Combined Threshold Implementation

Jakob Feldtkeller[1], Jan Richter-Brockmann[1], Pascal Sasdrich[1] and Tim Güneysu[1,2]

[1] Ruhr University Bochum, Horst Görtz Institute for IT Security, Bochum, Germany
[2] DFKI, Bremen, Germany
firstname.lastname@rub.de

Abstract. Physical security is an important aspect of devices for which an adversary can manipulate the physical execution environment. Recently, more and more attention has been directed towards a security model that combines the capabilities of passive and active physical attacks, i.e., an adversary that performs fault-injection and side-channel analysis at the same time. Implementing countermeasures against such a powerful adversary is not only costly but also requires the skillful combination of masking and redundancy to counteract all reciprocal effects.
In this work, we propose a new methodology to generate combined-secure circuits. We show how to transform Threshold Implementation (TI)-like constructions to resist any adversary with the capability to tamper with internal gates and probe internal wires. For the resulting protection scheme, we can prove the combined security in a well-established theoretical security model.
Since the transformation preserves the advantages of TI-like structures, the resulting circuits prove to be more efficient in the number of required bits of randomness (up to 100%), the latency in clock cycles (up to 40%), and even the area for pipelined designs (up to 40%) than the state of the art for an adversary restricted to manipulating a single gate and probing a single wire.

Keywords: Physical Security · Hardware Security · Threshold Implementation · Consolidating Masking Schemes · Side-Channel Analysis · Fault-Injection Analysis · Combined Analysis

1 Introduction

Today, cryptographic schemes are widely considered secure in a theoretic black-box model, where the adversary has access to a set of (chosen) inputs and outputs. Ultimately, cryptographic algorithms need to be implemented and executed on real-world chips in a real-world environment, which often undermines the clean and simple assumptions of the black-box model. Instead, an adversary can often passively observe or actively manipulate the physical execution environment to gain some knowledge about the hidden internals of the computation.

Observing the physical execution characteristics is considered as passive Side-Channel Analysis (SCA). It has been shown that physical characteristics, such as timing behavior [Koc96], instantaneous power consumption [KJJ99], or electromagnetic emanations [GMO01], can be used by an adversary to break otherwise secure cryptographic implementations. The underlying principle of such an attack is the correlation between the (secret) internal state of the chip and the observed physical characteristics. To account for the existence of SCA, the black box model was extended by some limited access to internal state variables [ISW03, DDF14, BCP+20]. In contrast, *manipulating* the physical execution environment is considered as active Fault Injection Analysis (FIA). Again, it has been shown

that by manipulating the system clock [DEG+18], the voltage supply [ZDCT13], the electromagnetic field [DDRT12, DLM19], or through the use of focused laser beams [SA02], the execution of cryptographic algorithms can be disturbed in such a way that otherwise secure algorithms are broken. Here, the underlying problem is the data-dependent propagation of faults towards the output. To account for FIA, the black-box model has been extended for the capability of targeted manipulation of state variables [IPSW06, RBSG22]. While both SCA and FIA have a long-standing tradition in the research community the Combined Analysis (CA) of both attack vectors gained attraction only recently. However, first practical attacks are emerging [AVFM07, CFGR10, RLK11, SJB+18, SBJ+21, SRJB23] motivating the introduction of theoretical models, that combine both the observation and manipulation of intermediates [DN20b, RBFSG22].

However, applying countermeasures for CA is expensive and error-prone. To protect against SCA, *masking* [CJRR99] has emerged as a promising and sound solution. The core idea of masking is to split the computation into parts that are individually independent of the operated data by leveraging the principle of secret sharing [Sha79]. The complexity of masking is quadratic in the number of tolerable leaked intermediate values and, in hardware, requires the use of additional registers to counteract the effect of physical glitches[1]. This significantly increases the area consumption and affects the latency (in clock cycles) of the circuit. Additionally, masking often requires a steady stream of fresh and high-quality randomness, which needs to be generated alongside the cryptographic computation. To counteract FIA, some sort of *redundancy* is required, either in time, space, or information. For combined security, this needs to be implemented on top of masking, e.g., replicating the already expensive masked circuit multiple times. Not enough, early research in countermeasures against combined attacks has shown that the mere combination of masking with redundancy is insufficient to counteract all reciprocal effects [RLK11, APZ21, SBJ+21, RBFSG22, FGM+23, SRJB23]. Hence, additional resources (in randomness, area, and/or latency) have to be applied to ensure security in the context of CA.

Since a large portion of the complexity of establishing combined security stems from the complexity in SCA security, we can take inspiration from recent results in the SCA research community. Specifically, Threshold Implementation (TI) [NRR06, BGN+14] has emerged as a masking scheme particularly suitable for optimized SCA-protected circuits that particularly enables optimizations in terms of latency and randomness consumption. At its core, TIs create component functions that are *non-complete*, i.e., independent of at least one secret share, and in combination provide a uniform-random output. Since TI circuits are hard to generalize for arbitrary security orders (the number of tolerable leaked intermediates), it is well-established to start with a low-level security order and subsequently increase the attack complexity while applying the lessons-learned [WM18, SM21b, BDRS21, DSM22].

Contribution. In this work, we provide a methodology to generate combined-secure hardware implementations optimized for latency and randomness consumption based on TI constructions. Towards this goal, we have a twofold contribution:

1. We start by providing a generic, security-order independent method to turn every SCA-secure circuit generated according to Consolidating Masking Schemes (CMS) [RBN+15] and withstanding d probes to a combined secure variant, able to withstand any adversary with the capability to place d probes together with k faults (Section 3). We chose CMS as a starting point since it provides a more general approach to masking than TI. For the resulting Combined Consolidating Masking Schemes (CCMS) we formally prove the claimed combined security. We then continue by showing that the same transformation also applies to related masking schemes,

[1] Glitches are transient and temporarily incorrect values carried by intermediate wires caused by timing differences in the computation paths.

Table 1. Notations used throughout this work.

	Notation	Description
SCA	d	Security order of a masking scheme (countermeasure).
	s	Number of shares used by a masking scheme.
	s_i	Number of input shares used by a masking scheme.
	s_o	Number of output shares used by a masking scheme.
FIA	k	Security order of redundancy scheme (countermeasure).
	n	Number of replications used by a redundancy scheme.
Misc	C, G	Represents a digital logic circuit or gadget, respectively.
	t	Degree of a function.
	$\overline{x}$	A vector x with one component x_i missing.

namely TI [NRR06] and Nullifying Fresh Randomness (NFR) [SM21a] (Section 4). This shows that our transformation applies to a wide range of SCA-secure circuits, enabling different trade-offs between area consumption, latency, and randomness requirements.

2. We provide an extensive evaluation of the performance characteristics of the resulting combined-secure circuits for $d = 1$ and $k = 1$ (Section 6). Here, we provide implementations of ten equivalence classes for 4-bit S-boxes, allowing the cost estimation for all possible 4-bit S-boxes. Further, we provide different implementations for the AES S-box and compare the results with existing schemes from the literature. In particular, our schemes not only allow the optimization for latency and randomness requirements but also for area consumption of pipelined designs. Finally, we provide performance figures for the implementation of a full AES round.

2 Preliminaries

In the following section, we provide self-contained background information to facilitate understanding of the contributions of this paper. The notations used throughout this work are given in Table 1. In general, we use an upper-case calligraphic font for sets (e.g., $\mathcal{S}$) and a sans-serif font for functions (e.g., f). Further, we indicate the share index with subscripts and the replication index with superscripts.

2.1 Circuit Model

In this work, we model a circuit C as a *directed acyclic graph* $\mathcal{C} = \{\mathcal{V}, \mathcal{E}\}$, where vertices $v \in \mathcal{V}$ represent logical gates and edges $e \in \mathcal{E}$ represent wires connecting individual gates and carrying a binary value from the field $\mathbb{F}_2$. Without loss of generality, we restrict the combinatorial gates to the set $\mathcal{G}_\mathsf{c} = \{\mathsf{inv}, \mathsf{and}, \mathsf{xor}, \mathsf{maj}\}$, where maj is a correction function that gets $2k+1$ inputs and outputs the majority function of the inputs. For memory gates, we define the set $\mathcal{G}_\mathsf{m} = \{\mathsf{reg}\}$, where reg represents a register that is updated once per clock cycle. Further, we define a set of input and output gates $\mathcal{G}_\mathsf{io} = \{\mathsf{in}, \mathsf{out}\}$, where in gets no input but generates a value from $\mathbb{F}_2$ and out receives an input from $\mathbb{F}_2$ but generates no output, and a set of randomness-generating gates $\mathcal{G}_\mathsf{rand} = \{\mathsf{rand}\}$, where rand outputs a uniform random value from $\mathbb{F}_2$. We refer to and operations as multiplications and xor operations as additions in $\mathbb{F}_2$.

2.2 Side-Channel Security

Adversary Model and Security Definition. Throughout this work, we consider the Ishai-Sahai-Wagner (ISW) d-probing model [ISW03], where an adversary $\mathcal{A}_p$ can select up to d wires of a circuit, of which, on invocation, the exact values are leaked to $\mathcal{A}_p$. As we

strive to model the behavior of hardware circuits, we use the glitch-extended d-probing model [FGP+18], where $\mathcal{A}_p$ not only gets access to the values carried by the probed wires but to the last synchronization points, i.e., the last registers or input gates. Then, we define a circuit C to be *d-probing secure* iff the view of $\mathcal{A}_p$, defined by the probes, can be simulated without access to any secret [ISW03]. Here, simulation requires the recreation of the same output distribution with a restricted set of information. If such a simulation is possible then it is ensured that the view of $\mathcal{A}_p$ is always independent of any secret. Security in the d-probing model does not imply security against horizontal attacks [BCPZ16], which exploit the sequential processing of secret shares. While such attacks may also be possible in parallel hardware implementations, it was shown that they become significantly more difficult, even assuming a noiseless setting [BDF+17]. Taking the high algorithmic noise from parallel computation into account, we think this is an acceptable limitation.

Masking. A popular and well-researched way to achieve probing security is *Boolean masking* [CJRR99]. The idea is to split a value $x \in \mathbb{F}_2$ into a vector $\langle x_0, \ldots, x_{s-1} \rangle \in \mathbb{F}_2^s$, such that $x = \bigoplus_{i=0}^{s-1} x_i$ and each subset $\hat{\mathcal{X}} \subset \{x_i \mid i \in [s-1]\}$ with $|\hat{\mathcal{X}}| < s$ is statistically independent of x. We refer to a component x_i as a *share* of x and to all shares with index i as a *share domain*. A valid sharing can be easily generated by choosing $s-1$ shares uniformly random from $\mathbb{F}_2$ and the last share such that the sum adds up to x. To compute over shared values, the circuit is likewise transformed to a *shared circuit*, operating on shares instead of the real values. For such a shared circuit, we assume the initial sharing and final unsharing operations to be not part of the circuit itself [ISW03, AIS18], i.e., no probes can be placed in those parts.

