IACR Communications in Cryptology
ISSN 3006-5496, Vol. 2, No. 1, 15 pages.

https://doi.org/10.62056/aesgvurzn

Algebraic Side-Channel Attacks against ISAP's Re-Keying: one Ascon Round May not be Enough for Serial Implementations

Vincent Grosso[1] and François-Xavier Standaert[2]

[1] Université Jean Monnet Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien UMR 5516, F-42023, SAINT-ETIENNE, France, Saint-Étienne, France
[2] UCLouvain, ICTEAM Institute, Crypto Group, Louvain-la-Neuve, Belgium

Abstract. We investigate the side-channel security of ISAP against Algebraic Side-Channel Attacks (ASCA) in a simulated setting where the Hamming weight leakages of its intermediate computations can be recovered. For this purpose, we first describe how these attacks, so far only used to target 8-bit implementations, can be applied to 16-bit or 32-bit implementations. We then use ASCA to discuss the side-channel security claims of ISAP's re-keying, where a single bit of nonce is absorbed per permutation call. Theoretically, this re-keying aims to ensure that attacking more than one permutation call jointly does not improve over attacking the same number of permutation calls independently. Yet, while this expectation is expected to be met for ISAP's conservative parameters (where permutation calls are made of 12 Ascon rounds), the extent to which it does (not) hold for ISAP's aggressive parameters (where permutation calls are made of a single Ascon round) remains an open question. We contribute to this question by showing that for 16-bit implementations, combining the leakages of multiple permutation calls can improve over attacking the same number of permutation calls independently, which contradicts ISAP's (theoretical) leakage-resistance claims. By contrast, for 32-bit leakages, we only show similar weaknesses by guessing a large part of the target state (i.e., more than 128 bits), which only impacts the initialization of ISAP's re-keying and does not contradict its security reduction. These results confirm that for hardware implementations with a sufficient level of parallelism, ISAP's aggressive parameters are probably sufficient, but that for more serial (e.g., software) implementations, slightly more conservative parameters, or the addition of implementation-level countermeasures, are needed.

Keywords: Algebraic Side-Channel Attacks (ASCA) · Lightweight Cryptography · ISAP · Authenticated Encryption · Re-keying · Leakage-Resistance

1 Introduction

1.1 State of the art

ISAP is a lightweight permutation-based authenticated encryption algorithm from ToSC 2017/2020 [DEM+17, DEM+20], which was finalist of the NIST lightweight cryptography competition.[1] It is designed to ease protection against side-channel attacks.

Informally, this simplified protection is obtained thanks to re-keying: ISAP first generates a fresh key K^* thanks to a permutation-based leakage-resilient PRF, which is used to encrypt and authenticate with a sponge construction that also embeds a re-keying

E-mail: vincent.grosso@univ-st-etienne.fr (Vincent Grosso)

[1] https://csrc.nist.gov/Projects/lightweight-cryptography/finalists.

Received: 2025-01-13 Accepted: 2025-03-11

process. As a result, the (expensive) requirement of security against Differential Power Analysis (DPA) of modes without leakage-resistance features is reduced to a (cheaper) requirement of security against Simple Power Analysis (SPA) [BBC+20]. Precisely, ISAP limits the adversary to the observation of two (resp., one) input state(s) per permutation call for its re-keying part (resp., its encryption and authentication parts).

Formally, this reduction is the focus of three independent works [DJS19], [DM19], [GPPS20]. All three works analyze ISAP in the ideal permutation model, under the additional (physical) assumption of oracle-free leakage function initially introduced in [YSPY10]. The latter is a convenient solution to avoid artificial "future computation attacks", where the execution of one permutation call leaks about following permutation calls. But the use of idealized primitives is also questioning in the context of side-channel attacks, since the very notion of leakage is inherently related to the definition of an implementation that ideal permutations (or random oracles) do not have. So the level of independence between the inputs and outputs of a leaking implementation that such an analysis requires is a strong (in part physical) assumption which, to the best of our knowledge, has not been discussed in the literature so far. Informally, ISAP's theoretical leakage-resistance claims are made explicit on page 14 of its specifications, which say: "*given two consecutive permutations p with leakages l_i and l_{i+1}, respectively, the maximum an adversary might learn about the state is $l_i + l_{i+1}$*".[2] Summing the leakage is possible because the proof assumes an adversary cannot combine them otherwise than independently. Yet, and even without leakage, the ideal permutation model that is needed for ensuring this independence is expected to be approached when using ISAP's conservative security parameters, where each permutation (between the absorption of a bit) is made of 12 Ascon rounds [DEMS21], not when using its aggressive parameters where a single Ascon round is used for this purpose.

Limiting the number of rounds in symmetric primitives up to the point where distinguishers exist but have no impact on the security of the modes using such (non-ideal) primitives has become a standard approach in lightweight cryptography. For permutation-based designs, it is captured by the difference between the so-called "hermetic sponge strategy", which targets the absence of distinguishers in an absolute sense (and therefore simplifies the interpretation of the proofs), and more efficient non-hermetic designs, where distinguishers on the underlying primitives are tolerated as long as they do not affect the security of the mode [BDPA11]. In the case of ISAP's re-keying with aggressive parameters, instantiating an ideal permutation with a single round of Ascon's permutation is obviously far from hermetic, raising the question whether it can lead to concrete weaknesses.

To some extent, the authors of ISAP already hinted towards the affirmative. For example, in the first (2017) version of their work, they say that in this case, "*the side-channel leakage between single permutation calls can clearly be combined*" [DEM+17]. However, and again to the best of our knowledge, the impact of such combinations on the gap between the aforementioned theoretical leakage-resistance of ISAP and its practical side-channel security guarantees has not yet been analyzed in the literature.

