IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2022, No. 1, pp. 64–87. DOI:10.46586/tosc.v2022.i1.64-87

# Bounds for the Security of Ascon against Differential and Linear Cryptanalysis

Johannes Erlacher[1], Florian Mendel[2] and Maria Eichlseder[1]

[1] Graz University of Technology, Graz, Austria
johannes.erlacher@student.tugraz.at, maria.eichlseder@iaik.tugraz.at
[2] Infineon Technologies AG, Munich, Germany
florian.mendel@infineon.com

**Abstract.** The NIST Lightweight Cryptography project aims to standardize symmetric cryptographic designs, including authenticated encryption and hashing, suitable for constrained devices. One essential criterion for the evaluation of the 10 finalists is the evidence for their security against attacks like linear and differential cryptanalysis. For Ascon, one of the finalists and previous winner of the CAESAR competition in the 'lightweight' category, there is a large gap between the proven bounds and the best known characteristics found with heuristic tools: The bounds only cover up to 3 rounds with 15 differentially and 13 linearly active S-boxes, insufficient for proving a level of security for the full constructions.

In this paper, we propose a new modeling strategy for SAT solvers and derive strong bounds for the round-reduced Ascon permutation. We prove that 4 rounds already ensure that any single characteristic has a differential probability or squared correlation of at most $2^{-72}$, and 6 rounds at most $2^{-108}$. This is significantly below the bound that could be exploited within the query limit for keyed Ascon modes. These bounds are probably not tight. To achieve this result, we propose a new search strategy of dividing the search space into a large number of subproblems based on 'girdle patterns', and show how to exploit the rotational symmetry of Ascon using necklace theory. Additionally, we evaluate and optimize several aspects of the pure SAT model, including the counter implementation and parallelizability, which we expect to be useful for future applications to other models.

**Keywords:** No keywords given.

## 1 Introduction

The NIST Lightweight Cryptography (LWC) project [Nat18] aims to standardize symmetric cryptographic designs suitable for constrained devices. After the CAESAR competition for authenticated encryption [CAE14], which introduced its category for lightweight use-cases in round 3, this is the second competitive effort aiming to fill this gap in the current cryptographic standard landscape. The NIST LWC project aims to standardize a lightweight authenticated encryption algorithm, plus potentially a lightweight hash function. Started in 2019 with 56 first-round candidates, the candidate designs have since been narrowed down to 10 finalists. In the final round, the remaining candidates are evaluated by several criteria, including their performance on different platforms and in different scenarios, as well as the level of trust in their security.

Ascon is one of the 10 NIST LWC finalists [DEMS21a] and the 'first choice' for lightweight authenticated encryption in the portfolio of the CAESAR competition [DEMS16, DEMS21b]. The Ascon suite includes authenticated ciphers and hash functions, all based on the same 320-bit Ascon permutation in different sponge [BDPV07, BDPV08] and

Received: 2021-11-23 Accepted: 2022-01-23 Published: 2022-03-11

duplex [BDPV11] constructions. The permutation is also used in another finalist of the NIST LWC competition, Isap [DEM+21, DEM+20]. The permutation and its use in these schemes have been subject to extensive cryptanalysis, particularly algebraic and differential attacks. The results confirm the security claim and a comfortable security margin.

However, for differential and linear cryptanalysis, there is a noteworthy discrepancy between the best known characteristics and attacks on the one hand, and provable bounds on the other hand: The best bounds only reach up to 3 rounds and show at least 15 differentially or 13 linearly active S-boxes [DEMS15], insufficient for proving a level of security for the full constructions, which use the 6-round, 8-round, and 12-round permutation. On the other hand, heuristic results indicate a high number of active S-boxes starting at 4 rounds [DEMS15, DEM15b]. We summarize the results in Table 1. The substantial gap can partly be attributed to the weak alignment and large state size of the Ascon permutation, which results in a very large search space for automated solvers and makes exhausting this space very costly. This may explain why the only known bounds for 3 rounds from 2015 [DEMS15] have not been extended since, nor have the best known characteristics improved significantly.

In this paper, we show how to manually partition the search space efficiently and thus derive new bounds for the Ascon permutation. We prove that 4 rounds already ensure that any single characteristic has at least 36 active S-boxes and thus a differential probability or squared correlation of at most $2^{-72}$. This is significantly below the bound that could be exploited within the query limit for keyed Ascon modes, for example for collisions during message processing to construct forgeries. We show how to reuse partial results to prove a bound of at least 54 active S-boxes for 6 rounds. The bound also implies at least 108 active S-boxes or a probability of at most $2^{-216}$ for any single characteristic for the full 12-round permutation.

This bound is most likely not tight; we found no matching characteristic. We provide a runtime estimate for checking the bound of at least 40 active S-boxes, which is feasible but beyond our own computational budget. Of course, the bounds need to be interpreted with a grain of salt. Clustering effects may lead to differentials with a higher overall probability [AK18]. Furthermore, the probability assumes independence between rounds as given under independent round keys; however, permutations have no round keys [DEM16]. So far, there are no indications that these effects significantly change the differential probability.