Probe Propagation. The specific information the adversary $\mathcal{A}_p$ can learn from a set of probes depends on the structure of the circuit leading up to the probes and can be determined by the concept of *probe propagation* [BBP+16]. Specifically, a probe propagates from wire w_1 to a wire w_0 if the value carried by w_0 is required for the simulation of wire w_1. Hence, probes always propagate from their placed location towards the inputs. Here, the addition of a unique random value r to w_0 (mask refreshing), i.e., $w_1 = w_0 + r$, always stops the propagation of probes. Please note, that while the concept of probe propagation has some similarities with the glitch-extension of probes those are fundamentally different concepts. In particular, glitch-extension is structure agnostic until the next register stage or input is reached, while probe propagation is not stopped by registers.

2.3 Fault Security

Adversary Model and Security Definition. We model a faulting adversary $\mathcal{A}_f$ according to Richter-Brockmann et *al.* [RBSG22]. Here, $\mathcal{A}_f$ can freely choose up to k gates and manipulate them according to a transformation $\mathcal{T}$. Essentially, $\mathcal{T}$ provides for each gate type a set of gate types to which the gate can be transformed. Popular choices for $\mathcal{T}$ are *set* or *reset* (replacing a gate by a constant one or zero, respectively) or *flip* (replacing a gate by its inverse). The circuit is then executed with the manipulated gates and the correctness of the result is leaked to $\mathcal{A}_f$. The correctness is defined by comparison to a *golden circuit*, which is the circuit without manipulation by $\mathcal{A}_f$. Then, we define a circuit C to be k-fault secure if all faults can be corrected at the output [DN20b, RBFSG22], i.e., there exists a circuit G^C such that the concatenation $G^C \circ C$ always yields the output of the golden circuit. Note, we focus on correction-based countermeasures since for the combined setting it is unclear how to securely abort a computation in hardware when a fault is detected (see below).

Replication. Protection against fault attacks always uses some form of redundancy, either in space, time, or information. A simple form of redundancy is replication, where a circuit is replicated multiple times in space and the different instances are compared to detect or correct occurring faults. In general, if an adversary can inject faults into k gates, $2k + 1$ replications are required for correction, e.g., via a majority function maj. Similar to shared circuits, we consider the initial replication of values and the final correction not part of the verified circuit [RBFSG22], i.e., no faults can be injected in those parts. The protected final correction is only required to meet the formal security notion, which does not allow any difference to the golden circuit at the output. In general, faulting the final correction of an implementation is equivalent to faulting a known output, which cannot provide any additional knowledge to the adversary.

Fault Propagation. Once injected, a fault propagates through the circuit causing differences to the golden circuit [AMR+20]. More precise, we say a fault propagates from a wire w_0 to a wire w_1 if a difference between the manipulated circuit and the golden circuit in w_0 causes such a difference in w_1. Hence, a fault propagates always from the fault injection location towards the outputs. Most importantly, a single fault propagates always through an xor gate while it propagates through an and gate only if the other input is equal to one. In contrast, a maj gate stops the incoming fault from propagating further (until a certain threshold).

2.4 Combined Security

Adversary Model and Security Definition. We consider a combined adversary $\mathcal{A}_c$ as the combination of $\mathcal{A}_p$ and $\mathcal{A}_f$ [DN20b, RBFSG22], i.e., an adversary with the ability to select up to d wires for probing and up to k gates for faulting. Then, on execution, the glitch-extended probes and the correctness of the circuit are leaked to $\mathcal{A}_c$, however, with a slightly modified golden circuit definition. For CA, we define the golden circuit to already be affected by faults injected into gates $g \in \mathcal{G}_{\mathsf{rand}}$ [RBFSG22]. The reason is that the correctness of the output does not depend on the value of the randomness. We then define combined security as follows:

Definition 1 (Combined Security [RBFSG22])**.** A circuit C is (d, k)-combined secure iff for any set of up to k faults injected into gates of C and any set of up to d probes placed on wires of C the following holds:

Correctness: There exists a circuit G^C such that the concatenation $G^C \circ C$ always yields an output equal to the golden circuit of C.

Privacy: The probes in the faulted circuit can be simulated without access to any secret-dependent inputs of C.

Masking & Replication. An intuitive approach to aspire combined security is the combination of masking and replications, i.e., creating a shared circuit and replicating it $2k + 1$ times. Unfortunately, the resulting circuit is not in general combined secure since faults can impact the probing security of a circuit [DN20b, RBFSG22, FRSG22]. For example, this can be true when faults remove the entropy of a mask refreshing or, more subtle, data-dependent fault propagation causes a recombination of shares, e.g., because a fault only propagates through an and gate when the other input is set to one and does not propagate otherwise [FGM+23]. Hence, more sophisticated methods are required that implicitly consider those reciprocal effects. Still, masking in combination with replication is a good starting point, however, requiring additional intermediate corrections and mask refreshing. In this paper, we do this by introducing a strong *isolation* layer in combination with non-completeness. Here, we focus on the use of fault correction instead of fault

detection. The reason is twofold: First, fault detection does not automatically prevent the computation with faulty values which complicates the security analysis or requires significant additional logic to ensure an immediate flushing of secret-dependent data on fault detection. Second, there exists SCA leakage from the detection flag that is not trivial to prevent. In general, the replication required for correction will amplify the leakage of internal values. Since computations not targeted by the attack will be replicated as well the noise is increased similarly. Hence, in general, the security reduction from the noisy-leakage model by Duc et *al.* [DDF14] should still hold, however, possibly requires more system noise. Note, that fault security mandates some kind of redundancy.

Probe & Fault Propagation. As already indicated, fault propagation can affect the propagation of probes. In particular, a fault can have two interesting effects on probe propagation [FGM+23]: (i) The fault can remove some refreshing and allow a probe to propagate further through the circuit. (ii) A set of probes can observe data-dependent fault propagation, which can leak additional shares of a secret. However, for this leakage to occur, the adversary has to observe whether the fault indeed propagates, which usually means the requirement to observe at least two replications (one faulted and one correct). Below we provide a simple and intuitive lemma that very conservatively captures the interaction of probes and faults in a circuit. In short, we claim that the worst-case leakage in that case is that all inputs are leaked.

Lemma 1. *Let C be a circuit implementing a function $\mathsf{f} : \mathbb{F}_2^m \to \mathbb{F}_2$, with inputs $x_0, \ldots x_{m-1}$. Further, let there be d probes placed on wires in C and k faults injected in gates of C. Then, the worst-case leakage is equivalent to probes placed on all inputs $x_0, \ldots, x_{m-1}$.*

The proof of this statement is straightforward: There cannot be more leakage than there is information in the circuit. However, please note that Lemma 1 places new probes on all the inputs, which in turn can propagate further through the circuit when only a subcircuit is considered.

3 Combined Consolidating Masking Schemes

Over the years, the side-channel community has proposed a wide range of different masking schemes to protect against a probing adversary [ISW03, Tri03, NRR06, CS20]. To structure the literature and highlight similarities between TI [NRR06] and the ISW masking scheme [ISW03], Reparaz et *al.* [RBN+15] introduced Consolidating Masking Schemes (CMS) as a unified method. The core idea is that masking schemes can be viewed as a combination of layers, each with specific properties, that work together to provide security in the context of SCA. In the following section, we first provide the details for CMS and then, for arbitrary security order, show how to transform an arbitrary CMS design to the context of CA.

3.1 The CMS Approach

In general, CMS [RBN+15] divides a masking construction into four layers, each with a specific purpose. In the following description, we name the third layer *isolation* instead of *refresh* layer to prepare for the CA extension (cf. Section 3.2). The general scheme is depicted in Figure 1a for an example circuit and consists of the following four layers:

1. **Non-linear Layer** N**.** The first layer is responsible for the computation of all non-linear terms of the target function. In our circuit model, this means the placement of all required and gates. The computation of a non-linear function in a shared manner

requires the computation of products of different share domains. For example, the computation of a simple multiplication in $\mathbb{F}_2$: $f(a,b) = ab$ with two shares requires the computation of all four cross products, i.e., $a_0b_0, a_0b_1, a_1b_0, a_1b_1$. This has two implications: (i) The non-linear layer increases the number of shares, i.e., the number of output shares is higher than the number of input shares. In particular, the number of output shares can be computed as $s_o = s_i^t$ for each non-linear term, where t is the order of the term and s_i the number of input shares. (ii) A placed probe does leak shares with different indices for different input variables. However, the leakage caused by a probe in this layer is still restricted to one share domain per input variable for each probed non-linear term.

2. **Linear Layer L.** The second layer introduces the linear computation of the target function (i.e., the xor between different inputs and non-linear terms from N) and makes a first reduction in the number of shares (by an xor between different shares). The first part ensures that the output is already a valid and correct sharing of the desired function f, in the sense of Definition 2.

   Definition 2 (Correctness [NRR06])**.** The sum of the output shares is equivalent to the output of the unshared function, i.e., $\sum_i^{s_o} f_i(x_0, \ldots, x_{s_i-1}) = f(x)$.

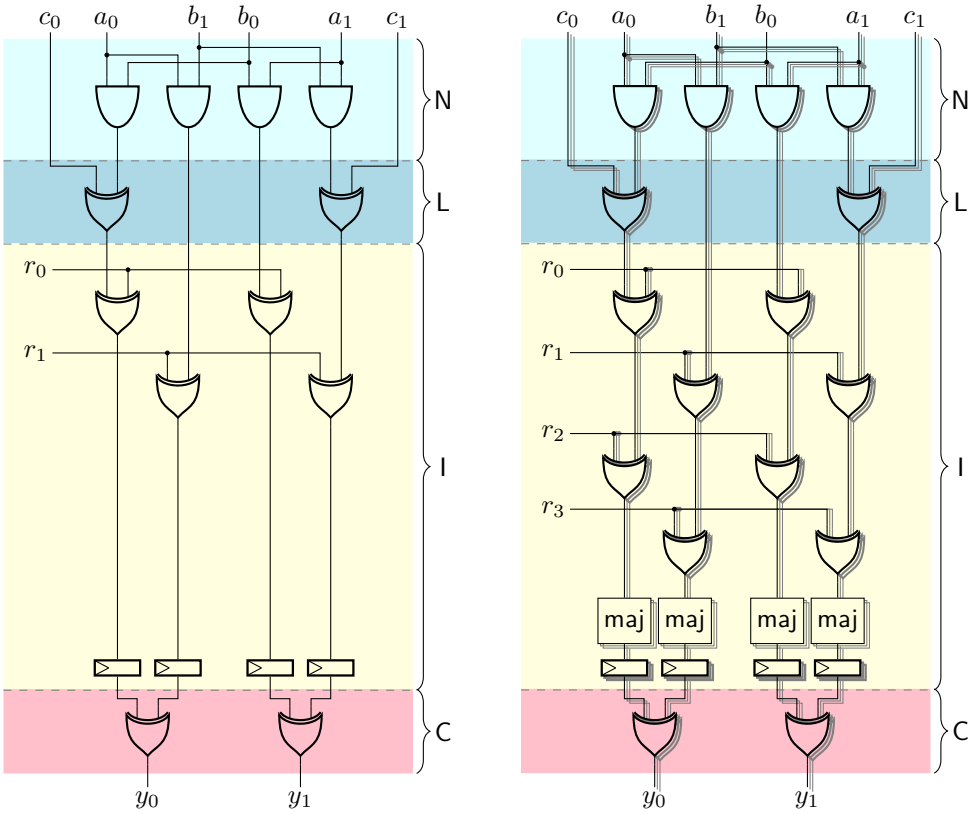
   The second part reduces the number of shares after the blow-up of the previous non-linear layer. However, the reduction of shares cannot be done without restriction. For probing security, the combination of N and L has to be non-complete (cf. Definition 3), i.e., any set of up to d probes placed on the output of L (and extended by glitches) has to be independent of at least one input share.