1.2 Contribution

Concretely, a minimum security requirement for the ideal permutation model to make sense is that a side-channel attack against two rounds of ISAP's re-keying should not significantly improve over two independent attacks against a single round. Or more generally, for a number of Ascon permutation rounds r per permutation call in ISAP's re-keying, it is expected that the probability of success $\mathsf{SR}(2r)$ for twice the number of rounds is not significantly better than $1 - \big(1 - \mathsf{SR}(r)\big)^2$. Analyzing such a requirement is non-trivial due to the limitation of ISAP's attack surface to SPAs, which puts strong constraints on the

[2] https://isap.iaik.tugraz.at/files/isapv20.pdf

exploitation of the side-channel information. As a result, and as an admittedly theoretical first step in this direction, we analyze the application of Algebraic Side-Channel Attacks (ASCA) [RS09, RSV09] in a simulated setting where the adversary is assumed to obtain standard leakages such as noise-free Hamming weights from her implementation.

Our first (technical) contribution in this direction is to describe how to efficiently encode Hamming weight leakages for intermediate computations of up to 32 bits (while previously published ASCA results were limited to 8-bit leakages).

We then use this contribution to analyze ISAP's re-keying against side-channel adversaries able to observe 8-bit, 16-bit and 32-bit Hamming weights. We do that for $2^{\kappa r}$-bounded attacks, where the adversary can combine the leakage collected for r Ascon rounds, with an absorption of κ bits between each round. By computing the probability of success for different number of rounds, we observe that one round may not be enough for ISAP's (theoretical) leakage-resistance claims to hold in software implementations. In particular, for attacks using 16-bit leakages, we show successful state recoveries against $r = 2$ rounds that beat the $1 - \big(1 - \mathsf{SR}(r = 1)\big)^2$ barrier with practical complexity for the main (nonce absorption) steps of ISAP's re-keying. By contrast, for attacks using 32-bit leakages, we can only show similar results by guessing a large proportion of the secret state (i.e., more than 128 bits). Those results may still have impact to attack the re-keying's initialization phase (where 192 bits are public out of 320) but they do not contradict ISAP's security reduction, since this initialization uses 12 Ascon rounds and the improvement of multi-round attacks we show only holds for less rounds. So the 32-bit results rather show a context where the level of leakage is just too high for this (initialization) part of the implementation. For completeness, we also show results of a "noise-tolerant" version of ASCA, introduced in [RSV09] and further elaborated in [ZWG+11], which we apply to 8-bit Hamming weight leakages. They provide another example where "one round of Ascon is not enough" for ISAP's theoretical leakage-resistance claims to hold.

We note that these analysis are inherently heuristic: they are based on combining guessing (for a part of the state) and ASCA running with bounded time. While the impact of guessing is easy to predict, the running time of ASCA is much harder to anticipate. Despite this heuristic nature, our results lead to the important conceptual reminder that leakage-resistance assumptions can be both algorithm-dependent and implementation-dependent. We further discuss the impact of these results for ISAP in conclusions.

We also note that our results apply nearly identically to the variant of ISAP using Keccak's rounds in place of Ascon's rounds. The only difference is that (without special care) Keccak's baseline implementation leverages 16-bit words whereas Ascon's baseline implementation is based on (harder to exploit) 32-bit words [BBC+20, KPP20].

1.3 Related works

In the recent literature, ASCA have essentially been superseded by Soft Analytical Side-Channel Attacks (SASCA) [VGS14], which have a better tolerance to noise. For example, [KPP20], [BBC+20], [YK21] and the recent [CDSU23] consider single-trace SASCA against unprotected implementations of Keccak and Ascon. However, all of them fail to provide non-negligible success rates against a 32-bit implementation of ISAP's re-keying, for which a large proportion of the permutation state has to be recovered. More precisely, [VGS14] and [BBC+20] are only conclusive for small (8-bit and 16-bit) implementations. As for [KPP20], [YK21] and [CDSU23], which target the 32-bit case, the two first references marginalize the distributions on small chunks (hence loose information) to limit the attacks' computational cost and the three of them struggle to recover 319 bits out of the 320 ISAP state bits. As a result, ASCA re-appears as an interesting first step that may better deal with the combination of large target intermediate variables and large proportions of permutation states to recover, allowing an admittedly theoretical discussion

of ISAP's leakage-resistance assumptions. It also suggests investigating whether improved noise-tolerant strategies can make our conclusions more concrete as an open problem.

2 Background

2.1 ISAP authenticated encryption

ISAP is an authenticated encryption algorithm designed for improved side-channel security without strong expertise. The core component enabling this goal is the re-keying scheme of Figure 1, which can be viewed as a permutation-based leakage-resilient PRF [BSH+14]. After an initialization phase that mixes a 128-bit long-term key K with a 192-bit (constant) IV, it absorbs a 128-bit nonce Y bit by bit (so that the capacity $\kappa = 1$ on the figure and we call $P_B, w = 128$ times) in order to produce a fresh key K^*.

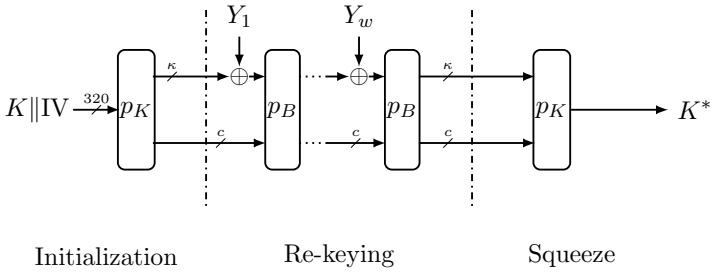
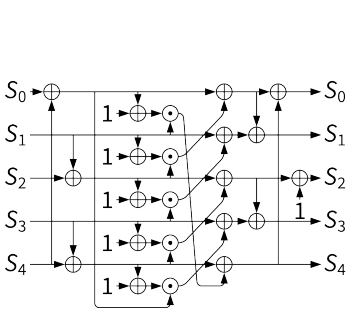
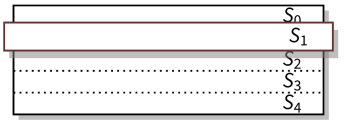


Figure 1: ISAP's re-keying function.

