Resilient Uniformity: Applying Resiliency in Masking

Siemen Dhooghe and Svetla Nikova

imec-COSIC, KU Leuven, Belgium
siemen.dhooghe@esat.kuleuven.be, svetla.nikova@esat.kuleuven.be

Abstract Threshold Implementations are known countermeasures defending against side-channel attacks via the use of masking techniques. While sufficient properties are known to defend against first-order side-channel attacks, it is not known how to achieve higher-order security. This work generalizes the Threshold Implementation notion of uniformity and proves it achieves second-order protection. The notion is applied to create a second-order masking of the PRESENT cipher with a low randomness cost.

1 Introduction

Side-Channel Analysis (SCA) is considered to be a powerful method which can be used to extract secrets, e.g. keys or passwords, from cryptographic implementations when running on embedded devices. Side-channel analysis exploits the information leaked during the computation of a cryptographic algorithm. The most common technique is to analyze the power consumption of a cryptographic device using Differential Power Analysis (DPA). This side-channel attack exploits the correlation between the instantaneous power consumption of a device and the intermediate results of a cryptographic algorithm. A popular method to protect implementations against SCA is masking [6, 10]. Masking works by splitting every intermediate variable that depends on the secret into several shares, such that knowledge of any share does not reveal any information about the intermediate variable. This splitting breaks the dependency between the average instantaneous power consumption and the sensitive intermediates, and thus thwarts first-order DPA attacks. In theory, however, a masked implementation can always be broken by a higher-order attack. Higher-order attacks consider information from several shares simultaneously. Second-order attacks have been shown to be practical to mount and hence developing methods to protect implementations is of importance.
When implemented in hardware, masking can lead to insecure designs. Standard CMOS gates can glitch, and these glitches can cause the power consumption to depend on an unexpected number of shares in a nonlinear way. This behavior degrades the security claims. An approach based on properties of masked Boolean functions has been proposed by Nikova *et al.* [15] which provides security even in the presence of glitches in the hardware. An implementation made following

this approach is called a Threshold Implementation. Besides providing security in hardware, Threshold Implementations require minimal randomness costs in their design due to the guaranteed uniformity of their masking. See, for example, the sharing of NOEKEON in the original paper [16], or the more recent AES sharing of Sugawara [20] both of which require no fresh randomness. This significantly reduces costs in hardware and relinquishes the costs of a cryptographic strong random number generator.
While Threshold Implementations have provable security against first-order attacks, they can often be practically broken using higher-order attacks.

Contribution. The main contribution of the work is the proposition of a new notion, called resilient uniformity. This notion is a generalization of the uniformity property of Threshold Implementations. We formally prove that functions achieving the notion of resilient uniformity provide sufficient security against second-order probing adversaries.

We use the new notion to show that the work by Reparaz *et al.* [18] called "Consolidating Masking Schemes", provides provable second-order protection as the scheme is a specific application of resilient uniformity. Finally, we apply the notion to create a second-order masking of Present. This masking is made to minimize randomness costs requiring only 686 bits of randomness to secure the state. The previous record on randomness cost of a second-order masked Present was given by Cassiers *et al.* [5] which requires 5952 random bits.

2 Preliminaries

In this section we introduce the main building blocks in the context of our work.

2.1 Boolean Masking and Threshold Implementations

A popular method to defend against side-channel attacks is masking which was independently introduced by Goubin and Patarin [10] and Chari *et al.* [6]. The technique is based on splitting each secret variable x in the circuit into shares $\bar{x} = (x^1, x^2, \ldots, x^{s_x})$, such that $x = \sum_{i=1}^{s_x} x^i$ over a finite field $\mathbb{F}_2^n$. A random Boolean masking of a fixed secret is uniform if all maskings of that secret are equally likely. Define $Sh(x)$ as the set of possible maskings of the secret x, for example, for two-shared Boolean masking $Sh(0) = \{(a, a) \mid a \in \mathbb{F}_2\}$.

Definition 1 (uniform masking). *A random masking $\bar{X}$ in s_x shares is uniform if for all $x \in \mathbb{F}_2^n$ we have*

$$P(\bar{X} = \bar{x} \,|\, X = x) = \begin{cases} 2^{-n(s_x-1)} & \text{if } \bar{x} \in Sh(x)\,, \\ 0 & \text{else}\,. \end{cases}$$

Intuitively, an adversary observing the power consumption of a masked implementation needs to combine several time samples, namely those related to each share, in order to have data correlated to the secrets of the implementation. This combination makes it exponentially more difficult to recover secret values. Masking a variable is simple, but masking computation can be much more challenging as care must be taken to ensure no weaknesses are introduced which reduce its security. This is especially challenging in hardware implementations due to the presence of glitches. Nikova *et al.* [15] introduced Threshold Implementations as a particular approach to share circuits and functions. In the following, the main properties of Threshold Implementations are reviewed.

A Threshold Implementation consists of several layers of Boolean functions, as shown in Figure 1. As for any masked implementation, a black-box encoder function generates a uniform masking of the input before it enters the shared circuit and the output shares are recombined by a decoder function. At the end of each layer, synchronization is ensured by means of registers.

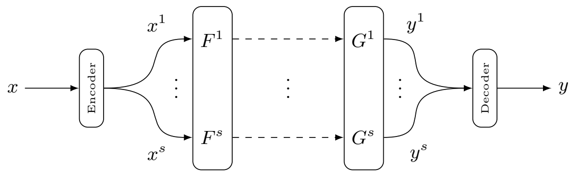


Figure 1: Representation of a threshold circuit assuming an equal number of input and output shares [1].

Let $\bar{F}$ be a layer in the Threshold Implementation corresponding to a part of the circuit $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$. For example, F might be an S-box of a cipher. The function $\bar{F} : \mathbb{F}_2^{ns_x} \rightarrow \mathbb{F}_2^{ms_y}$, where we assume s_x shares per input bit and s_y shares per output bit, will be called a *masking* of F. A share of a share function $\bar{F}$ is denoted by $F^i : \mathbb{F}_2^{ns_x} \rightarrow \mathbb{F}_2^m$, for $i \in \{1, .., s_y\}$. The properties of Threshold Implementations are summarized in Definition 2.

Definition 2 (properties of threshold implementations [15]). *Let $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$ be a function and $\bar{F} : \mathbb{F}_2^{ns_x} \rightarrow \mathbb{F}_2^{ms_y}$ a masking of F. The masking $\bar{F}$ is said to be*

1. correct *if $\sum_{i=1}^{s_y} F^i(x^1, \ldots, x^{s_x}) = F(x)$ for all $x \in \mathbb{F}_2^n$ and for all shares $x^1, \ldots, x^{s_x} \in \mathbb{F}_2^n$, such that $\sum_{i=1}^{s_x} x^i = x$,*
2. non-complete *if each F^i depends on at most $s_x - 1$ input shares,*
3. uniform *if $\bar{F}$ maps a uniform masking of each $x \in \mathbb{F}_2^n$ to a uniform masking of $F(x) \in \mathbb{F}_2^m$.*

If all layers of a Threshold Implementation adhere to the above properties, the resulting shared circuit can be proven secure in the first-order probing model considering glitches, which is the security model defined further on [8]. In the higher-order setting, the situation is more complicated. Perfect multivariate security, when probes are placed in different clock cycles, can not be guaranteed using uniform maskings alone [17]. Instead, the Threshold Implementation approach was generalized for second-order security using fresh randomness by the "Consolidating Masking Schemes" (CMS) approach [18]. Only recently a secure extension of the threshold model was proposed for second-order security [1]. Interestingly enough, this extension makes heavy use of the cryptanalytic properties of the shared cipher such as its diffusion layers and nonlinearity of its S-box. As a result, certain ciphers can be difficult to protect. In particular, PRESENT was highlighted as one such cipher due to the cipher's linear layer and well-known weaknesses with respect to linear cryptanalysis [7]. In this work, we study properties of maskings which defend them against second-order attacks.

