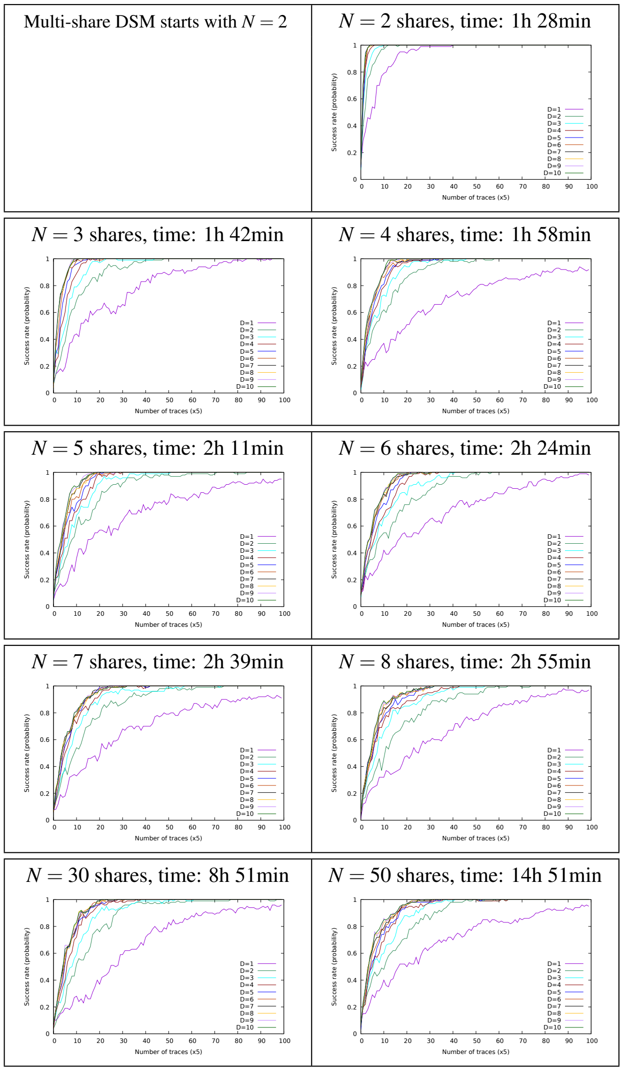
$N = 7$ shares, time: 4h 41min

$N = 8$ shares, time: 5h 24min

$N = 9$ shares, time: 5h 49min

$N = 10$ shares, time: 6h 34min

Figure 9: Attacks on $D = 1, \ldots, 10$ samples on Boolean Masking of PRESENT lightweight block cipher ($n = 4$ bit) for noise $\sigma = 1.0$, and various number of shares (for 100 attacks)



Multi-share DSM starts with $N = 2$

$N = 2$ shares, time: 1h 28min

$N = 3$ shares, time: 1h 42min

$N = 4$ shares, time: 1h 58min

$N = 5$ shares, time: 2h 11min

$N = 6$ shares, time: 2h 24min

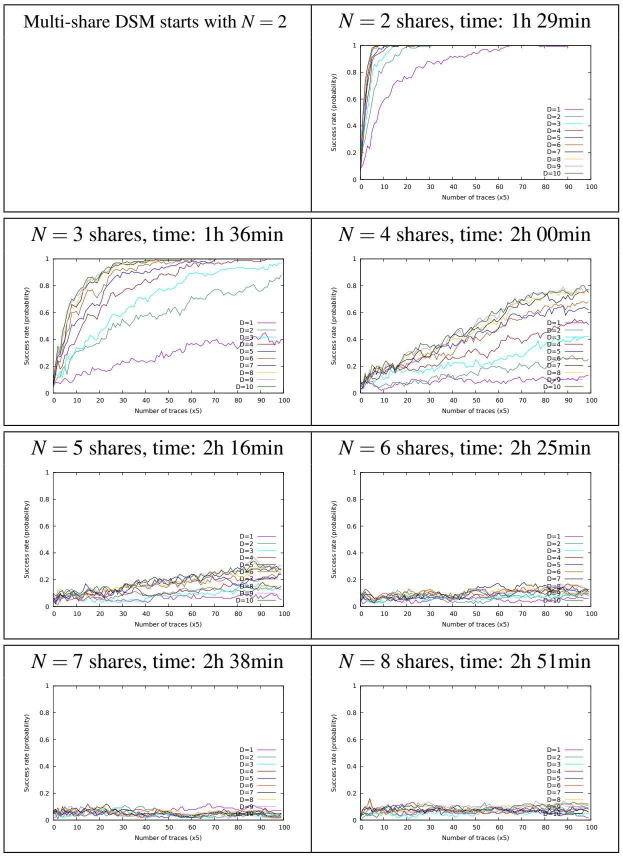
$N = 7$ shares, time: 2h 39min

$N = 8$ shares, time: 2h 55min

$N = 30$ shares, time: 8h 51min

$N = 50$ shares, time: 14h 51min

Figure 10: Attacks on $D = 1, \ldots, 10$ samples for noise $\sigma = 1.0$, and various number of shares (for 100 attacks) against (incorrect, see App. A.1) Multi-Share DSM on PRESENT ($n = 4$ bit)



Multi-share DSM starts with $N = 2$

$N = 2$ shares, time: 1h 29min

$N = 3$ shares, time: 1h 36min

$N = 4$ shares, time: 2h 00min

$N = 5$ shares, time: 2h 16min

$N = 6$ shares, time: 2h 25min

$N = 7$ shares, time: 2h 38min

$N = 8$ shares, time: 2h 51min

Figure 11: Attacks on $D = 1,\ldots,10$ samples for noise $\sigma = 1.0$, and various number of shares (for 100 attacks) against (correct, see App. A.2) Multi-shares DSM on PRESENT ($n = 4$ bit)