The primary instance of ISAP, on which we focus in this work, is using the Ascon permutation, which essentially alternates a bitslice S-box layer ρ_S and a linear layer ρ_L on a $(5 \times 64) = 320$-bit state, as represented in Figure 2. It also includes a constant addition layer ρ_C that we do not illustrate. This instance comes with two sets of parameters. A conservative one where the number of Ascon rounds in p_B equals 12 and an aggressive ones, on which we focus, where the number of Ascon rounds in p_B is reduced to 1.





$$x_0 = x_0 \oplus (x_0 >>> 19) \oplus (x_0 >>> 28)$$
$$x_1 = x_1 \oplus (x_1 >>> 61) \oplus (x_1 >>> 39)$$
$$x_2 = x_2 \oplus (x_2 >>> 1) \oplus (x_2 >>> 6)$$
$$x_3 = x_3 \oplus (x_3 >>> 10) \oplus (x_3 >>> 17)$$
$$x_4 = x_4 \oplus (x_4 >>> 7) \oplus (x_4 >>> 41)$$

Figure 2: Ascon's permutation: S-box (left) and linear (right) layers.

2.2 Algebraic side-channel attacks

ASCA are a type of horizontal side-channel attack first introduced in the context of block ciphers [RS09, RSV09]. Their core idea is to represent the target implementation as a

system of equations. Side-channel information is then added to the system in order to help its resolution. Once the system is described as a Boolean SATisfiability problem (SAT), an off-the-shelf solver can be used to try finding a solution. Such an attack is analytical since contrary to divide-and-conquer attacks, it can (at least theoretically) leverage the leakage of all the intermediate computations of a block cipher implementation.

Originally, ASCA were presented to break word-oriented implementations of the AES and other block ciphers. When considering such implementations, one of the main challenge is the S-box representation. Indeed, the compactness of the representation (i.e., its number of clauses, number of variables per clause, and number of variables) was shown to have a hard-to-characterize impact on the solver performances. This challenge turns out to be less difficult in the context of permutation-based cryptography, which encourages bitslice implementations allowing a more straightforward correspondence between the algorithm's specifications and its implementation (which typically happens for ISAP).

As mentioned in introduction, one of the main drawback of ASCA is that they require error-free information, whereas actual side-channel attacks generally provide statistical information. This is a reason why ASCA were gradually superseded by SASCA. Yet, since the re-keying of ISAP only limits the number of different inputs on which a permutation is run and does not prevent the adversary to average out the noise, this limitation is less critical in our context. As a natural starting point, we will therefore assume an adversary who can measure the Hamming weight of the target intermediate computations. We nevertheless show in Sections 3.5 and 4.2 that mild amounts of noise can be tolerated with a SAT-based approach, leading to a tradeoff between the level of noise and the size of the target implementations's variables, which both decrease the leakage's informativeness.

As for the precise solver we used, in 2021 a crypto track was present in the SAT competition: different solvers had to find a solution for different problems in cryptography (e.g., proof of work, preimage computation, prime testing, quadratic residue computation, fault attacks against PRESENT, LED or the AES). Since the results of the competition do not suggest a clear winner, we decided to use CryptoMiniSat that is often among the best solvers in SAT competitions and has special treatment for XOR clauses, which is a significant advantage when considering symmetric key cryptography problem.[3]

3 Algebraic representation

We now present the choice of representation for the system of equations representing the Ascon permutation and the Hamming weight leakages that we use in our experiments.

3.1 1-bounded representation

In this setting, we assume an adversary who can observe a single execution of ISAP's re-keying function with a nonce set to zero, which prevents her to learn anything from the absorption (since Hamming weight of the state is the same before and after the absorption). It is conceptually similar to the ASCA in the context of block cipher implementations: the goal of the adversary is to recover the internal state (in place of the key). The only difference is that the secret part of the permutation state can be larger than the size of the secret key. In Figure 3, we schematize the different rounds' states we consider. Since in that case no absorption is performed at the end of each round, we have only one possible state (which should be compared to the more complex Figure 4 in Subsection 3.2).

For the description of the operations ρ_C, ρ_S, ρ_L, we use the Tseytin transform on the Boolean operations [Tse83]. It gives a CNF that grows linearly with the size of the

[3] https://github.com/msoos/cryptominisat.

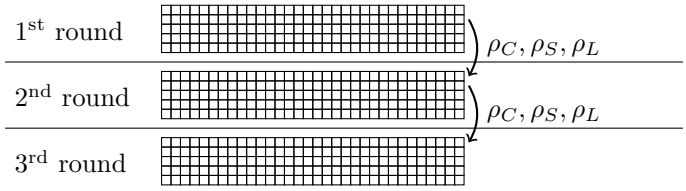


Figure 3: Rounds' states representation for 1-bounded attack.

underlying circuit, by adding variables to the representation for the output of each gate. These variables will also be used to give side-channel information in Section 3.4.

More precisely, and in order to transform an arbitrary combinatorial logic formula, ψ, into one of its CNF representations, a natural approach is to utilize De Morgan's laws. However, employing this strategy will result in an exponential increase in the number and size of the clauses, rendering the formula incompatible with SAT solvers.

To circumvent this exponential growth, Tseytin proposed a general method for constructing a CNF representation of a formula with a linear number of clauses that are small (i.e., contain few variables per clause) but require additional variables (namely, a number that is linear with the number of gates needed to represent the initial formula).