2.2 Probing Security

This section introduces the formal adversary and security models. Both are derived from the probing model by Ishai *et al.* [12]. This work considers a countermeasure built according to the Threshold Implementation structure as defined above. Additionally, the extension of the probing model by Faust *et al.* [9] is considered to model glitches.

Consider an algorithm to be a sequence of calculations of Boolean functions. In symbols, we denote $(x_1, ..., x_\ell) \in \mathbb{F}_2^\ell$, such that there exist Boolean functions h_i for which $x_i = h_i((x_j)_{j \in J_{h_i}})$ for a set $J_{h_i} \subset \{1, ..., i-1\}$, meaning that the algorithm can be represented by a sequence of functions h_i, each calculating a new variable x_i, with x_1 as the input of the algorithm and x_ℓ the output.

Let $\ell \geq d$ be positive integers. A *d-threshold-probing adversary on* $\mathbb{F}_2^\ell$ is an algorithm $\mathrm{Adv}_{d\text{-}\mathsf{thr}}$ that interacts as follows with an oracle that computes $(x_1, ..., x_\ell) \in \mathbb{F}_2^\ell$:

1. $\mathrm{Adv}_{d\text{-}\mathsf{thr}}$ specifies a set $\mathcal{I} \subset \{1, ..., \ell\}$, such that $|\mathcal{I}| \leq d$,
2. $\mathrm{Adv}_{d\text{-}\mathsf{thr}}$ then receives all inputs of $(h_i)_{i \in \mathcal{I}}$, *i.e.* all x_j for $j \in \cup_{i \in \mathcal{I}} J_{h_i}$.

The considered security games are derived from the one introduced by Ishai *et al.* [12]. In particular, perfect security is considered where the adversaries are computationally unbounded and must specify probe locations before querying the circuit. Take an arbitrary adversary $\mathrm{Adv}_{d\text{-}\mathsf{thr}}$ and $\mathsf{Dec} \circ C' \circ \mathsf{Enc}$ a randomized stateless sequence (circuit) which is the masked version of C. Importantly, the adversary's interaction with the circuit is mediated through the encoder and decoder algorithms Enc and Dec, neither of which can be probed. There are two notions of security for stateless circuits. The circuit C' is considered *sound* if for all inputs k, $\mathsf{Dec}(C'(\mathsf{Enc}(k))) = C(k)$. The circuit C' is d^{th}-order *private* if it can be simulated from scratch, meaning the simulator is given nothing, such that no adversary $\mathrm{Adv}_{d\text{-}\mathsf{thr}}$ can distinguish $\mathsf{Dec} \circ C' \circ \mathsf{Enc}$ from the simulation. A circuit is considered d^{th}-order probing secure if it is both sound and d^{th}-order private following the above games.

2.3 Adversary Structures

A key component to generalize the notion of uniformity is the notion of an adversary structure. To that end, this section provides the required definitions.

Consider a secret x is shared in s shares $X = \{x^1, ..., x^s\}$. Let us denote all subsets of X by $P(X)$. We call the groups of shares which do not reconstruct the secret as unqualified. The set of unqualified groups, also called the privacy structure, is denoted by $\Delta \subset P(X)$. Let us denote Δ^+ as the maximal set of Δ, *i.e.* the elements in Δ for which no proper superset is also in Δ. In this work, we consider Boolean masking for which Δ consists of all sets containing $s-1$ or fewer shares. For example, consider a secret x is shared into three shares x^1, x^2, x^3, such that $\sum_{i=1}^{3} x^i = x$ then

$$\Delta^+ = \{\{x^1, x^2\}, \{x^2, x^3\}, \{x^1, x^3\}\}.$$

In case multiple secrets are considered, the privacy structure would consist of all sets containing $s-1$ or fewer shares of each secret. For example, for two secrets x, y shared into two shares

$$\Delta^+ = \{\{x^1, y^1\}, \{x^1, y^2\}, \{x^2, y^1\}, \{x^2, y^2\}\}.$$

We denote the *adversary structure* $\Delta_{\mathcal{A}}$ as a monotone decreasing subset of Δ, *i.e.* for all sets in $\Delta_{\mathcal{A}}$ each of their subsets is also in $\Delta_{\mathcal{A}}$. Intuitively, in this work the adversary structure will contain all possible sets of shares a probing adversary can view using a single probe. Conversely, using a probe, an adversary can thus see one element from Δ.

Hirt and Maurer introduced the notion of $\mathcal{Q}^\ell$ adversary structures [11]. An adversary structure $\Delta_{\mathcal{A}}$ is $\mathcal{Q}^\ell$ if no ℓ sets in $\Delta_{\mathcal{A}}$ cover all shares of a secret. In this work, a focus is put on second-order probing security, as such we will work with $\mathcal{Q}^2$ adversary structures.

2.4 Resiliency

The notion of resilient uniformity resembles the notion of resiliency. As such, this section briefly introduces resiliency as first given by Siegenthaler [19].

Recall that a balanced Boolean function is one which gives an output, such that each output vector is equally likely to occur if given a random input. In particular, permutations are balanced. The notion of balancedness is then generalized to resiliency which we give in terms of monotone decreasing sets Δ following the work by Braeken *et al.* [4].

Definition 3 (Δ resilient). *Let $f(x)$ be a Boolean function and Δ be a monotone decreasing set. Then $f(x)$ is called Δ resilient if any of its restrictions obtained by fixing an input set $A \in \Delta$ of inputs is balanced.*

3 Resilient Uniformity

This is the main section of the paper introducing the notion of resilient uniformity which provides second-order probing security of Threshold Implementations. This notion can be seen as the masking's equivalent of a resilient Boolean function. While this section expands the notions of non-completeness and uniformity, the next subsection proves the second-order probing security using these notions. In Section 4 the notion of resilient uniformity is applied to the "Consolidating Masking Schemes" (CMS) approach [18] and in Section 5 to make a second-order masking of the PRESENT cipher.

We start by generalizing the notion of non-completeness. Recall that the original notion requires that each share of a function can only work on a limited number of shares namely that each share of a function can not work on all shares of a secret. By using adversary structures as introduced in Section 2.3, we give a more generalized notion of non-completeness by tightening this limitation.

Definition 4 (Δ non-completeness). *A function $\bar{F}$ is Δ non-complete, for an adversary structure Δ, if each share of $\bar{F}$ only uses inputs A for an $A \in \Delta$.*

Thus, the above notion states that each share of the function can only work on a set of input shares specified by the adversary structure Δ. We provide an example of Δ non-completeness. Consider the three-sharing of the multiplication $F(x, y) = xy$

$$F^i = x^i y^i + x^i y^{i+1} + x^{i+1} y^i,$$

where the convention is used that superscripts wrap around at three. The above function is Δ non-complete with

$$\Delta^+ = \{\{x^1, x^2, y^1, y^2\}, \{x^2, x^3, y^2, y^3\}, \{x^1, x^3, y^1, y^3\}\}.$$

In particular, the above Δ is a $\mathcal{Q}^1$ adversary structure as probing two shares of $\bar{F}$ gives back all input shares. Equivalently, taking two elements in Δ^+ covers all shares of a secret, in this case both x and y.