To achieve this result, we propose a new search strategy of dividing the search space into a large number of subproblems based on 'girdle patterns'. This approach addresses the main problems found with previous models: the large search space and the lack of parallelizability. From the SAT solver's perspective, the search space appears much larger than it actually is. Our experiments show that the solver cannot exploit the strong rotational symmetry of the Ascon permutation and its characteristics. We show how to take advantage of this symmetry by applying necklace theory to our girdle patterns. This also allows us to reuse the computational efforts spent on the 4-round bound for proving the 6-round bound. Additionally, we evaluate and optimize several aspects of the pure SAT model, including a pre-filtering strategy for patterns, tailoring the counter implementation, and optimizing the parallelization strategy, which we expect to be useful for future applications to other models.

We provide the source code of our SAT model framework and the intermediate filtered 3-round patterns on https://extgit.iaik.tugraz.at/castle/tool/ascon_sat_bounds.

**Outline.** In Section 2, we recall the specification of Ascon and previous results on its differential and linear properties obtained with automated tools. In Section 3, we describe our basic SAT model. We introduce our new search strategy and the underlying combinatorial details in Section 4 and present the resulting bounds for 4-round Ascon in Section 5. Finally, we conclude in Section 6.

**Table 1:** Bounds for differential cryptanalysis of the reduced $R$-round Ascon permutation. "Min/Max" refers to provable bounds on the minimum number of active S-boxes and maximum probability, where "$\geq/\leq$" indicates not necessarily tight bounds without a matching characteristic. "Best" (wrt. either the number of S-boxes #S, probability $p$, or squared correlation $c^2$) refers to best characteristics found using heuristic search tools. DDT = Differential distribution table of the S-box, LAT = Linear approximation table, $\mathcal{B}$ = Branch number of the linear layer, SAT = Satisfiability, SMT = Satisfiability modulo theories, CP = Constraint programming, MILP = Mixed-integer linear programming, `nldtool` is a dedicated guess-and-determine tool for differential cryptanalysis [MNS11].

**(a)** Differential characteristics

| $R$ | Min # S-boxes, max probability | | | | Best found characteristics | | | |
|---|---|---|---|---|---|---|---|---|
| | #S | $p$ | Reference | Method | #S | $p$ | Reference | Method |
| 1 | 1 | $2^{-2}$ | | DDT | 1 | $2^{-2}$ | DDT | |
| 2 | 4 | $2^{-8}$ | | DDT, $\mathcal{B}$ | 4 | $2^{-8}$ | DDT, $\mathcal{B}$ | |
| 3 | 15 | $\leq 2^{-30}$ | [DEMS15] | SMT | 15 | $2^{-40}$ | [DEMS15] | `nldtool` |
| 4 | $\geq 36$ | $\leq 2^{-72}$ | Section 5 | SAT | 44 | $2^{-107}$ | [DEMS15] | `nldtool` |
| 5 | – | | | | 78 | $2^{-190}$ | [DEMS15, GPT21] | CP |
| 6 | $\geq 54$ | $\leq 2^{-108}$ | Section 5 | SAT | – | | | |

**(b)** Linear characteristics

| $R$ | Min # S-boxes, max correlation | | | | Best found characteristics | | | |
|---|---|---|---|---|---|---|---|---|
| | #S | $c^2$ | Reference | Method | #S | $c^2$ | Reference | Method |
| 1 | 1 | $2^{-2}$ | | LAT | 1 | $2^{-2}$ | LAT | |
| 2 | 4 | $2^{-8}$ | | LAT, $\mathcal{B}$ | 4 | $2^{-8}$ | LAT, $\mathcal{B}$ | |
| 3 | 13 | $\leq 2^{-26}$ | [DEMS15] | SMT | 13 | $2^{-28}$ | [DEM15b] | `lineartrails` |
| 4 | $\geq 36$ | $\leq 2^{-72}$ | Section 5 | SAT | 43 | $2^{-98}$ | [DEM15b] | `lineartrails` |
| 5 | – | | | | 67 | $2^{-186}$ | [DEM15b] | `lineartrails` |
| 6 | $\geq 54$ | $\leq 2^{-108}$ | Section 5 | SAT | – | | | |

# 2 Background

## 2.1 Specification of the Ascon Family

Ascon was first published as a candidate and eventual 'first choice' for lightweight scenarios in the final portfolio of the CAESAR competition for authenticated encryption [DEMS16]. Since then, the family has been extended by hashing schemes and is now a finalist in the NIST LWC lightweight cryptography standardization process [DEMS21a, DEMS21b]. The Ascon permutation underlying all these schemes of the Ascon family is also used in a second LWC finalist, Isap [DEM+21, DEM+20].