For illustration, consider the formula $\psi = (a \wedge b) \oplus c$. First, we look at the two-input AND gate. For the variables a and b, we introduce an additional variable d that verifies $d = \text{AND}(a, b)$. The resulting formula is as follows: $\gamma = (\neg a \vee \neg b \vee d) \wedge (a \vee \neg d) \wedge (b \vee \neg d)$, and $\psi = d \oplus c \wedge ((\neg a \vee \neg b \vee d) \wedge (a \vee \neg d) \wedge (b \vee \neg d))$. Next, we can write $d \oplus c$ with the variable $e = d \oplus c$, $(\neg e \vee \neg d \vee \neg c) \wedge (\neg e \vee d \vee c) \wedge (e \vee \neg d \vee c) \wedge (e \vee d \vee \neg c)$ and we obtain a CNF representation of $\psi = (\neg e \vee \neg d \vee \neg c) \wedge (\neg e \vee d \vee c) \wedge (e \vee \neg d \vee c) \wedge (e \vee d \vee \neg c) \wedge ((\neg a \vee \neg b \vee d) \wedge (a \vee \neg d) \wedge (b \vee \neg d))$. In general, the Tseytin transformation introduces such an additional variable for the output c of every gate G in ψ. The description of such elementary gates is then directly transformed into a CNF representation γ through the use of De Morgan's law. And eventually, all instances of the gate G are replaced by the additional variables c.

ASCON has a simple description and can be implemented with three gates: XOR, NOT (for constant addition), and NIMP (for the non-linear part) that inverts one input. We give the CNF description we use of these three gates in Table 1. We represent constant additions thanks to NOT gates instead of describing them with XOR gates, in order to reduce the number of clauses. We use the NIMP representation in order to reduce the size of the system, and note that no information can be obtained from the Hamming weight leakages collected for the NOT of a word (since if we know the Hamming weight w of a fixed length l word, the not of this word has Hamming weight $l - w$).

Note that the use of a bitslice representation allows having an efficient and compact representation of the cipher. This contrasts with previous applications of ASCA where the S-boxes' table look-up representation was a bottleneck for efficient representation.

Table 1: CNF representation of the building gate of ASCON.

Name	Formula	CNF
XOR	$c = a \oplus b$	$(\neg c \vee \neg a \vee \neg b) \wedge (\neg c \vee a \vee b) \wedge (c \vee \neg a \vee b) \wedge (c \vee a \vee \neg b)$
NOT	$c = \neg a$	$(\neg c \vee \neg a) \wedge (c \vee a)$
NIMP	$c = a \wedge \neg b$	$(\neg c \vee \neg b) \wedge (\neg c \vee a) \wedge (c \vee \neg a \vee b)$

Looking at the implementation of one permutation, we can infer the number of operations needed (i.e., the number of additional variables) in Table 2, and deduce the number

of clauses. In total, we have 6472 clauses and 1732 additional variables per permutation call. We recall that the constant for the permutation calls has Hamming weight 4.

Table 2: Building gates per layer of the permutation.

	XOR	NOT	NIMP
ρ_S	64×11	64×1	64×5
ρ_C	0	HW(cst)	0
ρ_L	64×10	0	0

3.2 2^r-bounded representation

In this setting, we assume an adversary who can observe executions of ISAP's re-keying for different strings Y. The strings differ on the first bit, allowing her to exploit a 1-bit difference for each call to P_B. That means the internal state is identical before the first permutation call, but we have 2 different internal states for the first round (not fully different since diffusion is not complete) and 2^r ones for the rth round.

We can see that the size of the system (i.e., its number of clauses and variables) grows exponentially with the number of permutation calls. In order to limit this size, we keep the same representation until the string Y considered differs (e.g., for the first round we have only 2 systems and not 2^r systems). Besides, a naive representation would rewrite a complete subsystem for each string Y considered. We rather use the tree-based structure of Figure 4, dividing the system into 2 subsystems if we add a permutation call.

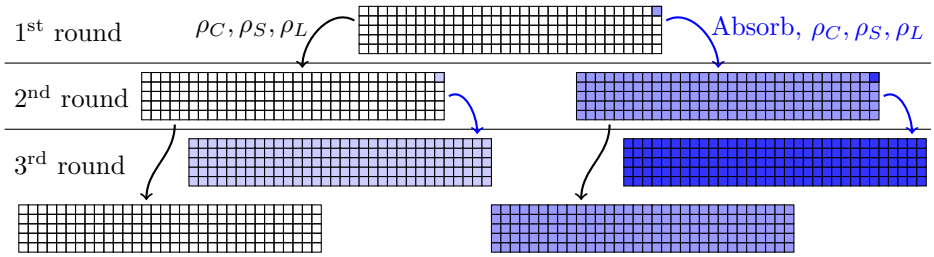


Figure 4: Rounds' states representation for 2^r-bounded attack.

For the rest, we use the same representation as for the 1-bounded case presented in Section 3.1. The nonce absorption is included into the system with a NOT when the string bit is 1. We give numbers for the system size depending on the number of rounds considered in Table 3. For comparison, the system corresponding to 4 permutation calls in the 1-bounded case would be composed of 6928 additional variables and 25888 clauses.

Table 3: Equation systems for 2^r-bounded attacks.

	2^1	2^2	2^3	2^4
Our representation				
# Variables	3465	10 395	24 255	51 975
# Clauses	12 946	38 838	90 622	194 190
Basic representation				
# Variables	3465	13 860	55 440	221 760
# Clauses	12 946	51 784	207 136	816 544

Table 4: Equation systems for $2^{\kappa r}$-bounded attacks.