There is a connection between Δ non-completeness and higher-order non-completeness by Bilgin *et al.* [2].

Definition 5 (d^{th}-order non-completeness). *A shared function $\bar{F}(\bar{x})$ is d^{th}-order non-complete if every set of d shares of $\bar{F}$ jointly operate on at most $s_x - 1$ input shares.*

It is clear that if Δ is a $\mathcal{Q}^d$ adversary structure then a Δ non-complete function is also d^{th}-order non-complete.

Now, assume that each shared function in the Threshold Implementation is Δ non-complete. Looking back at the notion of resiliency from Section 2.4, we transform the notion of uniformity to its resilient equivalent. A first attempt in this transformation would give a notion where fixing certain inputs to constants again gives a uniform function thus ensuring that the output is still uniform even after one probe was placed. Nevertheless, this simple adaptation is not sufficient. As a glitch-extended probe gives back all input shares of a share of a function, the output of a probed function can never be uniform. An adaptation to the notion of resiliency is necessary.

We denote a *restriction* of a masking $\bar{x}$ on a set of shares A by $\bar{x}_A$ to be the set of shares of $\bar{x}$ indicated by A. For example, for a three-sharing $\bar{x} = (x^1, x^2, x^3) = (0, 1, 0)$ and $A = \{x^1, x^2\}$ then $\bar{x}_A = (0, 1)$. We start with the definition of a Δ resilient uniform masking.

Definition 6 (Δ resilient uniform masking). *A masking of a variable x is Δ resilient uniform if there is a set $I \in \Delta$, such that when $\bar{x}_I$ are fixed to constants then $\bar{x}_{X \setminus I}$ is uniform, with X the set of all shares.*

For example, taking a three-shared Boolean masking of a secret $x \in \mathbb{F}_2$, $Sh(x) = \{(x^1, x^2, x^3) \mid \sum_{i=1}^{3} x^i = x\}$. Considering Δ as the set of sets containing at most one share, *i.e.* $\Delta = \{\{x^1\}, \{x^2\}, \{x^3\}, \emptyset\}$. The sharing with one share set to zero would be a Δ resilient uniform masking. In other words, $\{(0, x^2, x^3) \mid x^2 + x^3 = x\}$ would no longer be uniform, but it is a Δ resilient uniform masking.

Having defined what is a Δ resilient uniform masking, a (Δ, Δ') resilient uniform function is a function which maps a Δ resilient input masking to a Δ' resilient output masking. The definition of a (Δ, Δ') resilient uniform function is given in its combinatorial form.

Definition 7 ((Δ, Δ') resilient uniformity). *A function $\bar{F}(\bar{x}) = \bar{y}$ is (Δ, Δ') resilient uniform if $\forall I \in \Delta$, $\exists J \in \Delta'$, such that $\forall \bar{x}_I^* \in \mathbb{F}_2^{|I|}$, $\bar{y}_J^* \in \mathbb{F}_2^{|J|}$, $\exists c \in \mathbb{N}$, such that $\forall x \in \mathbb{F}_2^n$ and $\bar{y} \in Sh(F(x))$ with $\bar{y}_J = \bar{y}_J^*$:*

$$\left| \left\{ \bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}_I^* \,\middle|\, \bar{F}(\bar{x}) = \bar{y} \right\} \right| = c\,.$$

Later on, in Section 4 we will see that each second-order non-complete function that is re-masked via the ring refreshing method achieves a notion of (Δ, Δ') resilient uniformity. In Section 5, we show that the cost of this refreshing can be lowered if the shared functions are uniform.

3.1 Proving Second-Order Probing Security

Having the notion of resilient uniformity, we prove that it is sufficient for perfect second-order probing security of Threshold Implementations as defined in Section 2.2. We start by showing that a (Δ, Δ') resilient uniform function indeed maps a Δ resilient uniform input masking to a Δ' resilient uniform output masking.

Lemma 1. *A (Δ, Δ') resilient uniform function maps a Δ resilient uniform input masking to a Δ' resilient uniform output masking.*

Proof. Consider a random input masking $\bar{x}$ of the secret x which is Δ resilient uniform. Meaning there is a set $I \in \Delta$, such that $\bar{x}_I \in \mathbb{F}_2^{|I|}$ are fixed to constants, denote these by $\bar{x}_I^*$, and $\bar{x}_{X \setminus I}$ is uniform. Also consider $\bar{F}$ a (Δ, Δ') resilient uniform function. We show that there exists a set $J \in \Delta'$, such that the output $\bar{y} = \bar{F}(\bar{x})$ is a Δ' resilient uniform masking for that J.

From the definition of (Δ, Δ') resilient uniformity, we know that there exists a set $J \in \Delta'$, such that for all $\bar{y}_J^* \in \mathbb{F}_2^{|J|}$, there exists a $c \in \mathbb{N}$, such that $\forall \bar{y} \in Sh(F(x))$ with $\bar{y}_J = \bar{y}_J^*$:

$$\left| \left\{ \bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}_I^* \,\middle|\, \bar{F}(\bar{x}) = \bar{y} \right\} \right| = c\,.$$

Since $\bar{F}$ is a function, there exists at least one vector of constants $\bar{y}^*_J$, such that $c \neq 0$. We claim that $\bar{y}$ is Δ' resilient uniform for $J \in \Delta'$ and the previous mentioned set of constants. Indeed, each instance $\bar{y}$ with $\bar{y}_J = \bar{y}^*_J$ occurs equally frequently, namely with frequency $c \neq 0$. As a result $\bar{y}_{Y \setminus J}$ is uniform and thus $\bar{y}$ is Δ' resilient uniform. □

We then show that the notion of resilient uniformity is stronger than regular uniformity.

Lemma 2. *A (Δ, Δ') resilient uniform function $\bar{F}$ is uniform.*

Proof. We take an arbitrary $I \in \Delta$ and an arbitrary $\bar{x}^*_I \in \mathbb{F}_2^{|I|}$. We know there is a $J \in \Delta'$, such that for all $\bar{y}^*_J \in \mathbb{F}_2^{|J|}$, there is a constant c, such that for all x and $\bar{y} \in Sh(F(x))$ with $\bar{y}_J = \bar{y}^*_J$:

$$|\{\bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}^*_I \,|\, \bar{F}(\bar{x}) = \bar{y}\}| = c\,.$$

We pick a $\bar{y}^*_J \in \mathbb{F}_2^{|J|}$ such that c does not equal to zero.

We now take another $\bar{x}'_I \in \mathbb{F}_2^{|I|}$, then we find that there is another constant c', such that for all x and $\bar{y} \in Sh(F(x))$ with $\bar{y}_J = \bar{y}^*_J$:

$$|\{\bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}'_I \,|\, \bar{F}(\bar{x}) = \bar{y}\}| = c'\,.$$

By taking the sum of these two we find that for all x and $\bar{y} \in Sh(F(x))$ with $\bar{y}_J = \bar{y}^*_J$:

$$|\{\bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}^*_I \text{ or } \bar{x}_I = \bar{x}'_I \,|\, \bar{F}(\bar{x}) = \bar{y}\}| = c + c'\,.$$

Thus, by looping over all possible $\bar{x}_I$, we find that there is a constant c^*, such that for all x and $\bar{y} \in Sh(F(x))$ with $\bar{y}_J = \bar{y}^*_J$:

$$|\{\bar{x} \in Sh(x) \,|\, \bar{F}(\bar{x}) = \bar{y}\}| = c^*\,.$$

Since each share in $\bar{y}$ and $\bar{x}$ is fixed, c^* needs to equal $2^{n(s_x-1)}/2^{m(s_y-1)}$ if $x \in \mathbb{F}_2^n$ with s_x input shares and $y \in \mathbb{F}_2^m$ with s_y output shares. Since the constant c^* has the same value for each choice of secret x and $\bar{y}^*_J$, we have shown the uniformity of $\bar{F}$. □

There is also a link between resilient uniform functions of different adversary structure. Namely a (Δ, Δ') resilient uniform function is (Δ, Δ'') resilient uniform when $\Delta' \subset \Delta''$.