The Ascon permutation works on 320-bit blocks, represented as a state of five 64-bit words illustrated in Figure 1 and denoted as $S = x_0 \parallel x_1 \parallel x_2 \parallel x_3 \parallel x_4$.

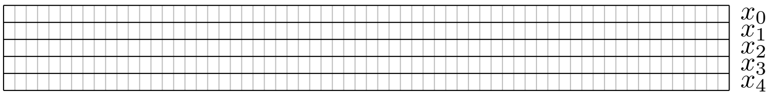


**Figure 1:** The Ascon state of $320 = 5 \times 64$ bits.

**Table 2:** Number of rounds and rate used in the Ascon and Isap family members [DEMS21a, DEM+21]. In the initialization and finalization phases, the rate refers to the absorbing rate of the nonce $N$ and the squeezing rate of the tag $T$ or hash $H$, respectively.

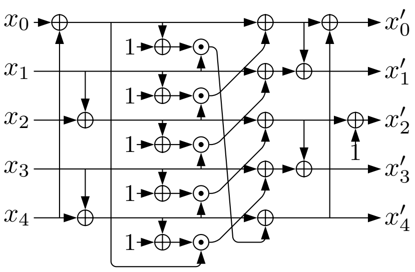
| Type | Variant | Initialization | | Data processing | | Finalization | |
|---|---|---|---|---|---|---|---|
| | | rounds | rate | rounds | rate | rounds | rate |
| | | $a$ | $\lvert N \rvert$ | $b$ | $r$ | $a$ | $r$ or $k$ |
| Hash | Ascon-Hash<br>Ascon-Xof | – | – | 12 | 64 | 12 | 64 |
| | Ascon-HashA<br>Ascon-XofA | – | – | 8 | 64 | 12 | 64 |
| AEAD | Ascon-128<br>Ascon-80pq | 12 | 128 | 6 | 64 | 12 | 128 |
| | Ascon-128a | 12 | 128 | 8 | 128 | 12 | 128 |
| Re-key | Isap-A-128a | – | – | 1 | 1 | 12 | 128 |
| | Isap-A-128 | – | – | 12 | 1 | 12 | 128 |

The permutation is used in different variants with a different number of rounds, denoted $p^a$ for $a$ rounds and $p^b$ for $b$ rounds, where $a, b \in \{6, 8, 12\}$ for Ascon, while Isap also uses a single-round variant. In all family members, the permutation is used in sponge or duplex constructions with different rates. Table 2 provides an overview of the number of rounds and rate used in the different Ascon variants. Additionally, we list the IsapRk re-keying phase of Isap, though we emphasize that the attacker has limited control over the processed 128-bit input hash value $Y$ and cannot directly observe the output nor collisions therein. We omit the IsapMac authentication/hashing and IsapEnc encryption/keystream phases, as they share the parameters of Ascon-Hash and Ascon-128 data processing, respectively.

Each round $p$ of the permutation consists of two main layers, the substitution layer $p_S$ followed by the linear layer $p_L$, as well as a round constant addition $p_C$ at the beginning of each round. The round constant addition Xors an 8-bit constant to state word $x_2$. Since it plays no further role for us, we refer to the design for a full specification [DEMS21b].

**The substitution layer** $p_S$ applies a 5-bit S-box $\mathcal{S}$ in parallel 64 times in a bitsliced fashion across the state words $x_0, \ldots, x_4$, i.e., to each column in Figure 1. The circuit representation of the S-box is illustrated in Figure 2a. We refer to the design document [DEMS21b] for alternative representations, including the lookup table.

**The linear diffusion layer** $p_L$ provides diffusion within each 64-bit word $x_i$. It applies the linear function $\Sigma_i(x_i) = x_i'$ in Figure 2b to word $x_i$, i.e., row $i$ in Figure 1.



**(a)** Ascon's 5-bit S-box $\mathcal{S}(x)$

$$\Sigma_0(x_0) = x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28) \to x_0'$$
$$\Sigma_1(x_1) = x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39) \to x_1'$$
$$\Sigma_2(x_2) = x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6) \to x_2'$$
$$\Sigma_3(x_3) = x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17) \to x_3'$$
$$\Sigma_4(x_4) = x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41) \to x_4'$$