$r =$	1	2	3
$\kappa = 2$			
# Variables	6932	34 660	138 640
# Clauses	25 892	129 460	543 732
$\kappa = 3$			
# Variables	13 867	124 803	-
# Clauses	51 784	466 056	-

3.3 $2^{\kappa r}$-bounded representation

In this last setting, we consider a slight modification of ISAP's re-keying, where the absorption can be done with $\kappa > 1$ bits (i.e., a rate that can be more than one in Figure 1). Hence, the different strings Y can differ on κ bits, which means that the internal state is identical before the first permutation call, but we have 2^{κ} different internal states for the first round (not fully different since diffusion is not complete) and $2^{\kappa r}$ ones for the rth round. Table 4 reports how the system size increases when κ and r grow.

3.4 Hamming weight encodings

Without additional information, the previous systems of equations are hard to solve. But in the side-channel context, the adversary is provided with (e.g., Hamming weight) leakages on the intermediate computations. One important question in this respect is the representation of such additional information. The basic representation of the Hamming weight that has been considered so far in the literature would list all possibles words of fixed Hamming weight in a Disjunctive Normal Form (DNF) and all impossible words in another DNF, before applying a transformation from DNF to CNF. The representation can alternatively be found by listing all words of Hamming weight + 1 and encoding in CNF that all these words should have a variable at 0, which will set an upper bound for the Hamming weight (with a lower bound that can be set similarly). Yet, these approaches need to enumerate all candidates, which is possible for the Hamming weight of small words (e.g., less or equal to 8 bits) but does not scale up for larger ones observed in practice.

The improved solution we follow in this work is to combine a circuit representation of the Hamming weights with the Tseytin transformation to derive a CNF representation. If the circuit is linear in the size of the word, then we can have a representation that grows linearly with this size. For example, Sinz proposed to represent the Hamming weight computation with (sequential and parallel) counters and derives the formula accordingly [Sin05]. We use the sequential counter representation with a close number of apparitions for each value (since solvers generally try to affect a value to the variable that appears the most, having a bias in the representation of the Hamming weight of a word may be counterproductive). Using such an encoding, the number of clauses and variables to represent a Hamming weight is in $\mathcal{O}(w \cdot h)$, where w is the size of word and k the hamming weight considered. In the basic case it would be $\mathcal{O}\left(\binom{w}{h}\right)$. We recall the equations we use for adding Hamming weight information into the system in Algorithm 1. We denote by $(a_1, \ldots, a_n)$ the leaking word and k its hamming weight. We use $b_{i,j}$ to denote the additional variables and each line corresponds to a new clause. The number of clauses and additional variables is dependent both on the size of the word and the Hamming weight guessed. The algorithm gives clauses for an upper bound on the Hamming weight. The variant using a lower bound uses the same formula but with $(\neg a_1, \ldots, \neg a_n)$ and a Hamming weight $n - k$.

As a result, the solution proposed allows us to have shorter clauses than in previous works like [RSV09] at the cost of additional intermediate variables. This is overall a good

Algorithm 1 Clause construction for Hamming weight upper bound.

Require: $(a_1, \ldots, a_n)$ a word of size n and an upper bound hamming weight k.
Ensure: A CNF clauses.

clauses$\leftarrow \neg a_1 \vee b_{1,1}$
for $1 < j \leq k$ **do**
    clauses$\leftarrow$ clauses $\wedge \neg b_{1,j}$
for $1 < j < n$ **do**
    clauses$\leftarrow$ clauses$\wedge \neg a_i \vee b_{i,1}$
    clauses$\leftarrow$ clauses$\wedge \neg b_{i-1,1} \vee b_{i,1}$
    for $1 < j < k$ **do**
        clauses$\leftarrow$ clauses$\wedge \neg a_i \vee \neg b_{i-1,j-1} \vee b_{i,j}$
        clauses$\leftarrow$ clauses$\wedge \neg b_{i-1,j} \vee b_{i,j}$
    clauses$\leftarrow$ clauses $\wedge \neg a_i \vee b_{i-1,k}$
clauses$\leftarrow$ clauses $\wedge \neg a_n \vee b_{n-1,k}$

tradeoff since it saves DNF to CNF transformation which were the main source for the unrealistic increase of the number of variables when the size of the Hamming weights exploited in an attack increases (e.g., from 8 bits to 16 or 32 bits as we consider next).

3.5 Noise-tolerant encodings

An important, and already recognized, drawback of ASCA is its need of accurate leakages. Indeed, if an adversary extracts erroneous side-channel information and incorporates it into her system of equations, it is likely that the system will become inconsistent. Consequently, the SAT solver will either return UNSAT (which is predominantly the case for block ciphers when information is available for the plaintext and the ciphertext) or SAT with an incorrect guess for the secret. As mentioned in introduction, this is an issue that has been tackled in the literature but that hardly scales with large target intermediate values. Besides SASCA, which offers a natural way to capture noise in analytical attacks, other solutions addressed the possibility of imperfect (e.g., Hamming weight) leakage recovery by rewriting the problem with a different representation. A typical example is to use pseudo-Boolean optimization [OKPW10, ORSW12], which capture possibly erroneous leakages by encoding a lower and an upper bound on their observed value. Quite naturally such an approach can also be exploited with SAT-based ASCA. The resulting "set-ASCA" was for example introduced in [RSV09] and further elaborated in [ZWG+11].

In the following, and despite our main technical result is to show a feasibility result for exploiting the leakage of large target intermediate variables with ASCA, we use set-ASCA to show that our conclusions regarding the number of Ascon rounds needed for ISAP's leakage-resistance claims to hold is not specific to perfect (noise-free) leakages. Quite naturally, such a noise-tolerance (and the reduction of information it implies) has a cost and we only exhibit attacks with (easier-to-exploit) 8-bit leakages in this case.

4 Simulated experiments

In this section, we report results for $2^{\kappa r}$-bounded attacks (for various κ and r values), both in the noise-free Hamming weight leakage case and with a mild amount of noise. We ran our experiment on a computing cluster equipped with AMD EPYC 7F72 processors, running at 3.2 GHz, each experiment is run on 6 cores with 1G of RAM memory.