Lemma 3. *Given two adversary structures $\Delta' \subset \Delta''$, a (Δ, Δ') resilient uniform function is also (Δ, Δ'') resilient uniform.*

Proof. Let $\bar{F}$ be a (Δ, Δ') resilient uniform function. We prove that $\bar{F}$ is (Δ, Δ'') resilient uniform. Take an arbitrary $I \in \Delta$, then from the (Δ, Δ') resilient uniformity, there there exists a $J \in \Delta'$, such that for all $\bar{x}^*_I \in \mathbb{F}_2^{|I|}$ and all $\bar{y}^*_J \in \mathbb{F}_2^{|J|}$ there exists a constant c, such that for all x and $\bar{y} \in Sh(F(x))$ with $\bar{y}_J = \bar{y}^*_J$:

$$\left|\,\left\{\bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}^*_I \,\middle|\, \bar{F}(\bar{x}) = \bar{y}\right\}\,\right| = c\,.$$

However, since $\Delta' \subset \Delta''$, J is also an element of Δ''. Thus, we can always find a suitable $\bar{y}^*_{J'}$ with $J' \in \Delta''$ for which the necessary property holds. □

Finally, we show that the composition between resilient uniform functions is again resilient uniform.

Lemma 4. *The composition between a (Δ, Δ') resilient uniform function and a (Δ', Δ'') resilient uniform functions is (Δ, Δ'') resilient uniform, given that the functions can compose.*

Proof. Let $\bar{F}$ be a (Δ', Δ'') resilient uniform function and $\bar{G}$ a (Δ, Δ') resilient uniform function such that the composition $\bar{F} \circ \bar{G}$ makes sense, we prove that $\bar{F} \circ \bar{G}$ is (Δ, Δ'') resilient uniform. We take an arbitrary $I \in \Delta$. From $\bar{G}$ being (Δ, Δ') resilient uniform, we know that there exists a $J \in \Delta'$, such that for all $\bar{x}^*_I \in \mathbb{F}_2^{|I|}$ and all $\bar{y}^*_J \in \mathbb{F}_2^{|J|}$ there exists a constant c, such that for all x and $\bar{y} \in Sh(G(x))$ with $\bar{y}_J = \bar{y}^*_J$:

$$\left| \left\{ \bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}^*_I \,\middle|\, \bar{G}(\bar{x}) = \bar{y} \right\} \right| = c\,.$$

From $\bar{F}$ being (Δ', Δ'') resilient uniform we also know that there exists a $K \in \Delta''$, such that for all $\bar{y}^*_J \in \mathbb{F}_2^{|J|}$ and $\bar{z}^*_K \in \mathbb{F}_2^{|K|}$ there exists a constant c', such that for all y and $\bar{z} \in Sh(F(y))$ with $\bar{z}_K = \bar{z}^*_K$:

$$\left| \left\{ \bar{y} \in Sh(y) \text{ with } \bar{y}_J = \bar{y}^*_J \,\middle|\, \bar{F}(\bar{y}) = \bar{z} \right\} \right| = c'\,.$$

Thus, we find that there are a constant number of $\bar{y}$ vectors with $\bar{y}_J = \bar{y}^*_J$ which are mapped to a vector $\bar{z}$ with $\bar{z}_K = \bar{z}^*_K$. For each of these $\bar{y}$ vectors we find a constant number of $\bar{x}$ vectors with $\bar{x}_I = \bar{x}^*_I$ that are mapped to that $\bar{y}$. Thus, we find that there exists a set K, such that for all $\bar{x}^*_I \in \mathbb{F}_2^{|I|}$ and $\bar{z}^*_K \in \mathbb{F}_2^{|K|}$ there exists a constant c^*, namely $c^* = cc'$ or $c^* = 0$, such that for all x and $\bar{z} \in Sh(F(G(x)))$ with $\bar{z}_K = \bar{z}^*_K$:

$$\left| \left\{ \bar{x} \in Sh(x) \text{ with } \bar{x}_I = \bar{x}^*_I \,\middle|\, \bar{F}(\bar{G}(\bar{x})) = \bar{z} \right\} \right| = c^*\,.$$

Since I was chosen arbitrarily, we have proven the lemma. □

The above two lemmas form a compositional security argument effectively showing that (Δ, Δ') resilient uniformity is sequentially composable. We now show that the notion of (Δ, Δ') resilient uniformity leads to second-order probing security. We require that each function in the circuit is $(\Delta_i, \Delta'_{i+1})$ resilient uniform such that $\Delta'_i \subset \Delta_i$ with i the index enumerating the shared functions in the circuit.

Theorem 1. *The composition of shared functions $\bar{F}_i$ which are correct, Δ_i non-complete, and $(\Delta_i, \Delta'_{i+1})$ resilient uniform, with each Δ_i, Δ'_i being $\mathcal{Q}^2$ adversary structures and $\Delta'_i \subset \Delta_i$, is secure against a second-order probing adversary.*

Proof. Due to all functions being correct, the circuit is also correct. We show the circuit is also second-order private as defined in Section 2.2. Recall that privacy is shown as the existence of a simulator who can simulate the probed values from scratch. Take two arbitrary probed functions from the threshold circuit and give the adversary all the inputs of those functions. We denote the set of shares given to the adversary by $\mathcal{I}$.

We treat the case where the two probed functions lie in the same shared function $\bar{F}_i$. Due to the Δ_i non-completeness of $\bar{F}_i$ and since Δ_i is a $\mathcal{Q}^2$ adversary structure, $\mathcal{I}$ never contains all shares of an input. The security of this case relies only on the original definition of uniformity. Since each shared function is also uniform, the input masking was also uniform, thus each input share can be simulated as uniformly random.

We look at the case where the two probed functions do not lie in the same shared function. Denote the to probed functions $\bar{F}_i$ and $\bar{F}_j$ with $i \neq j$. Due to the Δ_i and Δ_j non-completeness of both functions, $\mathcal{I}$ consists of a set of shares $\bar{x}_I$ with $I \in \Delta_i$ and a set of shares $\bar{y}_J$ with $J \in \Delta_j$. Since the first probed function has a uniform input, the shares $\bar{x}_I$ can be simulated as uniform randomness. Consider the function $\bar{F}$ which maps the input of $\bar{F}_i$ to the input of $\bar{F}_j$.

We first prove that $\bar{F}$ is (Δ_i, Δ'_j) resilient uniform. We know that $\bar{F}$ is the composition of functions which are $(\Delta_k, \Delta'_{k+1})$ resilient uniform, for $k \in \{i, ..., j-1\}$. From Lemma 3, we know from each $\Delta'_k \subset \Delta_k$, that each function is also (Δ_k, Δ_{k+1}) resilient uniform. From Lemma 4, we then know that the composition of these functions is (Δ_i, Δ_j) resilient uniform. Thus, $\bar{F}$ is (Δ_i, Δ_j) resilient uniform.