   Definition 3 (Non-Completeness [BGN+14])**.** Any combination of up to d component functions is independent of at least one share of each input.

   In general, the non-completeness property requires the use of $td+1$ shares [BGN+14]. In the case where all inputs are independent of each other, this can be reduced to $d+1$ shares which is made explicit in a relaxed non-completeness property [RBN+15]. We emphasize this, by the introduction of a separate definition (Definition 4). In particular, the independence of inputs is required to ensure that no secret is recombined by the combination of different shares of different inputs.

   Definition 4 (Relaxed Non-Completeness)**.** Any combination of up to d component functions is independent of at least one share index.

3. **Isolation Layer I.** The third layer was originally termed *refresh* layer by Reparaz et *al.* because it is essentially a mask refreshing with fresh random values. For this work, we rename this layer to *isolation* layer to generalize the security-providing nature of this layer in preparation for the extension to CA. To further reduce the number of shares beyond the restrictions dictated by non-completeness, it is essential to isolate the previous two layers from the rest of the circuit with respect to probe propagation and glitch extension. As discussed in Section 2.2, probes propagate until they reach an xor with a fresh and unique random value and glitches are extended until the previous register stage. Hence, the isolation layer consists of a mask refreshing with fresh randomness and a register stage at the output. Performing mask-refreshing has to be ineffective for the correctness of the sharing but ensure security in a d-order attack. This is a well-researched problem and several solutions exist [BBD+16, CGLS21].
4. **Compression Layer C.** The final layer consists of a net of xor gates to further reduce the number of shares. In particular, it is usually desirable to have the same



(a) Structure of CMS for $y = ab + c$. **(b)** Structure of CCMS for $y = ab + c$.

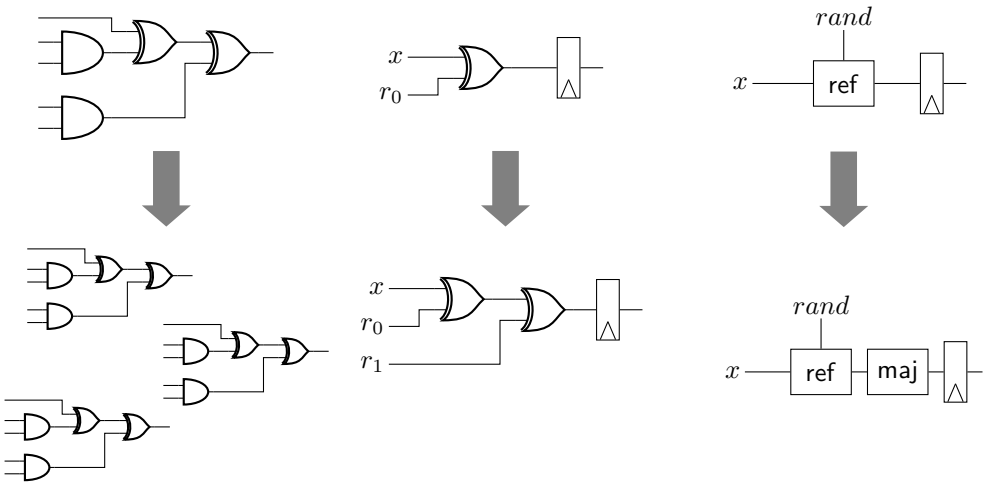
Figure 1. Differences between CMS and CCMS for $d = 1$ and $k = 1$.

amount of input and output shares. While this layer provides no particular security service, it has to be constructed in such a way that the mask refreshing of the previous layer remains intact.

The security of the CMS masking is guaranteed by the combination of the non-completeness within a single component function and the isolation layer. More precisely, the non-completeness ensures that probes within the layers N, L and I are independent of any secret input (at least one share remains unobserved), while the isolation layer stops any probe outside of those three layers from interfering with the internal probes. Therefore, the isolation layer not only allows for further reduction in the number of shares in C but also the composition of several circuits designed with the CMS principle.

Originally, CMS introduces a strict separation between the non-linear and linear layers. While any function can be represented with this separation (e.g., with the Algebraic Normal Form (ANF)) this separation is not necessary for security as long as non-completeness is not violated. Indeed, it is often more beneficial for performance to mix the computation of linear and non-linear terms.

We note, that Moos et *al.* [MMSS19] showed a flaw in a CMS-multiplication circuit for $d \geq 3$ resulting from the combination of the refresh method and the specific compression. In particular, the refreshing and compression are structured in such a way, that some of the random bits in the refreshing are canceled out and the remaining bits of randomness are not sufficient to provide the aimed security order. Hence, this is an issue with the specific multiplication circuit and not with the general CMS methodology.



(a) Step 1: Circuit Replication. **(b)** Step 2: Refresh Replication. **(c)** Step 3: Correction.

Figure 2. The different steps for the transformation from CMS to CCMS for $d = 1, k = 1$.

3.2 From CMS to Combined Security

3.2.1 Transformation

We now provide a simple transformation that makes every probing secure CMS-based design secure under a combined adversary. In particular, our transformation T^{CA}_{CMS} consists of three steps, which are depicted in Figure 2 and described in further detail below.

1. **Circuit Replication.** The first step is to provide the circuit with stand-alone fault-injection protection. For this, we will use a simple replication of the circuit, such that a majority function maj at the end can correct all faults. Hence, to provide k-order security the circuit is replicated $2k + 1$ times.
2. **Refresh Replication.** The second step is to ensure that a faulting adversary cannot mess with the mask refreshing in the isolation layer I. For this, we ensure that each share is refreshed with at least $k + 1$ random bits in an iterative way. For instance, for $k = 1$ we replace every computation $x' = x + r_i$ with $x' = (x + r_{i_1}) + r_{i_2}$. The iterative approach (given by the parentheses) is necessary to ensure that a single fault cannot remove all random bits at once. For this, special care has to be taken during synthesis to ensure that no optimization results in the computation $x' = x+(r_{i_1}+r_{i_2})$ where the adversary can remove the mask refreshing with a single fault. This can usually be achieved by an appropriate macro for the synthesis. Actually, for some refresh methods, the replication of the refresh is not required. This is the case if the non-replicated refreshing already has sufficient random bits per share, i.e., at least $k + 1$ random bits.
3. **Correction.** The third and final step is to enhance the isolation layer with the ability to stop the propagation of faults even if the isolation layer is targeted by faults itself. For this, we require a *fault-isolating* correction, which we define as a correction module where the output is always independent of incoming fault propagation, even in the presence of faults injected into the correction module itself. For $k = 1$, this can be done by a simple majority vote (as discussed in Section 2.3). As always, the corrections must be implemented such that each gate affects at most one output (independence property [AMR+20]). To leave the isolation properties for probe propagation intact the corrections are placed between refreshing and registers.

Unfortunately, and as already observed by Feldtkeller et *al.* [FGM+23], the naive implementation of a majority vote is not fault-isolating for $k \geq 2$. In particular, it is possible to inject a fault into the correction, such that an incoming faulty value propagates to the output of the correction. A naive but costly way to implement a k-order fault-isolating correction is via k consecutive corrections. This ensures that at least one correction module is unaffected by a fault and, hence, removes the dependency of any incoming fault propagation. We leave the research into more efficient fault-isolating corrections for future research.

The result of T^{CA}_{CMS} applied to the circuit from Figure 1a can be seen in Figure 1b. Most of the transformation is concentrated in the isolation layer. This concentration makes the security discussion much easier to handle.

3.2.2 Soundness

In the following, we will discuss the soundness and security guarantees given by T^{CA}_{CMS} at arbitrary security order. For this, we start with an intuitive and high-level description of the security argument before delving into the formal proof for combined security.

High-level Intuition. The core security is provided by the combination of non-complete component functions and the isolation layer. Specifically, under isolation, the security provided by non-completeness remains intact even in the presence of injected faults. That is, the most powerful probes in the component functions (i.e., within layers N, L and I) are those placed directly in front of a register, as those probes provide the adversary with all inputs of that component function (via glitch extension). Injection of a fault in the same component function therefore cannot provide more information and a fault in a different component function does not reach the probe due to functional isolation.

Hence, the security argument comes down to the isolation properties of I given a combined adversary $\mathcal{A}_c$. The isolation for probes, both propagation and glitch extension, is already given by the underlying CMS construction. Isolation of faults is added by the fault-isolating correction modules and their fault-independent implementation (where each gate of the correction impacts at most one output). Hence, any fault is either corrected (if injected before the correction) or affects at most one output of I. This ensures stand-alone fault security since at most k replications can be affected with k faults and those can be corrected by a majority function at the output. For combined security, we observe that a fault has to be injected within or after the correction module of the previous isolation layer I to have an impact on a probe. The propagation of such a fault is in the worst case dependent on the input of the correction (with fault-isolation of the correction). However, the input of the correction is the same value that the adversary would observe anyway at the output of I. This is the reason why the correction must be placed between the refresh and the registers. Otherwise, the fault propagation could transport leakage across the mask refreshing.

Formal Argument. We now provide a more formal argument for the claimed (d, k)-combined security after T^{CA}_{CMS}. The formal security guarantees are captured in the following theorem.

Theorem 1. *Let C be a d-order probing-secure circuit, designed according to the CMS paradigm. Then, $C' = T^{CA}_{CMS}(C)$ is (d, k)-order combined secure.*

Proof. We first prove privacy and then correctness.

Privacy: We first note that the replication and insertion of correction modules do not affect probing security. The reason is, that a correction will only propagate a probe to the same values at all replications as it would to anyway in the original circuit.

Let $\texttt{Sim}$ be the probing-secure simulator for C. We construct a simulator $\texttt{Sim}'$ for C' by manipulating the simulation of $\texttt{Sim}$ according to the injected faults.

This is always possible and results in the correct simulation for the following reasons: Without loss of generality, we assume a set of probes placed directly in front of registers in I' (other probes in the same component functions are less powerful, due to glitch-extension). Probes on the output of the circuit can be seen as a sub-case, i.e., a component function only consisting of the previous compression layer.