4.1 Noise-free $2^{\kappa r}$-bounded attacks

For all experiments, we set a solving time limit of 60 minutes (also monitoring intermediate results for 15 and 30 minutes) and we use CryptoMiniSat v5.8.0. We consider 2 different word sizes: 16-bit and 32-bit. (We ran experiments with 8-bit leakages but all attacks trivially succeed in this case). For these 16-bit and 32-bit cases, we additionally consider that a part of the state is guessed, which implies that the solving should be reproduced for each guess for the attack to succeed in practice. Concretely, we always give the correct guess for the state in our experiments. Yet, we note that, in general, giving a bad guess for the state allows to get an UNSAT from the SAT solver much faster. Our results are given in Tables 5 to 9. Each success rate is estimated over 100 independent attacks. For reproducibility, we put the source code of our experiments on an anonymized Git.[4]

Table 5: Success rates of $2^{\kappa r}$-bounded attacks with 16-bit leakage and 32-bit state guess.

κ	0			1		
time (min)	15	30	60	15	30	60
1 round	0	0	0	0	0	0
2 rounds	0	0	0	0.07	0.15	0.27
3 rounds	0	0.01	0.01	0.01	0.01	0.02
κ	2			3		
time (min)	15	30	60	15	30	60
1 round	0.01	0.02	0.02	0.06	0.11	0.16
2 rounds	0.16	0.34	0.49	0.04	0.1	0.19
3 rounds	0	0	0	0	0	0

Table 6: Success rates of $2^{\kappa r}$-bounded attacks with 16-bit leakage and 64-bit state guess.

κ	0			1		
time (min)	15	30	60	15	30	60
1 round	0	0	0	0.06	0.09	0.13
2 rounds	0.01	0.05	0.08	0.64	0.81	0.93
3 rounds	0	0.04	0.07	0.16	0.21	0.33
κ	2			3		
time (min)	15	30	60	15	30	60
1 round	0.18	0.23	0.28	0.48	0.52	0.55
2 rounds	0.82	0.96	1	0.6	0.67	0.74
3 rounds	0	0	0	0	0	0

Starting with preliminary observations, we see that increasing the adversary's time complexity, by giving her more SAT solving time or larger key guesses, always improves the success rate. By contrast, this is not the case when we provide this adversary with more information by increasing r or κ. This is because increasing these parameters increases the size of the systems to solve (as discussed in the previous section). As a result, it can (and does) happen that for a given SAT solving time bound, the additional information of larger systems cannot be exploited. Unsurprisingly, increasing the rate (i.e., κ) nevertheless has a stronger (positive) impact on the attacks' success, since it increases the amount of leakage exponentially (vs. only linearly when increasing the number of rounds r).

[4] https://anonymous.4open.science/r/ASCA-ISAP-COSADE-F34A.

Table 7: Success rates of $2^{\kappa r}$-bounded attacks with 32-bit leakage and 160-bit state guess.

κ	0			1		
time (min)	15	30	60	15	30	60
1 round	0	0	0	0	0	0
2 rounds	0	0	0	0.02	0.11	0.27
3 rounds	0	0	0	0	0.01	0.08
κ	2			3		
time (min)	15	30	60	15	30	60
1 round	0.02	0.02	0.05	0	0.02	0.07
2 rounds	0.01	0.04	0.21	0	0	0.04
3 rounds	0	0	0.02	0	0	0

Table 8: Success rates of $2^{\kappa r}$-bounded attacks with 32-bit leakage and 192-bit state guess.

κ	0			1		
time (min)	15	30	60	15	30	60
1 round	0.13	0.17	0.22	0.13	0.2	0.34
2 rounds	0.41	0.53	0.6	0.22	0.45	0.76
3 rounds	0.33	0.46	0.6	0.14	0.27	0.45
κ	2			3		
time (min)	15	30	60	15	30	60
1 round	0.3	0.43	0.46	0.13	0.39	0.63
2 rounds	0.18	0.41	0.77	0.03	0.11	0.36
3 rounds	0.07	0.21	0.38	0	0.01	0.06

Despite this limitation, Tables 5 and 6 exhibit clear cases of attacks with 16-bit leakage where moving from a 1-round attack to a 2-round attack leads to improvements of the success rate that beat the $\mathsf{SR}(2r) \approx 1 - \big(1 - \mathsf{SR}(r)\big)^2$ barrier, with only a small part of the target state to guess (e.g., 32 or 64 bits out of 320, which is below 128 bits and concretely reachable). As a typical illustration, we can mention the success rates in the first two lines of Table 6 for the $\kappa = 1$ case which exactly corresponds to ISAP with aggressive security parameters. For example with 60 minutes, $1 - (1 - 0.13)^2 = 0.24$ is significantly less than 0.93. So such results show that in the context of software implementations giving rise to 16-bit Hamming weight leakages, the theoretical leakage-resistance guarantees of ISAP do not apply because the independence assumption on which they rely is too significantly contradicted. Concretely, it implies that the security of the full ISAP cannot always be claimed by ensuring the security of its individual blocks, as its security reduction conveniently shows. Importantly (and annoyingly), this can theoretically happen even if each of these individual blocks are SPA-secure when analyzed independently. It is likely that such examples could be showed with larger number of rounds by better balancing the (hard to predict) SAT solving time and the key guesses – which we did not do because the current experiments already correspond to hundreds of hours of computation.