As a result, since the input of $\bar{F}$ is a Δ_i resilient uniform masking, the output of $\bar{F}$ is also a Δ_j resilient uniform masking following Lemma 1. Thus, the output of $\bar{F}$ consists of constants and a uniform masking. Since Δ_j is a $\mathcal{Q}^2$ adversary structure, the resulting uniform masking is still non-complete. Thus, the second probe only gives back constants, which are already simulated, or uniform randomness. As a result, all probed values can be simulated from scratch. □

In case functions are only uniform but not (Δ, Δ') resilient, one can find examples such that, when composed, the functions become insecure against a second-order attack. An example is given by Reparaz [17] of a shared identity function using five shares.

$$(a^1, a^2, a^3, a^4, a^5) \to (a^1, a^1 + a^2 + a^4 + a^5, a^1 + a^3 + a^4 + a^5, a^4, a^5)$$

This function is uniform but it is clearly not second-order secure as one probe can read a^3 and the other $a^1 + a^2 + a^4 + a^5$ to reveal a. The above function is not (Δ, Δ') resilient as, for example taking a^3 in the input equal to zero gives the non-uniform output distribution $(a^1, a^1 + a^2 + a^4 + a^5, a^1 + a^4 + a^5, a^4, a^5)$. Fixing an output coordinate to a constant does not help make the distribution uniform, for example one can fix the third coordinate to zero $(a^1 + a^4 + a^5 = 0)$

and remove it. However, the resulting output distribution $(a^1, a^2, a^4, a^1 + a^4)$ is again non-uniform. This will be the case for any other output coordinate one fixes.

4 Consolidating Masking Schemes Revisited

This section applies the notion of resilient uniformity to the "Consolidating Masking Schemes" (CMS) approach [18]. The CMS approach was proposed to secure Threshold Implementations against d^{th}-order probing adversaries using fresh randomness. However, its security was only argued and formal definitions were never given. Later on, Moos *et al.* [14] showed that the CMS approach was insecure against third- or higher-order attacks showing a need for formal security proofs. In this work, we prove the second-order security of the CMS approach.

4.1 The CMS Approach

We quickly introduce the CMS approach which guarantees second-order probing security. In short, the approach takes a second-order non-complete map and refreshes the uniformity by adding extra randomness in a ring refreshing approach. The expanded shares are then re-compressed by XORing them together.

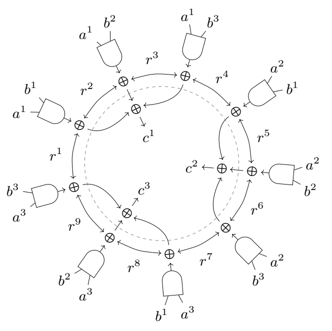
We give an example of the approach, namely for a shared AND gate which is depicted in Figure 2a. First, the CMS approach calculates all the linear and nonlinear terms, of the shared function in this case all cross products $a^i b^j$. These terms are then added together, such that second-order non-completeness still holds, *i.e.* the joint inputs of every pair of functions do not cover all shares of a secret. In the AND gate example, each cross product is held separately as XORing some of them would already invalidate second-order non-completeness. In order to re-compress the shares, they are first added with randomness and synchronized. The addition of randomness follows a "ring refreshing" approach as shown in Figure 2b with the dashed line denoting a register stage.

While the example is that of a multiplication, the CMS approach can be applied to any Boolean function.

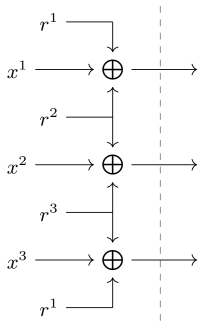
4.2 Applying Resilient Uniformity

We split up the CMS approach in two main layers; An expansion layer which expands the input shares and adds randomness; and a compression layer re-compressing the shares. The two layers are depicted in Figure 3. We show that both layers conform to the notion of resilient uniformity.

To that end we introduce the two main adversary structures. The first is denoted by Δ_{one_share} which is the adversary structure consisting of all sets with one share, in total, and the empty set. Second, for a shared function $\bar{F}$, we denote the natural adversary structure given by all sets of shares a single probe



(a) A second-order secure AND-gate.



(b) Three-shared ring refreshing.

Figure 2: The CMS secure AND gate and refreshing method.

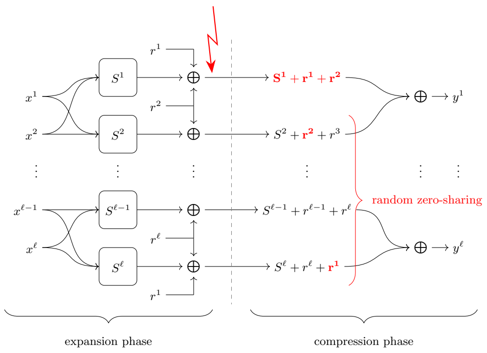


Figure 3: A depiction of the CMS approach. The gray dashed line denotes a register stage. The adversary probes the first expanded share. Values known to the adversary are denoted in bold red.

can observe by $\mathit{\Delta}_{probing}$. For example, consider the function $G(x, y, z) = xy + z$ which is shared in the following way

$$G^i = x^i y^i + x^i y^{i+1} + x^{i+1} y^i + z^i,$$

where the convention is used that superscripts wrap around at three. Then $\mathit{\Delta}_{one\_share}$ and $\mathit{\Delta}^+_{probing}$ equal the following sets

$$\mathit{\Delta}_{one\_share} = \{\{x^1\}, \{x^2\}, \{x^3\}, \{y^1\}, \{y^2\}, \{y^3\}, \{z^1\}, \{z^2\}, \{z^3\}, \emptyset\},$$

$$\mathit{\Delta}^+_{probing} = \{\{x^1, x^2, y^1, y^2, z^1\}, \{x^2, x^3, y^2, y^3, z^2\}, \{x^1, x^3, y^1, y^3, z^3\}\}.$$

We claim that if a function is shared according to the CMS approach, it is $(\mathit{\Delta}_{probing}, \mathit{\Delta}_{one\_share})$ resilient uniform.

We show the expansion phase of the CMS is $(\mathit{\Delta}_{probing}, \mathit{\Delta}_{one\_share})$ resilient uniform. This expansion phase $\bar{F}$ is a $\mathbb{F}_2^{m+n} \to \mathbb{F}_2^m$ function with inputs

$(a^1, ..., a^n, r_1, ..., r_m)$ given by the equations

$$F^i = G^i(a^1, ..., a^n) \oplus r_i \oplus r_{i+1} \,,$$

for a second-order non-complete function $\bar{G}$, $i \in \{1, ..., m\}$ and where the subscripts wrap around at m.

Theorem 2. *The function $\bar{F}$ as defined above is $(\Delta_{probing}, \Delta_{one\_share})$ resilient uniform.*

Proof. We take an arbitrary set I from $\Delta_{probing}$. Without loss of generality we assume this set consists of the elements $\{r_i, r_{i+1}\}$ for some $i \in \{1, ..., m\}$ and a non-complete set of shares of $\bar{a}$, *i.e.* a set which misses at least one share of $\bar{a}$, which we denote by A. We take $J = \{F^i\}$, set $\{A, r_i, r_{i+1}\}$ to arbitrary constants $\bar{a}^*_A \in \mathbb{F}_2^{|A|}, r^*_i \in \mathbb{F}_2, r^*_{i+1} \in \mathbb{F}_2$, and set $F^i = G^i(a^1, ..., a^n) \oplus r_i \oplus r_{i+1}$. Setting F^i to a different value would make the proof evident for $c = 0$. We show that there exists a constant c, such that for an arbitrary a and $\bar{y} \in Sh(F(a))$ where $F^i = y^i = G^i(a^1, ..., a^n) \oplus r_i \oplus r_{i+1}$, there are c inputs $(a^1, ..., a^n, r_1, ..., r_m)$, adhering to the constraints $\bar{a}_A = \bar{a}^*_A, r_i = r^*_i, r_{i+1} = r^*_{i+1}$, mapping to $\bar{y}$.