We only need to consider cases, where the fault propagation reaches the glitch-extended and propagated probe, since otherwise it reduces to plain probing security. Hence, there are three different scenarios for fault locations. (i) Faults injected into refresh randomness. (ii) Faults injected in the same component functions as the probes, i.e., L, N, I', or in case of composition the previous C. (iii) The fault influences the input of the probed component functions in the case of composition, i.e., placed in I' previous to the component function (the probes are independent of other earlier faults because of the fault-isolating correction).

(i) First, faults injected into refresh randomness do not affect probing security due to the replication of the refreshing in I' (Step 2 of T^{CA}_{CMS}).

(ii) Let us consider the second case where the fault is placed in the same component functions as the probes. By the rules of glitch extension and because there is no register within the component function, the simulator $\texttt{Sim}$ gets access to all inputs of the probed component functions (in the case of composition this is the input of the previous C). With Lemma 1 this does also not change when faults are injected within the component functions. Hence, $\texttt{Sim}'$ can use the same simulation as $\texttt{Sim}$ and modify the value afterwards according to the faults injected.

(iii) Now, consider a faults injected in the previous I', more precisely in correction modules. Such faults are independent of any incoming fault propagation (due to the fault-isolating correction) and hence, can only leak information about the correct input to the correction module. However, the correct input of the correction module is equivalent to the output of the non-faulted correction module. Hence, again, $\texttt{Sim}'$ can use the same simulation as $\texttt{Sim}$ and modify the value afterwards according to the fault.

Correctness: The circuit C' is replicated $2k+1$ times and the only interaction between different replications is the correction modules, which are implemented separately for each replication (independence property). Hence, a fault will always affect at most one replication. Let G^C be a majority vote with $2k+1$ inputs. Then it holds that the concatenation $G^C \circ C$ outputs always a correct result, as at most k of the $2k+1$ replications are faulty. □

4 Related Constructions

After showing how to transform an arbitrary CMS construction into a combined secure circuit, we now focus on constructions that, like CMS, are based on non-completeness. In particular, the above proof does not specifically rely on the mask refreshing as long as the output of the isolation layer is a secure and valid Boolean sharing. In the following, we will show that the same transformation also holds for TI [NRR06] and NFR [SM21a] constructions, which use different mechanisms to ensure a valid sharing after the isolation layer. The application of the introduced transformation is expected since CMS is explicitly constructed as a generalization of TI, and NFR can be seen as a special case of TI. As a result, the security proofs essentially boil down to the proof given in Section 3.2.2.

4.1 Threshold Implementation.

Threshold Implementation is a first-order masking technique proposed by Nikova et *al.* [NRR06] and is one of the foundations for CMS. In particular, it also relies on the properties *Correctness* (Definition 2) and *Non-Completeness* (Definition 3) when constructing component functions. However, instead of using a mask refreshing with fresh randomness in the isolation layer, it introduces a third property *Uniformity* (or *Balancedness* – Definition 5), that requires the output sharing to have the same uniform distribution as the input sharing, making the observation independent of the computed function. Usually, TI tries to achieve uniformity by a clever arrangement of component functions. If that is not possible (or too costly), mask refreshing is always a way to get uniformity in the outputs. Similar to CMS, TI also requires an isolation layer with registers to stop glitch extension of probes. Later, TI was extended to higher-order masking [BGN+14], which was then shown insecure under certain compositions by Reparaz et *al.*, who argue that the refreshing layer is essential for higher-order security [RBN+15].

Definition 5 (Uniformity [NRR06])**.** A realization $z = F(x, y, \ldots)$ is uniform iff for all input distributions of the inputs $x, y, \ldots$, and for all input share distributions with $\mathsf{Pr}[\overline{x} = \overline{X}, \overline{y} = \overline{Y}, \ldots] = c\mathsf{Pr}[x = \sum_i X_i, y = \sum_i Y_i]$ it holds that $\mathsf{Pr}[\overline{z} = \overline{Z} \mid z = \sum_i Z_i]$ is constant.

Combined-Security Transformation. Constructing a combined secure circuit out of a first-order TI implementation requires the same transformation as CMS (cf. Section 3.2.1). However, as TI schemes (usually) perform no mask refreshing in the isolation layer, Step 2 of the transformation (Refresh Replication) can be skipped. Of course, this is quite beneficial as randomness generation is very costly. The formal security guarantees for Combined Threshold Implementation (CTI) are captured in the following lemma.

Lemma 2. *Let C be a 1^{st}-order probing-secure circuit, designed according to the TI paradigm. Then, $C' = T^{CA}_{CMS}(C)$ is $(1,1)$-order combined secure.*

The proof follows directly the line of the proof of Theorem 1. When carefully analyzing the proof of Theorem 1, we observe that the security argument does not rely on the mask refreshing at all. Indeed, the security is always reduced to the security of the underlying probing-secure scheme. More precisely, to the security when only observing the values carried by the registers (as the input to the correction is the same value as the value of the register). However, this is also true for a TI construction and, hence, the same proof applies to Lemma 2.

4.2 Nullifying Fresh Randomness.

The classical TI construction requires in general at least $td + 1$ shares to fulfill *non-completeness* and *uniformity*. However, it was shown that this is not always necessary and a secure masking scheme with only $d + 1$ shares can be constructed for $d = 1$ [SM21a]. In the following, we refer to this masking scheme as NFR. The core insight is that to apply the reduction of shares in C uniformity is indeed not required. Instead, it is sufficient that all values that are compressed to a single output share (output of layer I) are jointly statistically independent of the inputs. This allows the construction of probing secure circuits without mask refreshing and two shares. Hence, similar to TI the isolation layer only consists of a register stage to stop glitch extension of probes. However, in contrast to CMS and TI special care needs to be taken when composing different circuits constructed with NFR. In particular, it has to be assured that all values that are combined are jointly statistically independent of the inputs, which can be a challenging task and easily result in flawed designs [MM22].

Combined-Security Transformation. Again, we can apply the same transformation as for CMS to NFR to get a combined-secure circuit. As in the case of TI, the second step of the transformation (Refresh Replication) can be skipped, preserving the low randomness requirement of NFR. We state the formal security guarantees in Lemma 3, where the proof again follows directly the line of the previous proofs. In particular, now the observed values are jointly statistically independent of the inputs/secrets, regardless of the observation at the registers via glitch extension or at the input of the correction via conditional fault propagation.

Lemma 3. *Let C be a 1^{st}-order probing-secure circuit, designed according to the NFR paradigm. Then, $C' = T^{CA}_{CMS}(C)$ is $(1,1)$-order combined secure.*

5 Related Work

Early works showed that integrating countermeasures for both SCA and FIA can be effective in isolation, i.e., the design is protected against both attack types when conducted individually but not at the same time [SMG16, RG20, GPK+21, DOT23]. In the following, we provide an overview of security schemes for CA, which is still a rather new research direction.

MAC-based Schemes. The first schemes for combined security use fault detection via Message Authentication Codes (MACs) in combination with masking. As a pioneer, CAPA [RDB+18] splits the architecture into isolated *tiles*, where an adversary is allowed to probe and fault entire tiles, and uses methods from robust Multi-Party Computation (MPC) to let the tiles interact securely for a common computation. The use of methods from MPC should provide a solid foundation for the formal security argument as long as the sum of probed and faulted tiles does not exceed the probing-security order d, but suffers from high overhead. However, only recently an attack was found that exploits a deviation from the original MPC protocol, effectively breaking CAPA with one fault and one probe in a single tile [TNN24]. More specifically, a random fault is injected during the preprocessing phase such that the MAC is generated on the faulty value. When the faulty value is used in a multiplication the detection of the fault is dependent on the unmasked value of the other multiplication input. While it is possible to make the exploitation of this vulnerability more complex [TNN24], a provable secure fix is an open problem.

Later, M&M [DAN+19] proposed a more efficient combination of a MAC with masking by the use of infection, i.e., a fault detection mechanism that tries to render an injected fault useless for the adversary. Again, the security is claimed as long as the sum of injected faults and placed (glitch-extended) probes does not exceed the security order d. While the resulting scheme is indeed quite efficient it comes without a formal security argument for CA and, unfortunately, a zero-value attack was discovered that breaks the security of the proposed AES implementation [HMA+24]. We would like to highlight that the shown attack is not a combined attack but relies on fault injection only and is formally excluded from the adversary model in M&M (however, such attacks are always possible in a combined attack setting). The authors of the attack then propose a so-called *λ-detection* for the specific AES S-box implementation that introduces a more fine-grained detection scheme [HMA+24]. This additional detection scheme adds another detection path parallel to the data path of the encryption with a latency of seven cycles (after the six cycles of S-box computation). Since this is a significant delay it remains an open question whether a combined attack, that performs SCA in the time between fault injection and fault detection, is still possible.

All MAC-based methods have an additional security parameter m that determines the unforgeability of the MAC, where a lower m results in lower security (with $m \geq 1$).

Replication-based Schemes. Another line of research focuses on masking in combination with replication, as a fault-detection/correction mechanism. This is the line of research most similar to our work. For this, Dhooghe and Nikova [DN20b] presented composable gadgets for CA, i.e., a circuit for a specific function (e.g., and) with certain security and compositional properties. The proposed gadgets are based on the probing notions Non-Interference (NI) [BBD+15] and Strong Non-Interference (SNI) [BBD+16], which limit the amount of probe propagation through the circuit. The formal security analysis was conducted in the standard probing model (i.e., without glitch extension) and is therefore insufficient for hardware. Indeed, the first proposed gadget cannot be transferred to hardware, as it uses an abort that is hard to implement in hardware, and the second gadget was later shown insecure by Richter-Brockmann et *al.* [RBFSG22]. Another approach proposed by Dhooghe and Nikova [DN20a] is a gadget constructed with the concepts of MPC in mind (similar to CAPA [RDB+18]). Specifically, they also divide the shared multiplication into different *tiles*, one for each replication of a partial product, and place registers between each tile to stop glitch-extension of probes. Fault propagation is stopped by error detection just in front of each partial product computation (requiring only $k+1$ replications). Then, assuming an *ideal abort signal*, i.e., a signal that reaches all parts of the circuit without delay and which can delete all secret-dependent data, all computation is stopped when a fault is detected. Like with CAPA, an adversary is allowed to probe and fault entire tiles, as long as the sum of probed and faulted tiles is smaller or equal to d (a faulted tile is automatically also probed). The construction via tiles allows for easy handling of physical defaults such as glitches, transitions, and couplings. Also, they assign each Random Number Generator (RNG) to a specific tile, to explicitly define the security domain of the RNG. In all other works, including ours, the specifics of randomness generation are lost in the abstraction and only a single random bit can be faulted with one fault. However, it remains an open question how to implement the ideal abort signal in hardware. This is a non-trivial and (probably) costly challenge since all leaking intermediate values have to be erased. We can also analyze our approach using the tile-based model. For this, we assign each replication of a component function (with refreshing but without correction), each randomness-generating gate, and each majority vote to a different tile. Then, security holds as long as at most k tiles are faulted and at most d tiles are probed (with a similar argument as for Theorem 1). Later, Feldtkeller et *al.* [FRSG22] presented a different kind of composable gadgets for CA, this time based on the probing notion of Probe-Isolating Non-Interference (PINI) [CS20] which allows trivial composition (each combination of PINI gadgets is PINI again). Those gadgets were directly proposed in the glitch-extended probing model. However, again, the gadgets were later shown to be flawed, due to unconsidered side-channel leakage of fault propagation [FGM+23]. At the same time, a fix was proposed in the form of Combined Private Circuits (CPCs) [FGM+23], gadgets that support the same compositional Combined-Isolating Non-Interference (CINI) notion. When fixing the gadgets, the authors also explore the option of gadgets based on non-completeness in the compression stage. However, those gadgets turned out to be more costly and harder to generalize than the introduction of additional corrections. In contrast to that, we apply non-completeness not on a gadget but on a circuit level, which yields better results. All the defined compositional notions for CA come in two flavors: (i) where the sum of the injected faults and placed probes need to be smaller or equal to the probing order d, (ii) where the fault and probing order is independent, i.e., up to k faults and up to d probes can be placed by $\mathcal{A}_c$. While hardware gadgets for arbitrary security orders currently only exist for the first type, the $(1,1)$-$\mathsf{CPC}_1^{\mathsf{C}}$ gadget can be transformed into a $(1,1)$-$\widehat{\mathsf{CPC}_1}$ gadget of the second type, by replicating the randomness (similar to Step 2 in T_{CMS}^{CA} - cf. Section 3.2.1) [FGM+23].