**(b)** Ascon's linear layer with 64-bit functions $\Sigma_i(x_i)$

**Figure 2:** Ascon's substitution layer and linear diffusion layer.

## 2.2 Bounds for Differential and Linear Cryptanalysis

**Differential Cryptanalysis.** Biham and Shamir's differential cryptanalysis [BS90] is one of the main two statistical cryptanalysis techniques. It considers pairs of related inputs to a primitive, i.e., similar input message blocks $a$ and $a'$ to a block cipher $E_K$ with a fixed Xor difference $\alpha = a \oplus a'$. This difference is then tracked throughout the cipher to the output difference $\beta = b \oplus b' = E_K(a) \oplus E_K(a')$ based on a *differential characteristic*, which tracks the difference after each round and holds with a certain probability (averaged over all inputs and keys, assuming independent round keys). The probability depends on the *active S-boxes*, i.e., those S-boxes whose input has a nonzero difference in the characteristic: it is the product of all S-box transition probabilities, which are 1 for inactive S-boxes and at most the S-box's maximum differential probability (MDP) for active S-boxes. The *differential distribution table* (DDT) lists the transition probability $\mathbb{P}[\alpha \overset{\mathcal{S}}{\to} \beta] = \mathbb{P}[\mathcal{S}(x) \oplus \mathcal{S}(x \oplus \alpha) = \beta]$ (or number of solutions) for the differential transition of each input difference $\alpha$ to each output difference $\beta$ of the S-box. The true probability of a block cipher differential, $\mathbb{P}[\alpha \overset{E_K}{\to} \beta]$ for a fixed secret key $K$, may vary due to (1) trail clustering, i.e., multiple characteristics contributing to the same differential, and (2) dependencies between rounds due to the fixed key and key schedule. The cost of a differential attack is inversely proportional to this differential probability.

Ascon's 5-bit S-box has an MDP of $2^{-2}$: 5 of its 32 input differences permit such high-probability transitions; conversely, 20 of its output differences can be reached with the MDP. Additionally, the S-box has a differential branch number of 3, i.e., there is no valid transition from an input difference with a single active bit to an output difference with a single active bit, so it contributes to the diffusion. Ascon's linear layer has a branch number of $\mathcal{B} = 4$. Its diffusion in the backward direction is much stronger than forward.

**Linear Cryptanalysis.** Matsui's linear cryptanalysis [Mat93] is the second important statistical attack and shares many concepts with differential cryptanalysis. It considers *masks* $\alpha$ for inputs and $\beta$ for outputs instead of differences, and evaluates the *bias* $\varepsilon = \mathbb{P}[\alpha \cdot a = \beta \cdot b] - \frac{1}{2}$, or equivalently, the squared correlation $c^2 = (2\varepsilon)^2$, where $\alpha \cdot a$ denotes the inner product of vectors $\alpha$ and $a$. The squared correlation of a characteristic is the product of the squared correlation of its S-boxes. Similarly, the true squared correlation of the linear hull over the entire cipher may differ from the product due to multiple characteristics: the *linear hull effect*. The cost of a linear attack is inversely proportional to this squared correlation of the linear hull. Ascon's properties are very similar between differential and linear cryptanalysis: Its S-box has a maximum squared correlation of $2^{-2}$ and a branch number of 3; its linear layer has a branch number of $\mathcal{B} = 4$.

**Automated Tools.** To argue resistance against differential and linear analysis, designers aim to lower-bound the minimum number of (differentially or linearly) active S-boxes in any characteristic, thus upper-bounding the maximum probability or squared correlation of the characteristic. For some strongly aligned designs such as AES, bounds on the minimum number of active S-boxes can be determined using pen-and-paper arguments based on the wide-trail design strategy [DR01]. Where this is not possible, automated solvers can be used to exhaust the search space of valid characteristics and thus prove lower bounds. The most popular types of general-purpose solvers include Boolean satisfiability (SAT) or Satisfiability Modulo Theories (SMT), Mixed-Integer Linear Programming (MILP) [BFL10, MWGP11, WW11, ZHWW20, SHW+14], and the very general category of Constraint Programming (CP) [SGL+17, ENP19]. Each of these strategies has its advantages; for example, Boolean satisfiability is well-suited for bitwise descriptions of differential behaviour, but is less convenient for counting and bounding the probability. MILP is based on integers and floating-point numbers and thus perfect for counting, but inconvenient for binary circuits.

**The Boolean Satisfiability Problem (SAT).** Boolean satisfiability is defined by a Boolean formula, consisting of a set of variables $X = \{x_1, x_2, \ldots, x_n\}$ combined via the operators $O = \{\wedge, \vee, \neg\}$ (And, Or, Not). The problem is satisfiable if and only if there exists a truth assignment to each variable such that the Boolean formula evaluates to true ($\top$). Modern (pure) SAT solvers operate on a problem description in *Conjunctive Normal Form* (CNF). Many frameworks, such as the STP solver for Satisfiability Modulo Theories (SMT), accept a higher-level, more human-readable input language.

In this paper, we focus on pure SAT models, so we derive clauses of form $(x_0 \vee x_1 \vee \ldots x_n)$, with $x_0, \ldots, x_n$ negated or unnegated variables. Most SAT solvers accept input CNFs in the DIMACS format. The number of variables (#V) and clauses (#C) has a significant impact on the solver runtime, so we aim to minimize both.

**SAT for Bounding Differential and Linear Characteristics.** Most previous work related to the automatic search for linear and differential characteristics is based on *Mixed Integer Linear Programming* (MILP) and *Satisfiability Modulo Theories* (SMT) [MP13, Köl15, AK18]. SMT-based models are often also termed 'SAT-based', as SMT solvers often internally run a SAT solver, but the modelling language is different.