Since A consists of a non-complete set of shares, we find that for each secret a there are a constant number, denote this c', of share vectors $\bar{a} \in Sh(a)$ with $\bar{a}_A = \bar{a}^*_A$. We now take an arbitrary vector of shares $\bar{a}$ from the c' previous possibilities. We show that for each such vector we find unique values $\{r_1, ..., r_m\}$, such that $(a^1, ..., a^n, r_1, ..., r_m)$ maps to $\bar{y}$. This is evident, since the shares $\{a^1, ..., a^n\}$ are already fixed, we set

$$r_j \oplus r_{j+1} = y^j \oplus G^j(a^1, ..., a^n) \,.$$

The above gives $m - 1$ linear equations with $m - 2$ variables (excluding the equation for F^i and removing the variables r_i, r_{i+1}). Remark that due to the correctness property, one equation is the sum of all others. Removing that linear dependent equation, we have $m - 2$ equations and $m - 2$ variables. It is clear that one can assign a unique r_j to each equation, thus each remaining equation is linear independent and there is a unique solution for $\{r_1, ..., r_m\}$. As a result, we find that $c = c'$ and thus for each a and $\bar{y} \in Sh(F(a))$ where $y^i = G^i(a^1, ..., a^n) \oplus r_i \oplus r_{i+1}$ we find there are c' $\Delta_{probing}$-restricted inputs which map to $\bar{y}$. □

Thus, the expansion phase is $(\Delta_{probing}, \Delta_{one\_share})$ resilient uniform. The second phase, the compression phase, maps each expanded share to only one output. As a result, the inputs viewed by a probe are only used for one output share and thus a $\Delta_{probing}$ resilient input gets mapped to a Δ_{one_share} resilient output meaning that the second phase is $(\Delta_{probing}, \Delta_{one\_share})$ resilient uniform.

Since both layers adhere to the same notion of resilient uniformity, the entire CMS approach is $(\Delta_{probing}, \Delta_{one\_share})$ resilient uniform. Since $\Delta_{one\_share} \subset \Delta_{probing}$, from Theorem 1 we know that the approach offers second-order probing security.

5 Application to PRESENT

In this section, we give a sharing of the Present cipher which has a low-cost requirement for fresh randomness.

Intuitively, the Present round operations are sparse, meaning each operation only operates on a small portion of the state at a time. This sparsity together with using uniform shared functions allows for a relaxation for randomness cost to achieve a notion of resilient uniformity.

For simplicity for the security arguments, we only consider protecting the state function of Present. The key expansion function is left out of the analysis. As such, we consider the key as a constant throughout the state computation.

5.1 PRESENT

We introduce the Present-80 cipher, denoted Present, from the work of Bogdanov *et al.* [3]. Present accepts as input a 64-bit plaintext m and considers a 80 bit key. Present consists of 31 rounds, each comprising an XOR with the roundkey (`addRoundKey`), a substitution layer (`sBoxLayer`), and a bit permutation layer (`pLayer`). The cipher is depicted in Figure 4.

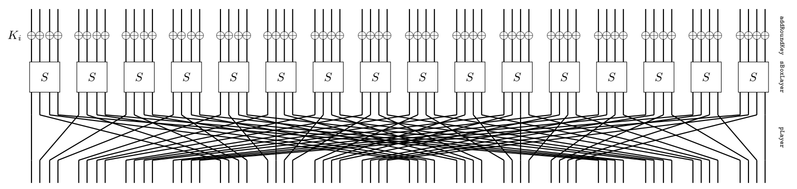


Figure 4: A round of Present.

5.2 Seven-Shared PRESENT

Following the notions explained in Section 3, this section constructs a masking of the Present cipher. Figure 5 gives an overview of the shared round function.

Masking state and key. For the masking of Present we use classical Boolean masking. The 64-bit state is shared using seven shares per bit. The 80-bit key is shared using seven shares and is assumed to be already split into shared round keys.

Masking affine operations. The masking of Present's linear operations such as the `pLayer`, the linear layers of the decomposed S-box, and the key addition are simply done share-wise. Constants are added to the first share of the corresponding variable.

Masking the S-box. Following Kutzner *et al.* [13], the PRESENT S-box is decomposed into two quadratic maps $S_1 = G \circ C$ and $S_2 = B \circ G$ where B and C are affine. Further details on this decomposition are given in Appendix A.1. The masking of the S-box is constructed from the masking of G which is inspired from the work by Beyne et al. [1] and is detailed in Appendix A.2. This masking consists of seven input shares and seven output shares and is uniform and second-order non-complete. We refer to the software added to the submission for this verification. After the $\bar{S}_1$ and $\bar{S}_2$ operations, randomness is added to make the operation resilient uniform. Both $\bar{r}_1$, $\bar{r}_2$ are ring refreshings as shown in Figure 2b. However, the random bits $\bar{r}_1, \bar{r}_2$ are re-used for every cell in the round. The details of this randomness is given in Appendix A.2. In total, the masking of PRESENT requires 49 fresh random bits per round.

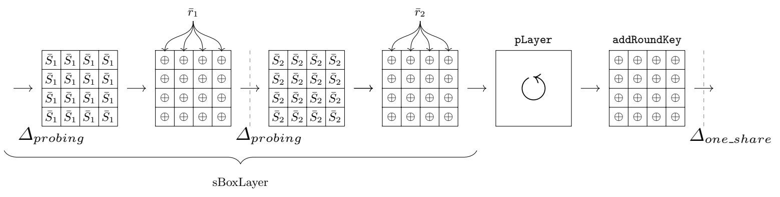


Figure 5: The adversary structures for the seven-shared PRESENT. The register stages are denoted by dashed lines.

Security. Figure 5 depicts the adversary structures considered and the location of register stages. The security argumentation is split for the two operations $\bar{F}_1 = \bar{r}_1 \circ \bar{S}_1$ and $\bar{F}_2 = \texttt{addRoundKey} \circ \texttt{pLayer} \circ \bar{r}_2 \circ \bar{S}_2$. We use the adversary structures $\Delta_{probing}$ and Δ_{one_share} as introduced in Section 4.2.

The first operation $\bar{F}_1$ consists of an affine layer, a nonlinear map $\bar{G}$, and a layer of added randomness. In view of the mathematical construction, this randomness is generated by the encoder function at the state of the masking and is mapped through identity functions until the randomness is needed in the operation. This does not need to be reflecting in the implementation as the randomness can be generated by a true random bit generator or a cryptographic secure generator.

Each share of $\bar{G}$ uses only three shares per input. Since each secret is shared in seven shares, $\Delta_{probing}$ is a $\mathcal{Q}^2$ structure. We show the first operation is $(\Delta_{probing}, \Delta_{one\_share})$ resilient uniform.