Polynomial Masking. A third line of research is based on polynomial masking, also called Shamir's Secret Sharing [Sha79]. The core idea is, that a polynomial of degree $d+1$ can be efficiently recovered with $d+1$ points on the function of the polynomial, while for d points there are still an infinite amount of possible polynomials left. Based on this principle, Seker et *al.* [SFES18] introduce fault resistance to an MPC-based, polynomial masking scheme, by adapting the shared multiplication in such a way, that effective faults always can be detected. While the side-channel analysis is done in the standard probing model, glitches are considered at the underlying MPC scheme. The authors claim security for k faults and d probes with $2d+k+1$ shares. However, their analysis for combined security only considers two specific attack vectors and provides no formal guarantees. Indeed, it was later shown that their refresh algorithm does not hold the claimed side-channel security [BEF+23].

Only recently, this approach was improved by Berndt et *al.* [BEF+23], by providing gadgets that only require $d+k+1$ shares for (d,k)-combined security in the region probing model (i.e., $\mathcal{A}_c$ is allowed to place d standard probes *per* gadget) with adaptive faults (i.e., fault values can be selected after observation of previous probes). They achieve this by constructing gadgets that preserve their compositional properties for SCA even under the influence of k faults. While the authors conduct a thorough formal analysis, they do not consider glitch-extended probes and application to hardware remains an open question. Nevertheless, the result is an efficient software scheme with a record-breaking low number of instructions.

6 Evaluation

In the following, we provide insights into the performance characteristics of our methodology. In doing so, it is not our goal to find the best implementation for combined security in a given criteria. Instead, we want to showcase the proposed transformation for different scenarios. For this, we focus on constructions with $d=1, k=1$ since higher-order security requires the implementation of k consecutive majority votes per isolation layer I, which we deem impractical. All our circuits [2] are synthesized using Synopsys Design Compiler and the Nangate 45nm OpenCell library. For our first case study, we do provide a tool-based security verification, highlighting the soundness of our approach. For combined security, there currently exists no established way for practical security evaluation, i.e., there is no analog for Test Vector Leakage Assessment (TVLA) [GGJR+11, BCD+13, SM15]. Hence, we refrain from providing practical security evaluations in this work.

6.1 4-Bit S-boxes

We start our evaluation with 4-bit S-boxes. For that, we implement ten representatives, each from a different equivalence class, taken from Bilgin et *al.* [BNN+12]. Together with affine transformations (consisting only of xor gates), those ten representatives allow the construction of all possible 4-bit S-boxes via composition.

We implement two variants of each representative: (i) As TI to protect against SCA with $d=1$. Here we use the minimal number of shares for a uniform and non-complete implementation and place a register stage at the output. The registers are put in place to stop glitches in the case of composition. While not all registers are strictly necessary for security (some outputs are just a pass-through of some input) we nevertheless place them to ensure a pipelined design. (ii) Applying T^{CA}_{CMS} to the TI implementation to protect against CA with $d=1$ and $k=1$. Here, we place only the corrections necessary for security, i.e., no corrections when there is no combinatorial logic for an output (cf.

[2] Available at https://github.com/Chair-for-Security-Engineering/Combined-Threshold-Implementation.

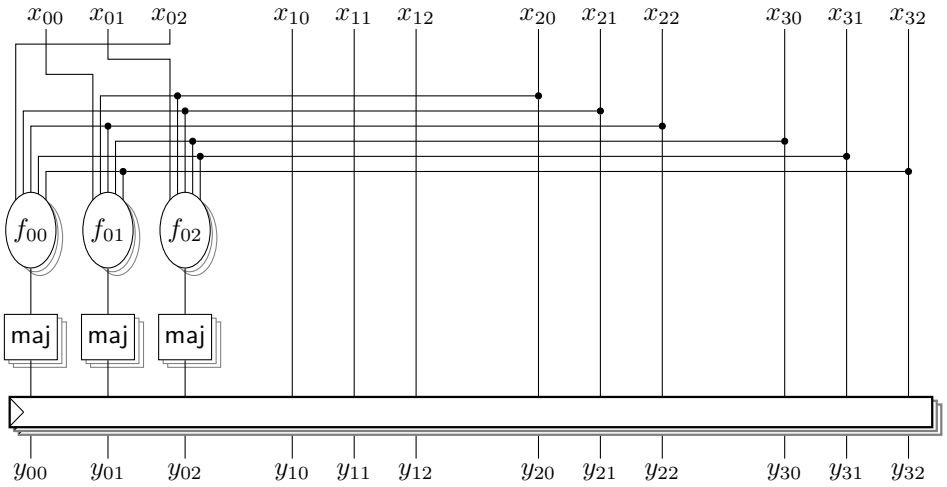


Figure 3. Structure of Q_4^4 as CTI.

Table 2. Performance and verification results for the TI and CTI implementations of ten equivalence classes for 4-Bit S-boxes. Here, s represents the number of shares and ×TI it the overhead factor over TI.

Design			TI			CTI				
Class	Representative	s	GE	Lat.	Rand.	GE	×TI	Lat.	Rand.	Verif
C_1^4	0123456789ABCDFE	4	304.0	1	0	964.0	3.17	1	0	$(1,1)^\checkmark$
C_3^4	0123456789ABDEFC	4	336.0	1	0	1112.0	3.31	1	0	$(1,1)^\checkmark$
Q_4^4	0123456789ABDCFE	3	96.0	1	0	327.0	3.41	1	0	$(1,1)^\checkmark$
Q_{12}^4	0123456789CDEFAB	3	132.0	1	0	490.0	3.71	1	0	$(1,1)^\checkmark$
C_{13}^4	0123456789CDEFBA	4	405.7	1	0	1398.0	3.45	1	0	$(1,1)^\checkmark$
Q_{293}^4	0123457689CDEFBA	3	160.0	1	0	613.0	3.83	1	0	$(1,1)^\checkmark$
Q_{294}^4	0123456789BAEFDC	3	123.0	1	0	447.0	3.63	1	0	$(1,1)^\checkmark$
Q_{299}^4	012345678ACEB9FD	3	210.3	1	0	748.0	3.56	1	0	$(1,1)^\checkmark$
C_{300}^4	01234589DC76BAFE	4	310.7	1	0	1159.3	3.73	1	0	$(1,1)^\checkmark$
C_{301}^4	01234589DC76ABFE	4	511.0	1	0	1875.3	3.67	1	0	$(1,1)^\checkmark$
–	Present	3	306.0	2	0	1106.00	3.61	2	0	$(1,1)^\checkmark$

Figure 3). In short, those corrections are not required since it does not matter whether the fault is injected at the output register or already existent in the input.

We report the performance characteristics for the area, latency and required bits of randomness in Table 2. We also report the overhead factor for the area over the TI design (×TI). Since there is no randomness for refreshing required, the overhead is entirely determined by the introduced replication and corrections. In addition, we conducted a security verification for CA using VERICA [RBFSG22]. Currently, VERICA is the only available tool able to verify for combined security since other tools are focused on either SCA [ANR18, BBC+19, BMRT22, MM22] or FIA [AWMN20, RBRSS+21, WLR+22]. As can be seen in Table 2, all our implementations passed the verification.

To showcase the application of those equivalence classes, we implement the PRESENT S-box [BKL+07] out of Q_{12}^4. In particular, we use the composition from Sasdrich et *al.* [SBM18] where $S = A'' \circ Q_{12}^4 \circ A' \circ Q_{12}^4 \circ A$, with

$$A = \mathsf{01AB892345EFCD67}$$
$$A' = \mathsf{0B835ED61A924FC7}$$
$$A'' = \mathsf{C98D6327AFEB0541}.$$

The performance and verification results can again be seen in Table 2. As expected, the overhead factor for combined security decreases since there is more logic without the need

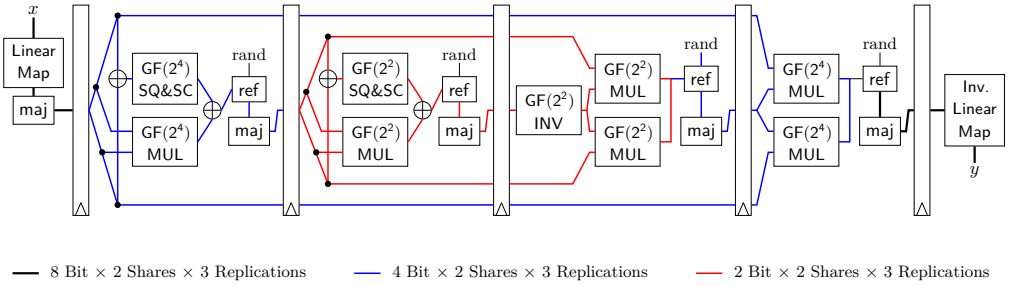


Figure 4. Structure of AES S-box as CCMS.

for additional protection (A, A', and A'').