There are few pure SAT results with model descriptions written directly as CNF, since this can be cumbersome to work with, but also very efficient to solve. First attempts with SAT-based cryptanalysis, such as *logical cryptanalysis* [MM00], focus on a direct encoding of the algorithm to solve search problems such as key recovery. With *CryptoSAT*, Lafitte [Laf18] provides a framework to describe cryptographic problems in SAT, offering a higher-level crypto-oriented interface. More recently, several authors have applied pure SAT models to differential cryptanalysis. Efficient, pure SAT encodings of several commonly used operations are described by Sun et al. [SWW21], including the encoding of additional bounding conditions as already proposed by Matsui [Mat94] for other search strategies. The resulting models are used to provide bounds for GIFT, SIMON and SPECK. Furthermore, Sun et al. [SWW18] use the efficient SAT models to derive differentials for LED64 and Midori64.

## 2.3 Previous Bounds for the Ascon Permutation

Several works have investigated the security of Ascon with automated tools and solvers. The known bounds and the best known characteristics are summarized in Table 1.

For bounds against differential cryptanalysis, Dobraunig et al. [DEMS15] proposed a MILP as well as an SMT model and used the latter to prove a minimum of 15 differentially active S-boxes for 3 rounds. Additionally, they used a custom heuristic tool to find characteristics for 4 and 5 rounds, which indicate a significant increase in the number of active S-boxes starting at 4 rounds [DEMS15], as well as constrained characteristics suitable for AEAD forgeries. The corresponding characteristics are illustrated in the NIST design document [DEMS21a]. Gerault et al. [GPT21] propose a CP model and provide a 5-round characteristic with slightly improved probability, as well as characteristics for AEAD forgeries. Udovenko [Udo21] showed how to model the differential distribution table using MILP. For linear cryptanalysis, the minimum number of linearly active S-boxes for 3 rounds is 13 [DEMS15]. Dobraunig et al. [DEM15b] introduced a heuristic search tool and found linear characteristics for up to 5 rounds, showing a roughly similar behaviour as in the differential case.

Automated tools were also used for additional properties, including CP for the bit-based division property [GD21] and MILP for the 3-subset bit-based division property [RHSS21].

# 3 SAT Model of Characteristics and Counters

Building on recent developments in the field of Boolean satisfiability (SAT) solvers and the respective encoding of cryptographic problem instances [SWW21], we derive efficient models representing the propagation of linear and differential trails through the Ascon permutation, to provide improved bounds on the minimum number of active S-boxes. For our purposes, we optimize the models to distinguish if a differential or linear trail with up to $S$ active S-boxes exists. Compared to the model by Sun et al. [SWW21], we describe and evaluate different counter approaches.

## 3.1 SAT Model of Ascon's Differential and Linear Characteristics

**Modeling the Internal State** The Ascon permutation updates an internal state of five 64-bit words (320 bits) $S = x_0 \parallel x_1 \parallel x_2 \parallel x_3 \parallel x_4$ by applying a substitution layer $p_S$ and linear diffusion layer $p_L$ over a total of $R$ rounds. In terms of the SAT model, each round $r \in \{1, \ldots, R\}$ requires two sets $x_S^{(r)}, x_L^{(r)}$ of 320 Boolean variables, with $x_L^{(r)} = p_S(x_S^{(r)})$ and $x_S^{(r+1)} = p_L(x_L^{(r)})$. Here, $p_S$ and $p_L$ denote the action of the respective operation on the difference or linear mask modelled by the 320-bit variable. Each $p_S$ additionally requires 64 variables to represent if an S-box is active in a given round.

Furthermore, we observe that the output of $p_L$ in the penultimate round defines the active S-boxes in the last round, and we can therefore omit the variables and encoding of the last round. A similar argument holds for the active S-boxes in the initial round, where the input to the first $p_L$ defines the active S-boxes in the first round. An encoding of $R$ rounds therefore requires $R \cdot (64 + 320) - 320$ variables for the state representations.

**Modeling the Substitution Layer.** $p_S$ requires deriving a set of clauses limiting the truth assignments to the state variables with respect to the *Differential Distribution Table* (DDT) or the *Linear Approximation Table* (LAT) of the S-box. In this section, we follow the approach of [SWW21] to derive a differential model of the Ascon S-box.