Theorem 3. *The function $\bar{F}_1$ is $(\Delta_{probing}, \Delta_{one\_share})$ resilient uniform.*

Proof. First, note that $\Delta_{probing}$ consists of sets of input shares of a single S-box together with the added randomness of $\bar{r}_1$. Take an arbitrary set I from $\Delta_{probing}$. Consider I consists of the input shares of the i^{th} S-box in the state and some randomness from $\bar{r}_1$. Since $\bar{r}_1$ is a ring refreshing operation, we know from Theorem 2 that the output of the i^{th} S-box is $\varDelta_{one\_share}$ resilient uniform. Because all other masked S-boxes are uniform even without the added randomness $\bar{r}_1$ their output is uniform even when the first S-box is probed. The concatenation of a $\varDelta$ resilient uniform masking with a uniform masking is again evidently a $\varDelta$ resilient uniform masking. As a result, the output of $\bar{F}_1$ is $\varDelta_{one\_share}$ resilient uniform. □

The second operation $\bar{F}_2$ consists of an affine layer, a nonlinear map $\bar{G}$, a layer of added randomness, the bit-wise permutation, and the addition of a round key. Similar to the first operation, the structure $\varDelta_{probing}$ for the second operation is also $\mathcal{Q}^2$ since the bit-wise permutations and round key additions are done share-wise. This operation is also $(\varDelta_{probing}, \varDelta_{one\_share})$ resilient uniform.

Theorem 4. *The function $\bar{F}_2$ is $(\varDelta_{probing}, \varDelta_{one\_share})$ resilient uniform.*

Proof. The proof is similar as for the first operation. Due to the `pLayer` working bit-wise, $\varDelta_{probing}$ still consists of inputs of a single S-box. Due to the randomness layer $\bar{r}_2$ being a ring refreshing, the operation is $(\varDelta_{probing}, \varDelta_{one\_share})$ resilient uniform. □

Since all operations in the shared Present are $(\varDelta_{probing}, \varDelta_{one\_share})$ resilient uniform, the entire sharing is second-order probing secure.

5.3 Moving to Bounded-Query Security

The above masking of Present requires a total of 1519 random bits (49 bits per round, for 31 rounds) to secure the state function. We can further reduce this cost to 686 bits (49 bits per round, for 14 rounds) by moving from perfect security, given perfect independent and identically distributed randomness, to bounded-query security. We argue that re-using the randomness every 14 rounds is secure as long as the adversary is limited in the number of probing queries. More specifically, we move from the simulation based security model as defined in Section 2.2 to the bounded query model defined by Beyne et al. [1] which defines a security model where the adversary is admitted only a limited number of probing queries. In the bounded query model, we make use of linear cryptanalytic security arguments using a trail-wise approach. The interested reader is referred to the work by Beyne et al. [1] for the definition of correlation matrices over maskings and how the coefficients of the Fourier transform of the probed distribution bound the advantage of the adversary.

The work by Beyne *et al.* verified that the maximum absolute correlation of $\bar{S}_2 \circ \bar{r}_1 \circ \bar{S}_1$ equals 2^{-3}. From the uniformity of the masked Present's round function we know that at least one masked S-box is active per round, thus the correlation of trails spanning at least 14 rounds is bounded by 2^{-42}. It then follows that the 2-norm of the nontrivial Fourier coefficients of the probed bits can be upper bounded by 2^{-64} where we have used that the support of the Fourier transform is bounded by 2^{20}, which follows from the fact if an output coordinate of $\bar{G}$ is read, at most 10 shares are learned. Thus, the advantage of a bounded query second-order probing adversary can be bounded by

$$\mathrm{Adv}_{\mathsf{2\text{-}thr}}(\mathcal{A}) \leq \sqrt{\frac{q}{2^{62}}}\,,$$

with q the number of probing queries by $\mathrm{Adv}_{\mathsf{2\text{-}thr}}$, which for a large number of queries admits the same advantage as the security achieved by the cipher in most modes of operation.

6 Conclusion and Future Work

In this work, a notion of uniformity was discussed which provides second-order side-channel protection. The notion is called "resilient uniformity" as it resembles resiliency of Boolean functions. The work formally proves its second-order protection in the framework of perfect probing security.

The notion is used to formally prove the security of the masking method of Reparaz *et al.* [18]. Specific applications of the new notion also lead to the significant reduction of randomness of maskings. This was shown by proposing a second-order masking of the PRESENT cipher. Previous work by Cassiers *et al.* [5] details a second-order masking of PRESENT. In the low-randomness variant in their work, they report using 5952 random bits to secure PRESENT's state function. In contrast, the design in this work requires only 686 random bits improving the randomness cost eight times over.

Future work involves research into resilient uniform maskings of symmetric primitives and the improvement of this work's PRESENT masking in terms of randomness but also area and latency. Even though the main theoretic foundations are laid out, the generalization to higher-order security is an interesting investigation to find whether randomness costs remain low.

Acknowledgment. We acknowledge Vincent Rijmen and Venci Nikov for helpful comments. This work was supported by CyberSecurity Research Flanders with reference number VR20192203. Siemen Dhooghe is supported by a PhD Fellowship from the Research Foundation – Flanders (FWO). Svetla Nikova was partially supported by the Bulgarian National Science Fund, Contract No. 12/8.

References

1. Beyne, T., Dhooghe, S., Zhang, Z.: Cryptanalysis of masked ciphers: A not so random idea. IACR Cryptol. ePrint Arch. **2020**, 993 (2020), https://eprint.iacr.org/2020/993
2. Bilgin, B., Gierlichs, B., Nikova, S., Nikov, V., Rijmen, V.: Higher-order threshold implementations. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology – ASIACRYPT 2014, Part II. Lecture Notes in Computer Science, vol. 8874, pp. 326–343. Springer, Heidelberg, Germany, Kaoshiung, Taiwan, R.O.C. (Dec 7–11, 2014). https://doi.org/10.1007/978-3-662-45608-8˙18

3. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4727, pp. 450–466. Springer (2007). https://doi.org/10.1007/978-3-540-74735-2_31, https://doi.org/10.1007/978-3-540-74735-2_31
4. Braeken, A., Nikov, V., Nikova, S., Preneel, B.: On boolean functions with generalized cryptographic properties. In: Canteaut, A., Viswanathan, K. (eds.) Progress in Cryptology - INDOCRYPT 2004, 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3348, pp. 120–135. Springer (2004). https://doi.org/10.1007/978-3-540-30556-9_11, https://doi.org/10.1007/978-3-540-30556-9_11
5. Cassiers, G., Grégoire, B., Levi, I., Standaert, F.: Hardware private circuits: From trivial composition to full verification. IACR Cryptol. ePrint Arch. **2020**, 185 (2020), https://eprint.iacr.org/2020/185
6. Chari, S., Jutla, C.S., Rao, J.R., Rohatgi, P.: Towards sound approaches to counteract power-analysis attacks. In: Wiener, M.J. (ed.) Advances in Cryptology – CRYPTO'99. Lecture Notes in Computer Science, vol. 1666, pp. 398–412. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 15–19, 1999). https://doi.org/10.1007/3-540-48405-1˙26
7. Cho, J.Y.: Linear cryptanalysis of reduced-round PRESENT. In: Pieprzyk, J. (ed.) Topics in Cryptology – CT-RSA 2010. Lecture Notes in Computer Science, vol. 5985, pp. 302–317. Springer, Heidelberg, Germany, San Francisco, CA, USA (Mar 1–5, 2010). https://doi.org/10.1007/978-3-642-11925-5˙21
8. Dhooghe, S., Nikova, S., Rijmen, V.: Threshold implementations in the robust probing model. In: Bilgin, B., Petkova-Nikova, S., Rijmen, V. (eds.) Proceedings of ACM Workshop on Theory of Implementation Security Workshop, TIS@CCS 2019, London, UK, November 11, 2019. pp. 30–37. ACM (2019). https://doi.org/10.1145/3338467.3358949
9. Faust, S., Grosso, V., Pozo, S.M.D., Paglialonga, C., Standaert, F.X.: Composable masking schemes in the presence of physical defaults & the robust probing model. IACR Transactions on Cryptographic Hardware and Embedded Systems **2018**(3), 89–120 (2018). https://doi.org/10.13154/tches.v2018.i3.89-120, https://tches.iacr.org/index.php/TCHES/article/view/7270
10. Goubin, L., Patarin, J.: DES and differential power analysis (the "duplication" method). In: Koç, Çetin Kaya., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems – CHES'99. Lecture Notes in Computer Science, vol. 1717, pp. 158–172. Springer, Heidelberg, Germany, Worcester, Massachusetts, USA (Aug 12–13, 1999). https://doi.org/10.1007/3-540-48059-5˙15
11. Hirt, M., Maurer, U.M.: Complete characterization of adversaries tolerable in secure multi-party computation (extended abstract). In: Burns, J.E., Attiya, H. (eds.) 16th ACM Symposium Annual on Principles of Distributed Computing. pp. 25–34. Association for Computing Machinery, Santa Barbara, CA, USA (Aug 21–24, 1997). https://doi.org/10.1145/259380.259412
12. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) Advances in Cryptology – CRYPTO 2003. Lecture Notes in Computer Science, vol. 2729, pp. 463–481. Springer, Heidelberg, Germany,