6.2 AES S-box

We continue our analysis with the AES S-box. In a masked implementation, the S-box is (due to its non-linearity) usually the most costly part and at least one needs to be implemented regardless of the used architecture. Hence, the S-box is a good predictor of the overall implementation costs. We implement all variants in a pipelined fashion, i.e., introduce registers not necessarily required for security. For TI-like implementations, this comes usually with only a small overhead since there are a lot of registers required anyway. The performance characteristics of all implementations can be found in Table 3. Where applicable, we also report the overhead factor for the area over the respective SCA variant. Unfortunately, we were not able to run VERICA [RBFSG22] for formal verification of our implementations, due to the size of the circuits.

Reviewing Previous Works. The AES S-box is probably the most researched construction in cryptography and we, therefore, use this S-box for a comparison with the existing literature. In particular, we compare our implementations to existing protection schemes for CA that are currently unbroken and suitable for hardware (i.e., consider glitch extension). This criterion leaves us with M&M with λ-detection [HMA+24], tessellate gadgets [DN20a], and CPC gadgets [FGM+23]. For all schemes, we report numbers for protection against a single, arbitrary fault and a simultaneous single, arbitrary probe. For M&M with λ-detection, and tessellate gadgets, three shares are required for the required security order. For the first two of those schemes, we report the numbers taken from the respective publication (which uses the same cell library). However, it is not entirely clear whether those implementations are pipelined. For M&M with λ-detection, we report the numbers including the detectors, match checking, and δ-function, as those are required for the CA security of the S-box. Here, the S-box itself requires six cycles for computation, while an additional seven cycles are required until potential faults are detected. The authors report an overhead factor of 3.49 for a byte-serialized AES round. Since all operations of AES except SubBytes are linear, the overhead factor should be slightly higher when considering only the S-box. For the tessellate gadgets, we use our own implementation, based on the Canright S-box [Can05], because there are no numbers for the costs in hardware (and no reference implementation). We did not specify how the ideal abort signal should be implemented and instead wired all detection signals to the output without performing any abort. For CPC we use the Canright S-box as a basis and apply gadgets that protect against one fault and one probe simultaneously. Here, we report the overhead factor compared to an implementation with HPC_1 gadgets [CGLS21], which forms the base for the $\widehat{\mathsf{CPC}_1}$ gadget.

Algorithm 1: CTI multiplication with 4 shares.

function `CTI_Mult`$(a_0^0, \ldots, a_3^2, b_0^0, \ldots, b_3^2)$**:**
  Require: $a_i^\ell = a_i^{\ell'}$ and $b_i^\ell = b_i^{\ell'}$ for $0 \leq \ell, \ell' \leq 2, 0 \leq i \leq 3$
  Require: $\sum_{j=0}^{3} a_j^\ell = a$ and $\sum_{j=0}^{3} b_j^\ell = b$ for $0 \leq \ell \leq 2$
  for $\ell = 0$ to 2 **do**
    `// Compute component functions`
    $\tilde{z}_0^\ell \leftarrow (a_1^\ell + a_2^\ell + a_3^\ell)(b_1^\ell + b_2^\ell) + b_3^\ell + b_2^\ell$
    $\tilde{z}_1^\ell \leftarrow (a_0^\ell + a_2^\ell)(b_0^\ell + b_3^\ell) + a_0^\ell b_2^\ell + a_3^\ell$
    $\tilde{z}_2^\ell \leftarrow (a_1^\ell + a_3^\ell)(b_0^\ell + b_3^\ell) + a_3^\ell + b_3^\ell$
    $\tilde{z}_3^\ell \leftarrow a_0^\ell b_1^\ell + b_3^\ell$
  for $\ell = 0$ to 2 **do**
    for $i = 0$ *to* 3 **do**
      `// Correction`
      $z_i^\ell \leftarrow \mathsf{maj}(\tilde{z}_i^0, \ldots, \tilde{z}_i^2)$
      `// Output register`
      $c_i^\ell \leftarrow \mathsf{reg}[z_i^\ell]$
  Ensures: $c_i^\ell = c_i^{\ell'}$ for $0 \leq \ell, \ell' \leq 2, 0 \leq i \leq 3$
  Ensures: $\sum_{i=0}^{3} c_i^\ell = a \cdot b$ for $0 \leq \ell \leq 2$
  return $c_0^0, \ldots, c_2^3$

Combined Consolidating Masking Schemes. Our first S-box is an application of T_{CMS}^{CA} to the CMS S-box from De Cnudde et *al.* [CRB+16], which is based on the tower field implementation from Canright [Can05]. The resulting S-box has five register stages, two shares, and three replications and requires 62 bits of randomness. This is only eight bit of randomness more than the original SCA variant because we apply the general ring refreshing instead of the optimized refreshing for $d = 1$. That is, for four values at the output of L we use four random bits in the following fashion:

$$\begin{aligned}
\tilde{x}_0 &= (x_0 + r_0) + r_1 \\
\tilde{x}_1 &= (x_1 + r_1) + r_2 \\
\tilde{x}_2 &= (x_2 + r_2) + r_3 \\
\tilde{x}_3 &= (x_3 + r_3) + r_0
\end{aligned}$$

Thereby, each value is already refreshed with two bits of randomness and a single fault cannot de-mask it. The structure of the resulting S-box can be seen in Figure 4. Again, we apply corrections only where necessary for security, i.e., no corrections for wires that are merely a pass-through to a later stage.

Combined Threshold Implementation. Next, we apply T_{CMS}^{CA} to a TI implementation from Ghoshal and De Cnudde [GC17], based on the construction from Boyar and Peralta [BP12]. In particular, we choose the variant with four shares and no randomness. Here, the last non-linear layer has no registers placed on the output. The reason is, that, in a round-based implementation, after the S-box only linear operations are performed and the state registers of the round can be used to stop glitches from leaking any information. The same argument also applies under consideration of a single fault. Hence, the resulting CTI variant has four shares, three replications, and three register stages and requires no fresh randomness. The general construction follows a gadget-like approach, where the TI multiplication is the most interesting part, as this is the only part where corrections are required. We give the precise computation of the CTI multiplication in Algorithm 1. Due

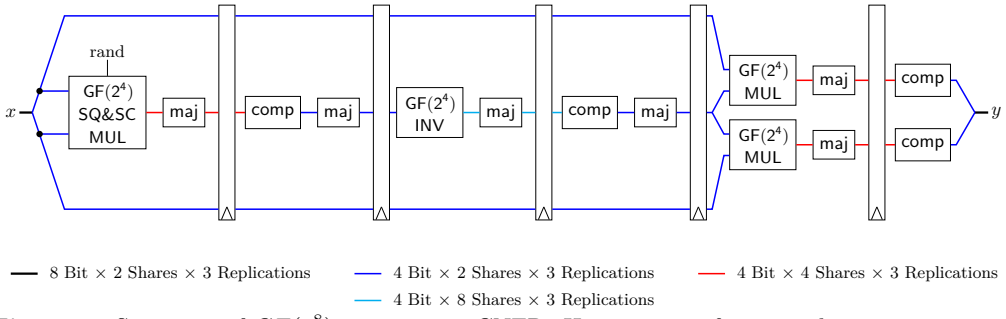


Figure 5. Structure of $GF(2^8)$ inversion as CNFR. Here, comp refers to a share compression.

Table 3. Performance results for CA schemes implementing an AES S-box. Here, s represents the number of shares, n the number of replications, and ×SCA the overhead factor over the SCA area, where applicable. Designs in bold are from this paper.

	Design			Performance			
	Version	s	n	GE	×SCA	Lat.	Rand.
	Plain [Can05]	1	1	226.00		0	0
	Plain [BP12]	1	1	233.33		0	0
SCA	CMS [CRB+16]	2	1	1 824.0		5	54
SCA	TI [GC17]	4	1	3 619.3		3	0
SCA	NFR [SM21a]*	2	1	1 945.7		5	1
CA	λ-detection M&M [HMA+24]†	3	-	29 300.0	3.49‡	6	903
CA	Let's Tessellate [DN20a]§	3	2	17 550.0	-	7	216
CA	$\widetilde{\mathsf{CPC}}_1$ [FGM+23]	2	3	10 882.0	4.01	6	144
CA	**CCMS**	2	3	6 576.0	3.61	5	62
CA	**CTI**	4	3	11 690.0	3.23	3	0
CA	**CNFR***	2	3	7 053.3	3.63	5	2

* Inversion in $GF(2^8)$ only. † Numbers are taken from the respective publication.
‡ For a full byte-serialized AES round. § Without abort handling.

to the gadget-like approach, the number of required corrections is quite low. This also means that the overhead in registers caused by pipelining is (for a TI) high, leading to an overall expensive implementation in area.

Combined Nullifying Fresh Randomness. As a third implementation, we apply T^{CA}_{CMS} to the NFR variant of Shahmirzadi and Moradi [SM21a], again based on the Canright implementation [Can05]. Here, we only implement the underlying inversion in $GF(2^8)$ without the preceding and following affine transformations, because, later, there was a flaw found in the complete construction [MM22]. Nevertheless, since the inversion alone is secure and forms the bulk of computation, we still report the results. We chose the variant with a single square-scale-and-multiply module and one bit of randomness. Hence, after applying T^{CA}_{CMS}, the inversion has two shares, three replications, and five registers stages and requires two bits of randomness (due to step 2 of T^{CA}_{CMS}). The overall structure can be seen in Figure 5. Again, we apply corrections only where required for security.

Discussion. As can be seen in Table 3, constructing $(1, 1)$-CA secure designs out of TI-like masking schemes is beneficial for randomness consumption and latency. In those two categories, the CTI implementation outperforms the state-of-the-art significantly (for NFR there also exists a variant of the $GF(2^8)$ inversion without any randomness). At least for pipelined designs, the benefit of our method also extends to the area consumption. Here, the CCMS variant is significantly smaller than the schemes from the literature. This can be traced to the comparable small amount of registers required for glitch handling,

Table 4. Performance results for an AES round implementation. Here, s represents the number of shares, n the number of replications, and ×SCA the overhead factor over the SCA area. Designs in bold are from this paper.

	Design			Performance			
	Version	s	n	GE	×SCA	Lat.	Rand.
SCA	CMS [CRB+16]	2	1	33 610.7		6	864
SCA	TI [GC17]	4	1	66 778.7		4	0
CA	**CCMS**	2	3	118 496.0	3.53	6	992
CA	**CTI**	4	3	220 304.0	3.30	4	0

which are comparably large components. While in the CMS design only 44% of the area is consumed by registers, this increases to roughly 60% in a (pipelined) design based on HPC_1 gadgets and even roughly 70% in a pipelined design based on tessellate gadgets.

In general, we can observe that T^{CA}_{CMS} keeps the performance hierarchy of TI-like designs constant, i.e., a better design for SCA will directly result in a better design for CA. Hence, we can benefit from the large knowledge and expertise already established in the side-channel community. Indeed, adding SCA protection is the most expensive part of the entire construction. For example, protecting the Canright S-box via CMS increases the area by a factor of 8, while the latency and randomness consumption increases from zero to 5 and 54, respectively. In contrast, adding fault security only increases the area by a factor of 3, while adding combined security on top again increases the area by a factor of 1.2 and the randomness consumption by 8 bits.