Given the input difference $(a_0, \ldots, a_4)$ and output difference $(b_0, \ldots, b_4)$ of Ascon's S-box as Boolean variables and $w$ as the representation of the differential weight, a sound differential model inhibits invalid assignments to $a_i$, $b_i$ and $w$ with respect to the DDT. In this model, we only count the number $s$ of active S-boxes rather than the actual probability or bias, so we consider only transitions with weight $w = s$ either 0 (for the 0 → 0 transition) or 1 (for any valid transition). Let $\alpha = (\alpha_0, \ldots, \alpha_4)$ and $\beta = (\beta_0, \ldots, \beta_4)$ represent a specific input and output difference pair in the DDT and $\sigma \in \{0, 1\}$ its differential activity (weight). Using all combinations of $(\alpha_0, \ldots, \alpha_4, \beta_0, \ldots, \beta_4, \sigma)$, the set of all invalid truth assignments can be represented as

$$\left\{ (\alpha_0, \ldots, \alpha_4, \beta_0, \ldots, \beta_4, \sigma) \in \mathbb{F}_2^{11} \;\middle|\; \begin{array}{l} P(\alpha \to \beta) = 0 \text{ or} \\ P(\alpha \to \beta) = 1 \wedge \sigma = 1 \text{ or} \\ P(\alpha \to \beta) > 0 \wedge \sigma = 0 \end{array} \right\}.$$

To build a differential model of the S-box, for each invalid assignment, we derive a clause of the form

$$\bigvee_{i=0}^{4} (a_i \oplus \alpha_i) \vee \bigvee_{i=0}^{4} (b_i \oplus \beta_i) \vee (s \oplus \sigma).$$

Hereby, each clause requires the truth assignment of at least one variable to diverge from an invalid assignment, with the complete set of clauses enforcing that no invalid assignments are possible.

The encoding represents a linear model of the S-box by replacing the DDT with the LAT and modifying the conditions for the invalid assignments accordingly. Furthermore, we can extend the idea to a probabilistic model using multiple variables to represent the weight of an S-box [SWW21]. We use the logical minimization tool *Espresso* to reduce the set of invalid combinations. In our case, this reduces the number of clauses per S-box to 64 in the differential case (4096 per round) and 90 in the linear case (5760 per round).

**Modeling the Permutation Layer.** $p_L$ requires the encoding of a 3-input bitwise Xor (Xor3) operation $a_0 = a_1 \oplus a_2 \oplus a_3$ with respect to the individual rotation constants. The differential model of the linear function is identical to the linear function itself. We follow [SWW21] to derive a model of the Ascon linear layer.

Given $(a_0, \ldots, a_3)$ as the Boolean variables representing the input and output of an Xor3 operation, there are $2^4$ possible truth assignments. By the definition of the Xor3 operation, a truth assignment $(\alpha_0, \ldots, \alpha_3)$ is invalid if $\sum_{i=0}^{3} \alpha_i \neq 0 \pmod 2$. Therefore, the model of an Xor3 is defined by a clause

$$\bigvee_{i=0}^{3} a_i \oplus \alpha_i$$

for each of the 8 invalid assignments where $\sum_{i=0}^{3} \alpha_i \neq 0 \pmod 2$. Thus, a full differential model of $p_L$ is given by encoding 64 Xor3 equations for each 64-bit word with respect to the rotation constants. This results in $5 \cdot 64 \cdot 8 = 2560$ clauses per round.

The linear model is almost identical: for a linear operation written as a matrix $b = L \cdot a$, a linear mask $(\alpha, \beta)$ is valid if and only if $\alpha = L^{\top} \cdot \beta$. Since the Ascon matrix $L$ for $\Sigma_j$ in word $j$ is zero except for 3 diagonals corresponding to the 3 rotation constants of word $j$, the transposed matrix has exactly the same format, but with the negated rotation constants, and can thus also be modelled with one Xor3 per bit.

## 3.2 Counting

Modeling a SAT-based distinguisher for differential and linear trails over $R$ rounds requires encoding a bound $B$ on the maximum number of active S-boxes. More specifically, it must hold that

$$\sum_{r=1}^{R} \sum_{i=0}^{63} s_i^{(r)} \leq B, \tag{1}$$

with $s_i^r$ representing whether the $i$-th S-box in round $r$ is active (1) or not (0). Encoding a limit on the maximum number of active variables in a given set is commonly referred to as a cardinality constraint [BCN+21]. To show a lower bound of at least $B + 1$ active S-boxes over $R$ rounds, we need to prove that a model of the Ascon primitive bounded to $\leq B$ active S-boxes is unsatisfiable. For a tight bound, we would additionally need to show satisfiability when bounded to $\leq B + 1$.

Sun et al. [SWW21] propose the use of an encoding based on a *Sequential Counter* [Sin05]. We evaluate the performance of the *Sequential Counter* encoding, compared to alternatives based on *Sorting Networks* [Bat68, ANOC09] and *Totalizers* [MJML14, OLH+13, BB03], using implementations by the *PySAT* project [IMMS18]. Table 3 summarizes our results.