Interestingly, the situation quite significantly differs with 32-bit leakages, as reported in Tables 7 to 9. Here, much larger parts of the state must be guessed to obtain similar results, and ISAP's security reduction is not contradicted. More precisely, our results still show that the initialization phase of ISAP's re-keying (see Figure 1) could be the target of an ASCA, since a much smaller fraction of the state is secret in this case (i.e., 128 bits out of 320, meaning 192 bits are public). Yet, this initialization uses 12 rounds which is likely enough to ensure that $\mathsf{SR}(2r) \approx 1 - \big(1 - \mathsf{SR}(r)\big)^2$. These results match the ones

Table 9: Success rates of $2^{\kappa r}$-bounded attacks with 32-bit leakage and 224-bit state guess.

κ	0			1		
time (min)	15	30	60	15	30	60
1 round	0.92	0.93	0.93	0.94	0.94	0.95
2 rounds	1	1	1	0.98	1	1
3 rounds	1	1	1	0.94	0.98	1
κ	2			3		
time (min)	15	30	60	15	30	60
1 round	0.95	0.95	0.95	0.92	0.95	0.95
2 rounds	0.92	1	1	0.6	0.87	0.98
3 rounds	0.86	0.97	0.99	0.45	0.71	0.87

Table 10: Success rates of $2^{\kappa r}$-bounded attacks with 8-bit noisy leakage (64-bit state guess).

κ	0			1		
time (min)	15	30	60	15	30	60
1 round	0.02	0.02	0.02	0	0	0.5
2 rounds	0.99	0.995	1	0.985	0.995	1
3 rounds	0.605	0.635	0.695	0.68	0.705	0.745
κ	2			3		
time (min)	15	30	60	15	30	60
1 round	0.065	0.065	0.065	0.55	0.55	0.55
2 rounds	1	1	1	1	1	1
3 rounds	0.26	0.335	0.42	0.36	0.7	0.895

in [CDSU23], which show a SASCA against a software implementation in this case.

4.2 Noisy 8-bit leakage

As already mentioned, all ASCA with perfect 8-bit Hamming weight leakages trivially succeed (with probability one). This is not the case anymore in case of noisy Hamming weight leakages. For illustration, we ran set-ASCA with a set corresponding to the two most likely Hamming weights for each leakage sample, with a (mild) noise variance of 0.1 for which all the pairs of Hamming weights of our target system are recovered with probability 0.9. The results of this final experiment are reported in Table 10. They lead to essentially similar conclusions as in the previous sections: a single Ascon round is not enough in this case, since exploiting two rounds of leakage jointly is significantly more effective than targeting them independently, and targeting three rounds does not improve the attacks' success rates, presumably due to computational constraints.

5 Conclusions

The leakage security of ISAP depends on two conditions: (*i*) SPA security for each iteration of the permutation (formally, bounded leakage), which is clearly emphasized by the ISAP designers, and (*ii*) that these SPA secure blocks cannot be attacked jointly more efficiently than independently (formally, the ideal permutation and oracle-free leakage assumptions), which is less explicitly discussed. We show that this second condition is not implied by the first one. It implies consequences from the design and evaluation viewpoints.

From the side-channel security evaluation viewpoint, it means that the use of aggressive

security parameters in a leakage-resilient re-keying scheme leads to the need to question the number of rounds that must be evaluated jointly. We showed that for 16-bit leakages, attacking several rounds of ISAP's re-keying jointly may be the best strategy. Hence, slightly increasing the number of Ascon rounds in the re-keying's permutation calls could be worth to step back in the convenient situation where only 1-bounded attacks must be evaluated, so that side-channel security evaluators avoid the hassle of combining the leakage corresponding to different inputs over multiple rounds with $2^{\kappa r}$-bounded SPA. For 32-bit leakages, such a weakness did not show up in our simulated experiments.

From the design viewpoint, improvements can be considered at the algorithmic or implementation level. At the algorithmic level, increasing the number of rounds in ISAP's permutations can also contribute to ensure the aforementioned condition (*ii*), but it cannot help to ensure condition (*i*). At the implementation level, our results confirm that exploiting the leakage of large parallel implementations with SPA is difficult. Hence, using countermeasures like shuffling that aim to emulate such implementations in software is a natural option [UBS21], which should contribute to solve problems (*i*) and (*ii*) at once.

So overall, increasing the number of rounds in ISAP's permutation calls appears as desirable mostly from the evaluation viewpoint. Combined with a large parallel implementation, it should directly lead to strong security guarantees against leakage. As for more serial implementations, our results rather suggest the addition of implementation-level countermeasures like shuffling in order to limit the level of SPA leakage.

We finally note that our heuristic results only put forward that one round may not be enough for ISAP's theoretical claims of leakage-resistance to hold. As usual with non-hermetic design strategies, this does not give any lower bound on the number of rounds that would be needed to prevent attacks taking advantage of the independence flaw we exhibit. For example, using two rounds between each bit absorption seems hard to attack jointly more effectively than independently with the tools considered in this work. But this may be due to a suboptimal attack strategy. Investigating alternative attack strategies able to exploit more rounds of leakage efficiently, ideally in a more noise-tolerant manner, is therefore an interesting scope for further research. A theoretical characterization of the number of rounds needed to get rid of any distinguishers with leakage (i.e., a generalization of the hermetic design strategy with leakage) would be interesting as well.

Acknowledgments. François-Xavier Standaert is a research director of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been supported by the French Agence Nationale de la Recherche through the grant ANR-22-CE39-0008 (project PROPHY) and funded in part by the ERC Advanced Grant number 101096871 (acronym BRIDGE). Views and opinions expressed in the paper are those of the authors and do not necessarily reflect those of the European Union nor the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

[BBC+20] Davide Bellizia, Olivier Bronchain, Gaëtan Cassiers, Vincent Grosso, Chun Guo, Charles Momin, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Mode-level vs. implementation-level physical security in symmetric cryptography - A practical guide through the leakage-resistance jungle. In *CRYPTO (1)*, volume 12170 of *Lecture Notes in Computer Science*, pages 369–400. Springer, 2020. doi:10.1007/978-3-030-56784-2_13.