6.3 AES-Round Implementation

Finally, we also report the performance characteristics for an entire AES-128 round in Table 4. Our implementation is without the key schedule and control logic. Hence, it contains the operations KeyAdd, SubBytes, ShiftRows, and MixColumns with a state register at the end for the entire 128-bit state at once, i.e., 16 S-boxes in parallel. All linear functions are implemented share-wise, replicated, and require no additional handling for combined security. In essence, we can assume those functions to be part of the non-linear and linear layer of the following S-box, which extends the security proof to those parts. For $d = 1$ and $k = 1$, it is also no problem, that this extended non-linear and linear layer is shared between different component functions since only one of the component functions can be probed. However, for the CTI variant, we added a correction layer in front of the state register to adjust for the missing isolation layer at the end of the S-box. This is the reason, why the overhead factor for the CTI implementation slightly increases compared to the S-box. For the CCMS design, the overhead factor decreases, because there are no additional corrections required and, hence, the relative cost for adding CA security decreases.

7 Conclusion

In this work, we showed how to transform an arbitrary SCA-secure, TI-like circuit into a circuit that can withstand combined attacks. The core idea is the strengthening of the isolation layer (formerly the refresh layer), by replicating the mask refreshing and adding a fault-isolating correction stage between refreshing and registers to ensure isolation with respect to a (d, k)-combined attack. We showed that the resulting circuits indeed surpass the state-of-the-art in $(1, 1)$-combined security for hardware circuits when considering latency (in clock cycles), randomness consumption, and even area for pipelined designs. We achieve this optimization by considering larger parts of the circuit at once, due to the use of TI-like construction as the base.

Future Work. Today, the research community has a good understanding of constructing isolation layers for higher-order probing security (via mask refreshing), glitch extension of probes (via registers), and first-order fault security (via correction). In this work, we extended the knowledge by constructing an isolation layer for combined security. However, the efficient isolation of multiple faults, i.e., removing any dependency on incoming faults under additional faults, is still an open question. While a k-order fault-isolation correction can be constructed via k independent correction modules, the practical application requires a more efficient way. This may be possible by a different correction circuit or by developing an appropriate majority gate at the transistor level. In another line of research, it would be useful to develop and establish a reasonable and meaningful practical evaluation methodology for combined security. The main challenge is to find a way to test a large amount of possible fault scenarios.

Acknowledgments

The work described in this paper has been supported in part by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA - 390781972 and through the projects VE-HEP (16KIS1345) supported by the German Federal Ministry of Education and Research BMBF.

References

[AIS18] Prabhanjan Ananth, Yuval Ishai, and Amit Sahai. Private Circuits: A Modular Approach. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 427–455. Springer, 2018.

[AMR+20] Anita Aghaie, Amir Moradi, Shahram Rasoolzadeh, Aein Rezaei Shahmirzadi, Falk Schellenberg, and Tobias Schneider. Impeccable Circuits. *IEEE Trans. Computers*, 69(3):361–376, 2020.

[ANR18] Victor Arribas, Svetla Nikova, and Vincent Rijmen. VerMI: Verification Tool for Masked Implementations. In *25th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2018, Bordeaux, France, December 9-12, 2018*, pages 381–384. IEEE, 2018.

[APZ21] Melissa Azouaoui, Kostas Papagiannopoulos, and Dominik Zürner. Blind Side-Channel SIFA. In *Design, Automation & Test in Europe Conference & Exhibition, DATE 2021, Grenoble, France, February 1-5, 2021*, pages 555–560. IEEE, 2021.

[AVFM07] Frédéric Amiel, Karine Villegas, Benoit Feix, and Louis Marcel. Passive and Active Combined Attacks: Combining Fault Attacks and Side Channel Analysis. In *Fourth International Workshop on Fault Diagnosis and Tolerance in Cryptography, 2007, FDTC 2007: Vienna, Austria, 10 September 2007*, pages 92–102, 2007.

[AWMN20] Victor Arribas, Felix Wegener, Amir Moradi, and Svetla Nikova. Cryptographic Fault Diagnosis using VerFI. In *2020 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2020, San Jose, CA, USA, December 7-11, 2020*, pages 229–240. IEEE, 2020.

[BBC+19] Gilles Barthe, Sonia Belaïd, Gaëtan Cassiers, Pierre-Alain Fouque, Benjamin Grégoire, and François-Xavier Standaert. maskVerif: Automated Verification of Higher-Order Masking in Presence of Physical Defaults. In *ESORICS*, volume 11735 of *Lecture Notes in Computer Science*, pages 300–318. Springer, 2019.

[BBD+15] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, and Pierre-Yves Strub. Verified Proofs of Higher-Order Masking. In 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*, pages 457–485. Springer, 2015.

[BBD+16] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, Pierre-Yves Strub, and Rébecca Zucchini. Strong Non-Interference and Type-Directed Higher-Order Masking. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 116–129. ACM, 2016.

[BBP+16] Sonia Belaïd, Fabrice Benhamouda, Alain Passelègue, Emmanuel Prouff, Adrian Thillard, and Damien Vergnaud. Randomness Complexity of Private Circuits for Multiplication. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 616–648, 2016.

[BCD+13] Georg T. Becker, Jim Cooper, Elizabeth K. DeMulder, Gilbert Goodwill, Joshua Jaffe, Gary Kenworthy, T. Kouzminov, Andrew J. Leiserson, Mark E. Marson, Pankaj Rohatgi, and Sami Saab. Test Vector Leakage Assessment (TVLA) Methodology in Practice, 2013.

[BCP+20] Sonia Belaïd, Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Abdul Rahman Taleb. Random Probing Security: Verification, Composition, Expansion and New Constructions. In *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part I*, pages 339–368, 2020.

[BCPZ16] Alberto Battistello, Jean-Sébastien Coron, Emmanuel Prouff, and Rina Zeitoun. Horizontal Side-Channel Attacks and Countermeasures on the ISW Masking Scheme. In *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, pages 23–39, 2016.

[BDF+17] Gilles Barthe, François Dupressoir, Sebastian Faust, Benjamin Grégoire, François-Xavier Standaert, and Pierre-Yves Strub. Parallel Implementations of Masking Schemes and the Bounded Moment Leakage Model. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 535–566, 2017.

[BDRS21] Tim Beyne, Siemen Dhooghe, Adrián Ranea, and Danilo Sijacic. A Low-Randomness Second-Order Masked AES. In Riham AlTawy and Andreas Hülsing, editors, *Selected Areas in Cryptography - 28th International Conference, SAC 2021, Virtual Event, September 29 - October 1, 2021, Revised Selected Papers*, volume 13203 of *Lecture Notes in Computer Science*, pages 87–110. Springer, 2021.

[BEF+23] Sebastian Berndt, Thomas Eisenbarth, Sebastian Faust, Marc Gourjon, Maximilian Orlt, and Okan Seker. Combined Fault and Leakage Resilience: Composability, Constructions and Compiler. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 377–409. Springer, 2023.

[BGN+14] Begül Bilgin, Benedikt Gierlichs, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Higher-Order Threshold Implementations. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 326–343. Springer, 2014.

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.

[BMRT22] Sonia Belaïd, Darius Mercadier, Matthieu Rivain, and Abdul Rahman Taleb. IronMask: Versatile Verification of Masking Security. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 142–160. IEEE, 2022.

[BNN+12] Begül Bilgin, Svetla Nikova, Ventzislav Nikov, Vincent Rijmen, and Georg Stütz. Threshold Implementations of All 3 ×3 and 4 ×4 S-Boxes. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings*, volume 7428 of *Lecture Notes in Computer Science*, pages 76–91. Springer, 2012.

[BP12] Joan Boyar and René Peralta. A Small Depth-16 Circuit for the AES S-Box. In *Information Security and Privacy Research - 27th IFIP TC 11 Information Security and Privacy Conference, SEC 2012, Heraklion, Crete, Greece, June 4-6, 2012. Proceedings*, pages 287–298, 2012.

[Can05] David Canright. A Very Compact S-Box for AES. In *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, pages 441–455, 2005.

[CFGR10] Christophe Clavier, Benoit Feix, Georges Gagnerot, and Mylène Roussellet. Passive and Active Combined Attacks on AES: Combining Fault Attacks and Side Channel Analysis. In *2010 Workshop on Fault Diagnosis and Tolerance*

in Cryptography, FDTC 2010, Santa Barbara, California, USA, 21 August 2010, pages 10–19, 2010.

[CGLS21] Gaëtan Cassiers, Benjamin Grégoire, Itamar Levi, and François-Xavier Standaert. Hardware Private Circuits: From Trivial Composition to Full Verification. *IEEE Trans. Computers*, 70(10):1677–1690, 2021.

[CJRR99] 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.

[CRB+16] Thomas De Cnudde, Oscar Reparaz, Begül Bilgin, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Masking AES with d+1 Shares in Hardware. 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 194–212. Springer, 2016.

[CS20] Gaëtan Cassiers and François-Xavier Standaert. Trivially and Efficiently Composing Masked Gadgets With Probe Isolating Non-Interference. *IEEE Trans. Inf. Forensics Secur.*, 15:2542–2555, 2020.

[DAN+19] Lauren De Meyer, Victor Arribas, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. M&M: Masks and Macs against Physical Attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(1):25–50, 2019.

[DDF14] Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying Leakage Models: From Probing Attacks to Noisy Leakage. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 423–440, 2014.

[DDRT12] Amine Dehbaoui, Jean-Max Dutertre, Bruno Robisson, and Assia Tria. Electromagnetic Transient Faults Injection on a Hardware and a Software Implementations of AES. In Guido Bertoni and Benedikt Gierlichs, editors, *2012 Workshop on Fault Diagnosis and Tolerance in Cryptography, Leuven, Belgium, September 9, 2012*, pages 7–15. IEEE Computer Society, 2012.

[DEG+18] Christoph Dobraunig, Maria Eichlseder, Hannes Groß, Stefan Mangard, Florian Mendel, and Robert Primas. Statistical Ineffective Fault Attacks on Masked AES with Fault Countermeasures. In *ASIACRYPT*, volume 11273 of *Lecture Notes in Computer Science*, pages 315–342. Springer, 2018.

[DLM19] Mathieu Dumont, Mathieu Lisart, and Philippe Maurine. Electromagnetic Fault Injection : How Faults Occur. In *2019 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2019, Atlanta, GA, USA, August 24, 2019*, pages 9–16. IEEE, 2019.

[DN20a] Siemen Dhooghe and Svetla Nikova. Let's Tessellate: Tiling for Security Against Advanced Probe and Fault Adversaries. In Pierre-Yvan Liardet and Nele Mentens, editors, *Smart Card Research and Advanced Applications - 19th International Conference, CARDIS 2020, Virtual Event, November 18-19, 2020, Revised Selected Papers*, volume 12609 of *Lecture Notes in Computer Science*, pages 181–195. Springer, 2020.