We observe that the performance of the different encodings depends on the satisfiability and complexity of the general problem. The *Sequential Counter* performs well for satisfiable and unsatisfiable instances with low bounds, while requiring a large number of auxiliary variables and clauses to encode larger bounds, potentially increasing the overall search space and therefore the runtime. The results indicate that alternative can encodings perform significantly better on satisfiable instances with larger bounds. In the case of the Ascon permutation, it is hard to evaluate the performance for an unsatisfiable problem with large bounds in a reasonable runtime (inconclusive experiments with a runtime $> 20$

**Table 3:** Runtime comparison of cardinality encodings of models with #C clauses and #V variables for $R$-round differential characteristics with $\#S \leq B$ active S-boxes. *Sequential Ctr* is our implementation of [SWW21], other encodings are from *PySAT* [IMMS18]. Results were acquired with SAT solver *Kissat* [BFFH20] on an *Intel i9-9900K @ 3.60GHz*.

| Encoding | $R=3, B=14$: UNSAT | | | $R=3, B=15$: SAT | | | $R=4, B=44$: SAT | | |
|---|---|---|---|---|---|---|---|---|---|
| | #C | #V | Time | #C | #V | Time | #C | #V | Time |
| Seq. Ctr. [Sin05] | 15510 | 4146 | 2:28:11 | 15891 | 4337 | 0:00.17 | 39292 | 13396 | 18:12:32 |
| Sorting Net [Bat68] | 21504 | 9151 | 3:01:37 | 21504 | 9151 | 0:00.01 | 28159 | 9854 | 2:00:37 |
| Card. Net [ANOC09] | 13838 | 4040 | 2:48:00 | 13838 | 4040 | 0:01.42 | 25462 | 8056 | 0:15:42 |
| Totalizer [BB03] | 29794 | 2944 | 2:05:43 | 29794 | 2944 | 0:00.29 | 51330 | 4224 | 0:52.49 |
| mTotalizer [OLH+13] | 14514 | 2419 | 2:59:09 | 14513 | 2419 | 0:00.12 | 23826 | 3520 | 1:38:59 |
| kmTotalizer [MJML14] | 12296 | 2233 | 3:44:02 | 12452 | 2248 | 0:01.84 | 20338 | 3219 | 1:04:16 |

days). We conclude that different encodings can significantly impact the runtime depending on the problem instance. For a dedicated analysis with our target problem, we refer to Table 7 in Section 5.

# 4 Search Strategy and Partitioning

In this section, we discuss the limitations of the straightforward SAT model and propose a new search strategy based on partitioning the search space efficiently. This allows us to eliminate redundant parts of the search space.

## 4.1 Parallelizing the Search

The main issue with finding bounds for weakly aligned ciphers with medium-sized or large states such as Ascon is their large search space, even for few rounds. Furthermore, the search complexity increases dramatically from round to round, making it difficult to estimate the runtime for target coordinates. For the case of 4-round Ascon, we obtained no results for bounds of at most 43 differentially active S-boxes after running the model for up to 40 days. This refers to wall-clock runtime in a single-threaded solver – still the "standard" setup for most SAT solvers. Clearly, good parallelization would significantly increase the problem space explorable with reasonable wall-clock runtimes. There are two basic approaches to parallelization in SAT solving: Either the parallelization is done by the solver and thus happens transparently for the user; or the user manually partitions the search and starts solvers in parallel for the subproblems.

**Solver-based Parallelization.** The two general approaches to parallelized SAT solving are *Cube-and-Conquer* and *Portfolio* solvers.

**Cube-and-Conquer** solvers use the heuristics of *Lookahead* solvers to efficiently partition a problem instance into many independent subproblems (Cubes). Heuristics determine at which point branches within the search tree of the *Lookahead* solver are cut off, generating a reduced instance of the problem with partially assigned variables. Multiple *Conflict Driven Clause Learning* (CDCL) solvers can then run in parallel solving the individual cubes [HKWB11, HvM09, SLM09]. We evaluate the performance of the *Cube and Conquer* solver *Paracooba* in terms of runtime, using *March* for the cube generation and multiple *CaDiCal* instances as the CDCL solvers [HFB20]. We observe that running 6, 8, or 12 solver instances in default configuration, solving 370 cubes, does not improve performance as shown in Table 4a. With this small degree of parallelization, the search actually performs worse than single-threaded solvers like *Kissat*, a close relative of CaDiCal. The reason is that in all cases, a single cube dominates the overall runtime, while the other cubes are trivial, thus not balancing well. We conclude that the approach only provides a significant performance improvement if a problem can be partitioned into many equally hard subproblems. Further work is required to evaluate good cube generation possibilities for the encoding of the Ascon permutation and cryptographic problems in general.

The **Portfolio** approach achieves scalability by providing an interface to efficiently exchange learned clauses for a set of CDCL solver instances. We evaluate the implementation of *Mallob* [SS21] based on *HordeSAT* [BSS15], providing interfaces to several different solvers. They instantiate a set of diversified solver instances using different configurations, variable polarities, and random seeds to learn different sets of clauses. The clauses can be exchanged between the different instances efficiently using the *Message Passing Interface* (MPI) [GLDS96], accelerating learning new clauses. [SS21] present performance data for highly scalable environments with up to 2500 CPUs. We observe that for smaller numbers of solver instances, the runtime is very dependent on the diversifications, as shown in Table 4b. We conclude that the approach needs further evaluation and optimization within a highly scalable environment to get conclusive performance results and useful improvements for cryptographic problems.