Santa Barbara, CA, USA (Aug 17–21, 2003). https://doi.org/10.1007/978-3-540-45146-4˙27

13. Kutzner, S., Nguyen, P.H., Poschmann, A., Wang, H.: On 3-share threshold implementations for 4-bit S-boxes. In: Prouff, E. (ed.) COSADE 2013: 4th International Workshop on Constructive Side-Channel Analysis and Secure Design. Lecture Notes in Computer Science, vol. 7864, pp. 99–113. Springer, Heidelberg, Germany, Paris, France (Mar 6–8, 2013). https://doi.org/10.1007/978-3-642-40026-1˙7
14. Moos, T., Moradi, A., Schneider, T., Standaert, F.: Glitch-resistant masking revisited or why proofs in the robust probing model are needed. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(2), 256–292 (2019). https://doi.org/10.13154/tches.v2019.i2.256-292, https://doi.org/10.13154/tches.v2019.i2.256-292
15. Nikova, S., Rechberger, C., Rijmen, V.: Threshold implementations against side-channel attacks and glitches. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 06: 8th International Conference on Information and Communication Security. Lecture Notes in Computer Science, vol. 4307, pp. 529–545. Springer, Heidelberg, Germany, Raleigh, NC, USA (Dec 4–7, 2006)
16. Nikova, S., Rijmen, V., Schläffer, M.: Secure hardware implementation of nonlinear functions in the presence of glitches. J. Cryptol. **24**(2), 292–321 (2011). https://doi.org/10.1007/s00145-010-9085-7, https://doi.org/10.1007/s00145-010-9085-7
17. Reparaz, O.: A note on the security of higher-order threshold implementations. Cryptology ePrint Archive, Report 2015/001 (2015), http://eprint.iacr.org/2015/001
18. Reparaz, O., Bilgin, B., Nikova, S., Gierlichs, B., Verbauwhede, I.: Consolidating masking schemes. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part I. Lecture Notes in Computer Science, vol. 9215, pp. 764–783. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2015). https://doi.org/10.1007/978-3-662-47989-6˙37
19. Siegenthaler, T.: Correlation-immunity of nonlinear combining functions for cryptographic applications. IEEE Trans. Inf. Theory **30**(5), 776–780 (1984). https://doi.org/10.1109/TIT.1984.1056949, https://doi.org/10.1109/TIT.1984.1056949
20. Sugawara, T.: 3-share threshold implementation of AES s-box without fresh randomness. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2019**(1), 123–145 (2019). https://doi.org/10.13154/tches.v2019.i1.123-145, https://doi.org/10.13154/tches.v2019.i1.123-145

A Masking of the Present S-Box

This appendix gives a decomposition of the Present S-box and a seven-sharing of the cipher.

A.1 Decomposition

Let (x, y, z, w) denote the input nibble from most significant to least significant bit. Similarly, $(G_1, ..., G_4)$ denotes the output from most significant to least significant bit.

$$S(x,y,z,w) = B'(G(G(C'(x,y,z,w)+d))+e)$$

In the above, the nonlinear function $G(x,y,z,w)$ is given as

$$G_1 = x + yz + yw \qquad G_2 = w + xy \qquad G_3 = y \qquad G_4 = z + yw\,,$$

the linear transformations as

$$B' = \begin{bmatrix} 1\,0\,1\,0 \\ 0\,1\,0\,0 \\ 1\,0\,0\,0 \\ 1\,0\,1\,1 \end{bmatrix}, \; C' = \begin{bmatrix} 1\,1\,0\,0 \\ 0\,1\,1\,0 \\ 0\,0\,1\,0 \\ 0\,1\,0\,1 \end{bmatrix},$$

and the constants as

$$d = \begin{bmatrix} 0\,0\,0\,1 \end{bmatrix}, \; e = \begin{bmatrix} 0\,1\,0\,1 \end{bmatrix}.$$

A.2 Seven-Sharing of $G(x,y,z,w)$

For each share $i \in \{1,...,7\}$, the permutation $G(x,y,z,w)$ is shared as

$$\begin{aligned}
G_1^i &= x^i + y^i z^i + y^i z^{i+1} + y^{i+1} z^i + y^i z^{i+3} + y^{i+3} z^i + y^{i+1} z^{i+3} + y^{i+3} z^{i+1} \\
&\quad + y^i w^i + y^i w^{i+1} + y^{i+1} w^i + y^i w^{i+3} + y^{i+3} w^i + y^{i+1} w^{i+3} + y^{i+3} w^{i+1}\,, \\
G_2^i &= w^i + x^i y^i + x^i y^{i+1} + x^{i+1} y^i + x^i y^{i+3} + x^{i+3} y^i + x^{i+1} y^{i+3} + x^{i+3} y^{i+1}\,, \\
G_3^i &= y^i\,, \\
G_4^i &= z^i + y^i w^i + y^i w^{i+1} + y^{i+1} w^i + y^i w^{i+3} + y^{i+3} w^i + y^{i+1} w^{i+3} + y^{i+3} w^{i+1}\,,
\end{aligned}$$

where the convention is used that superscripts wrap around at seven.

For each $i \in \{1,...,7\}$ and given 21 random bits (r_1^i, r_2^i, r_3^i), the randomness layer $\bar{r}_1(x,y,z,w)$ is given by

$$\begin{aligned}
G_1^i &= x^i + r_1^i + r_1^{i+1}\,, \\
G_2^i &= y^i + r_2^i + r_2^{i+1}\,, \\
G_3^i &= z^i\,, \\
G_4^i &= w^i + r_3^i + r_3^{i+1}\,,
\end{aligned}$$

where the convention is used that superscripts wrap around at seven.

For each $i \in \{1,...,7\}$ and given 28 random bits $(r_1^i, r_2^i, r_3^i, r_4^i)$, the randomness layer $\bar{r}_2(x,y,z,w)$ is given by

$$\begin{aligned}
G_1^i &= x^i + r_1^i + r_1^{i+1}\,, \\
G_2^i &= y^i + r_2^i + r_2^{i+1}\,, \\
G_3^i &= z^i + r_3^i + r_3^{i+1}\,, \\
G_4^i &= w^i + r_4^i + r_4^{i+1}\,,
\end{aligned}$$

where the convention is used that superscripts wrap around at seven.