[DN20b] Siemen Dhooghe and Svetla Nikova. My Gadget Just Cares for Me - How NINA Can Prove Security Against Combined Attacks. In *CT-RSA*, volume 12006 of *Lecture Notes in Computer Science*, pages 35–55. Springer, 2020.

[DOT23] Siemen Dhooghe, Artemii Ovchinnikov, and Dilara Toprakhisar. StaTI: Protecting against Fault Attacks Using Stable Threshold Implementations. *IACR Cryptol. ePrint Arch.*, 2023.

[DSM22] Siemen Dhooghe, Aein Rezaei Shahmirzadi, and Amir Moradi. Second-Order Low-Randomness d + 1 Hardware Sharing of the AES. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pages 815–828. ACM, 2022.

[FGM+23] Jakob Feldtkeller, Tim Güneysu, Thorben Moos, Jan Richter-Brockmann, Sayandeep Saha, Pascal Sasdrich, and François-Xavier Standaert. Combined Private Circuits - Combined Security Refurbished. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, pages 990–1004. ACM, 2023.

[FGP+18] Sebastian Faust, Vincent Grosso, Santos Merino Del Pozo, Clara Paglialonga, and François-Xavier Standaert. Composable Masking Schemes in the Presence of Physical Defaults & the Robust Probing Model. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(3):89–120, 2018.

[FRSG22] Jakob Feldtkeller, Jan Richter-Brockmann, Pascal Sasdrich, and Tim Güneysu. CINI MINIS: Domain Isolation for Fault and Combined Security. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pages 1023–1036. ACM, 2022.

[GC17] Ashrujit Ghoshal and Thomas De Cnudde. Several Masked Implementations of the Boyar-Peralta AES S-Box. In Arpita Patra and Nigel P. Smart, editors, *Progress in Cryptology - INDOCRYPT 2017 - 18th International Conference on Cryptology in India, Chennai, India, December 10-13, 2017, Proceedings*, volume 10698 of *Lecture Notes in Computer Science*, pages 384–402. Springer, 2017.

[GGJR+11] Benjamin Jun Gilbert Goodwill, Josh Jaffe, Pankaj Rohatgi, et al. A Testing Methodology for Side-Channel Resistance Validation. In *NIST non-invasive attack testing workshop*, volume 7, pages 115–136, 2011.

[GMO01] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. Electromagnetic Analysis: Concrete Results. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*, volume 2162 of *Lecture Notes in Computer Science*, pages 251–261. Springer, 2001.

[GPK+21] Michael Gruber, Matthias Probst, Patrick Karl, Thomas Schamberger, Lars Tebelmann, Michael Tempelmeier, and Georg Sigl. DOMREP-An Orthogonal Countermeasure for Arbitrary Order Side-Channel and Fault Attack Protection. *IEEE Trans. Inf. Forensics Secur.*, 16:4321–4335, 2021.

[HMA+24] Haruka Hirata, Daiki Miyahara, Victor Arribas, Yang Li, Noriyuki Miura, Svetla Nikova, and Kazuo Sakiyama. All You Need Is Fault: Zero-Value Attacks on AES and a New λ-Detection M&M. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2024(1):133–156, 2024.

[IPSW06] Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and David Wagner. Private Circuits II: Keeping Secrets in Tamperable Circuits. In *EUROCRYPT*, volume 4004 of *LNCS*, pages 308–327. Springer, 2006.

[ISW03] Yuval Ishai, Amit Sahai, and David A. Wagner. Private Circuits: Securing Hardware against Probing Attacks. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481. Springer, 2003.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[Koc96] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.

[MM22] Nicolai Müller and Amir Moradi. PROLEAD A Probing-Based Hardware Leakage Detection Tool. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(4):311–348, 2022.

[MMSS19] Thorben Moos, Amir Moradi, Tobias Schneider, and François-Xavier Standaert. 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.

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold Implementations Against Side-Channel Attacks and Glitches. In Peng Ning, Sihan Qing, and Ninghui Li, editors, *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*, volume 4307 of *Lecture Notes in Computer Science*, pages 529–545. Springer, 2006.

[RBFSG22] Jan Richter-Brockmann, Jakob Feldtkeller, Pascal Sasdrich, and Tim Güneysu. VERICA - Verification of Combined Attacks: Automated Formal Verification of Security Against Simultaneous Information Leakage and Tampering. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(4):255–284, 2022.

[RBN+15] Oscar Reparaz, Begül Bilgin, Svetla Nikova, Benedikt Gierlichs, and Ingrid Verbauwhede. Consolidating Masking Schemes. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 764–783, 2015.

[RBRSS+21] Jan Richter-Brockmann, Aein Rezaei Shahmirzadi, Pascal Sasdrich, Amir Moradi, and Tim Güneysu. FIVER – Robust Verification of Countermeasures

against Fault Injections. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(4):447–473, Aug. 2021.

[RBSG22] Jan Richter-Brockmann, Pascal Sasdrich, and Tim Güneysu. Revisiting Fault Adversary Models - Hardware Faults in Theory and Practice. *IEEE Trans. Computers*, pages 1 – 14, 2022.

[RDB+18] Oscar Reparaz, Lauren De Meyer, Begül Bilgin, Victor Arribas, Svetla Nikova, Ventzislav Nikov, and Nigel P. Smart. CAPA: The Spirit of Beaver Against Physical Attacks. In *CRYPTO 2018*, volume 10991 of *Lecture Notes in Computer Science*, pages 121–151. Springer, 2018.

[RG20] Jan Richter-Brockmann and Tim Güneysu. Improved Side-Channel Resistance by Dynamic Fault-Injection Countermeasures. In *31st IEEE International Conference on Application-specific Systems, Architectures and Processors , ASAP 2020, Manchester, United Kingdom, July 6-8, 2020*, pages 117–124, 2020.

[RLK11] Thomas Roche, Victor Lomné, and Karim Khalfallah. Combined Fault and Side-Channel Attack on Protected Implementations of AES. In *Smart Card Research and Advanced Applications - 10th IFIP WG 8.8/11.2 International Conference, CARDIS 2011, Leuven, Belgium, September 14-16, 2011, Revised Selected Papers*, pages 65–83, 2011.

[SA02] Sergei P. Skorobogatov and Ross J. Anderson. Optical Fault Induction 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 2–12. Springer, 2002.

[SBJ+21] Sayandeep Saha, Arnab Bag, Dirmanto Jap, Debdeep Mukhopadhyay, and Shivam Bhasin. Divided We Stand, United We Fall: Security Analysis of Some SCA+SIFA Countermeasures Against SCA-Enhanced Fault Template Attacks. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part II*, volume 13091 of *Lecture Notes in Computer Science*, pages 62–94. Springer, 2021.

[SBM18] Pascal Sasdrich, René Bock, and Amir Moradi. Threshold Implementation in Software - Case Study of PRESENT. In Junfeng Fan and Benedikt Gierlichs, editors, *Constructive Side-Channel Analysis and Secure Design - 9th International Workshop, COSADE 2018, Singapore, April 23-24, 2018, Proceedings*, volume 10815 of *Lecture Notes in Computer Science*, pages 227–244. Springer, 2018.

[SFES18] Okan Seker, Abraham Fernandez-Rubio, Thomas Eisenbarth, and Rainer Steinwandt. Extending Glitch-Free Multiparty Protocols to Resist Fault Injection Attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(3):394–430, 2018.

[Sha79] Adi Shamir. How to Share a Secret. *Commun. ACM*, 22(11):612–613, 1979.

[SJB+18] Sayandeep Saha, Dirmanto Jap, Jakub Breier, Shivam Bhasin, Debdeep Mukhopadhyay, and Pallab Dasgupta. Breaking Redundancy-Based Countermeasures with Random Faults and Power Side Channel. In *2018 Workshop*

on Fault Diagnosis and Tolerance in Cryptography, FDTC 2018, Amsterdam, The Netherlands, September 13, 2018, pages 15–22. IEEE Computer Society, 2018.

[SM15] Tobias Schneider and Amir Moradi. Leakage Assessment Methodology - A Clear Roadmap for Side-Channel Evaluations. 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 495–513. Springer, 2015.

[SM21a] Aein Rezaei Shahmirzadi and Amir Moradi. Re-Consolidating First-Order Masking Schemes Nullifying Fresh Randomness. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(1):305–342, 2021.

[SM21b] Aein Rezaei Shahmirzadi and Amir Moradi. Second-Order SCA Security with almost no Fresh Randomness. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(3):708–755, 2021.

[SMG16] Tobias Schneider, Amir Moradi, and Tim Güneysu. ParTI - Towards Combined Hardware Countermeasures Against Side-Channel and Fault-Injection Attacks. In *CRYPTO 2016*, volume 9815 of *Lecture Notes in Computer Science*, pages 302–332. Springer, 2016.

[SRJB23] Sayandeep Saha, Prasanna Ravi, Dirmanto Jap, and Shivam Bhasin. Non-Profiled Side-Channel Assisted Fault Attack: A Case Study on DOMREP. In *Proceedings of 29th Design, Automation and Test in Europe (DATE) 2023*, pages 1–6, Antwerp, Belgium, April 2023. IEEE.

[TNN24] Dilara Toprakhisar, Svetla Nikova, and Ventzislav Nikov. CAPABARA: A Combined Attack on CAPA. In Romain Wacquez and Naofumi Homma, editors, *Constructive Side-Channel Analysis and Secure Design - 15th International Workshop, COSADE 2024, Gardanne, France, April 9-10, 2024, Proceedings*, volume 14595 of *Lecture Notes in Computer Science*, pages 76–89. Springer, 2024.

[Tri03] Elena Trichina. Combinational Logic Design for AES SubByte Transformation on Masked Data. *IACR Cryptol. ePrint Arch.*, page 236, 2003.

[WLR+22] Huanyu Wang, Henian Li, Fahim Rahman, Mark M. Tehranipoor, and Farimah Farahmandi. SoFI: Security Property-Driven Vulnerability Assessments of ICs Against Fault-Injection Attacks. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 41(3):452–465, 2022.

[WM18] Felix Wegener and Amir Moradi. A First-Order SCA Resistant AES Without Fresh Randomness. In Junfeng Fan and Benedikt Gierlichs, editors, *Constructive Side-Channel Analysis and Secure Design - 9th International Workshop, COSADE 2018, Singapore, April 23-24, 2018, Proceedings*, volume 10815 of *Lecture Notes in Computer Science*, pages 245–262. Springer, 2018.

[ZDCT13] Loïc Zussa, Jean-Max Dutertre, Jessy Clédière, and Assia Tria. Power Supply Flitch Induced Faults on FPGA: An In-Depth Analysis of the Injection Mechanism. In *2013 IEEE 19th International On-Line Testing Symposium (IOLTS), Chania, Crete, Greece, July 8-10, 2013*, pages 110–115. IEEE, 2013.