[BDPA11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Cryptographic sponge functions, 2011.

[BSH+14] Sonia Belaïd, Fabrizio De Santis, Johann Heyszl, Stefan Mangard, Marcel Medwed, Jörn-Marc Schmidt, François-Xavier Standaert, and Stefan Tillich. Towards fresh re-keying with leakage-resilient PRFs: cipher design principles and analysis. *J. Cryptogr. Eng.*, 4(3):157–171, 2014. doi:10.1007/s13389-014-0079-5.

[CDSU23] Gaëtan Cassiers, Henri Devillez, François-Xavier Standaert, and Balazs Udvarhelyi. Efficient regression-based linear discriminant analysis for side-channel security evaluations towards analytical attacks against 32-bit implementations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(3):270–293, 2023. doi:10.46586/tches.v2023.i3.270-293.

[DEM+17] Christoph Dobraunig, Maria Eichlseder, Stefan Mangard, Florian Mendel, and Thomas Unterluggauer. ISAP - towards side-channel secure authenticated encryption. *IACR Trans. Symmetric Cryptol.*, 2017(1):80–105, 2017. doi:10.13154/tosc.v2017.i1.80-105.

[DEM+20] Christoph Dobraunig, Maria Eichlseder, Stefan Mangard, Florian Mendel, Bart Mennink, Robert Primas, and Thomas Unterluggauer. Isap v2.0. *IACR Trans. Symmetric Cryptol.*, 2020(S1):390–416, 2020. doi:10.13154/tosc.v2020.iS1.390-416.

[DEMS21] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3):33, 2021. doi:10.1007/s00145-021-09398-9.

[DJS19] Jean Paul Degabriele, Christian Janson, and Patrick Struck. Sponges resist leakage: The case of authenticated encryption. In *ASIACRYPT (2)*, volume 11922 of *Lecture Notes in Computer Science*, pages 209–240. Springer, 2019. doi:10.1007/978-3-030-34621-8_8.

[DM19] Christoph Dobraunig and Bart Mennink. Leakage resilience of the duplex construction. In *ASIACRYPT (3)*, volume 11923 of *Lecture Notes in Computer Science*, pages 225–255. Springer, 2019. doi:10.1007/978-3-030-34618-8_8.

[GPPS20] Chun Guo, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Towards low-energy leakage-resistant authenticated encryption from the duplex sponge construction. *IACR Trans. Symmetric Cryptol.*, 2020(1):6–42, 2020. doi:10.46586/tosc.v2020.i1.6-42.

[KPP20] Matthias J. Kannwischer, Peter Pessl, and Robert Primas. Single-trace attacks on keccak. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(3):243–268, 2020. doi:10.1007/978-3-030-97348-3_1.

[OKPW10] Yossef Oren, Mario Kirschbaum, Thomas Popp, and Avishai Wool. Algebraic side-channel analysis in the presence of errors. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*, pages 428–442. Springer, 2010. doi:10.1007/978-3-642-15031-9_29.

[ORSW12] Yossef Oren, Mathieu Renauld, François-Xavier Standaert, and Avishai Wool. Algebraic side-channel attacks beyond the hamming weight leakage model. In *CHES*, volume 7428 of *Lecture Notes in Computer Science*, pages 140–154. Springer, 2012. doi:10.1007/978-3-642-33027-8_9.

[RS09] Mathieu Renauld and François-Xavier Standaert. Algebraic side-channel attacks. In Feng Bao, Moti Yung, Dongdai Lin, and Jiwu Jing, editors, *Information Security and Cryptology - 5th International Conference, Inscrypt 2009, Beijing, China, December 12-15, 2009. Revised Selected Papers*, volume 6151 of *Lecture Notes in Computer Science*, pages 393–410. Springer, 2009. doi:10.1007/978-3-642-16342-5_29.

[RSV09] Mathieu Renauld, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. Algebraic side-channel attacks on the AES: why time also matters in DPA. In *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 97–111. Springer, 2009. doi:10.1007/978-3-642-04138-9_8.

[Sin05] Carsten Sinz. Towards an optimal CNF encoding of boolean cardinality constraints. In Peter van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005, Proceedings*, volume 3709 of *Lecture Notes in Computer Science*, pages 827–831. Springer, 2005. doi:10.1007/11564751_73.

[Tse83] Grigori S Tseitin. On the complexity of derivation in propositional calculus. In *Automation of reasoning*, pages 466–483. Springer, 1983.

[UBS21] Balazs Udvarhelyi, Olivier Bronchain, and François-Xavier Standaert. Security analysis of deterministic re-keying with masking and shuffling: Application to ISAP. In *COSADE*, volume 12910 of *Lecture Notes in Computer Science*, pages 168–183. Springer, 2021. doi:10.1007/978-3-030-89915-8_8.

[VGS14] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Soft analytical side-channel attacks. In *ASIACRYPT (1)*, volume 8873 of *Lecture Notes in Computer Science*, pages 282–296. Springer, 2014. doi:10.1007/978-3-662-45611-8_15.

[YK21] Shih-Chun You and Markus G. Kuhn. Single-trace fragment template attack on a 32-bit implementation of keccak. In *CARDIS*, volume 13173 of *Lecture Notes in Computer Science*, pages 3–23. Springer, 2021. doi:10.1007/978-3-030-97348-3_1.

[YSPY10] Yu Yu, François-Xavier Standaert, Olivier Pereira, and Moti Yung. Practical leakage-resilient pseudorandom generators. In *CCS*, pages 141–151. ACM, 2010. doi:10.1145/1866307.1866324.

[ZWG+11] Xinjie Zhao, Tao Wang, Shize Guo, Fan Zhang, Zhijie Shi, Huiying Liu, and Kehui Wu. Sat based error tolerant algebraic side-channel attacks, 2011.