**Table 4:** Runtime of the parallelized SAT solvers *Paracooba* [HFB20] and *Mallob* [SS21] on the 3-round problem with at most 14 differentially active S-boxes, proven UNSAT by *Kissat* in 31 minutes. All results acquired running on an *Intel i7-8750H @ 2.20GHz*.

**(a)** *Paracooba* with #T *CaDiCal* instances, 370 cubes

| #T | Time |
|---|---|
| 6 | 1:10:00 |
| 8 | 1:08:23 |
| 12 | 1:00:00 |

**(b)** *Mallob* with #T *Lingeling* instances.

| #T | Time |
|---|---|
| 8 | 1:54:45 |
| 12 | 1:00:34 |
| 16 | 1:08:15 |

**Manual Parallelization by Partitioning.** Since the automated tool-based parallelization does not achieve the desired reduction in wall-clock runtime for our application, we can consider a manual parallelization approach. By partitioning the search space into partitions that cover the entire original search space and showing that each of the partition problems is unsatisfiable, we can prove that the original problem is unsatisfiable, as discussed next. This can be considered a manual, optimized variant of the cube-and-conquer approach.

## 4.2 Partitioning the Search Space using Girdle Patterns

Due to the less-than-optimal parallelization gain with the solver-based approach, we instead focus on a manual approach. The most important advantage of this approach is that we can partition in a way that exploits symmetries in the cipher definition to eliminate redundant, equivalent parts of the search space by design.

We partition the search space of all possible characteristics for round-reduced Ascon based on S-box activity patterns. Unlike classical pattern-based approaches [OMA95], we do not just restrict the number of active S-boxes in each round. Instead, we partition characteristics based on their *precise S-box activity pattern in their round with the fewest active S-boxes*. We refer to this pattern in the round with the fewest active S-boxes as the *girdle pattern* of the characteristic, and to its Hamming weight as the *girdle weight* $S$.

For example, the best known differential 4-round characteristic for Ascon with 44 active S-boxes [DEMS15, DEMS21a] is illustrated in Figure 3 and would be classified by its second-round girdle pattern 8040000000040000 with 3 active S-boxes in bit positions 63, 54, 18, or equivalently by its third-round girdle pattern with the same weight. Note that in this representation, each 320-bit difference $\alpha = (\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4)$ in the round input state is represented by a 64-bit pattern $\sigma$ of S-boxes, where $\sigma_i = \alpha_{0,i} \vee \alpha_{1,i} \vee \alpha_{2,i} \vee \alpha_{3,i} \vee \alpha_{4,i}$ for each bit position $0 \leq i < 64$.

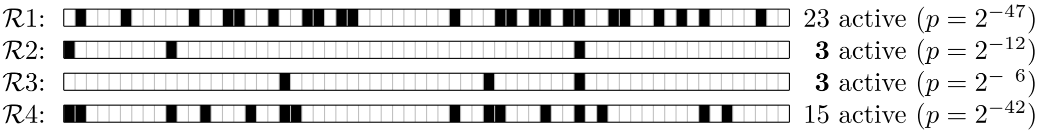

$\mathcal{R}1$: 23 active ($p = 2^{-47}$)
$\mathcal{R}2$: **3** active ($p = 2^{-12}$)
$\mathcal{R}3$: **3** active ($p = 2^{-6}$)
$\mathcal{R}4$: 15 active ($p = 2^{-42}$)

**Figure 3:** Best known differential 4-round characteristic with 44 active S-boxes [DEMS21a].

For a characteristic with $\leq B$ active S-boxes for $R$ rounds, the maximum possible girdle weight is $S_{\max} = \lfloor B/R \rfloor$. Thus, the total number of potential girdle patterns is

$$\#\mathcal{G} = \sum_{S=1}^{\lfloor B/R \rfloor} R \times \binom{64}{S}.$$

For $R = 4$ and $B = 31$ (for proving $p_{\max} \leq 2^{-64}$), this already gives more than $2^{31.3}$ potential girdle patterns; for $B = 35$ (for proving $p_{\max} \leq 2^{-72}$), $2^{34.3}$ girdle patterns.

We can now start a separate SAT task for each possible girdle pattern (in each possible round) where the girdled round is restricted to exactly this pattern, whereas all other rounds are restricted to at least $S$ active S-boxes, where $S$ is the girdle weight. Together, these tasks cover the entire search space, i.e., each candidate $R$-round characteristic with at most $B$ active S-boxes.

**Pre-Filtering Girdle Patterns.** To speed up the search time per girdle pattern, when aiming for an $R$-round bound, we can first test each pattern for $R' = R - 1$ rounds. Only for those few patterns that pass the much faster $R'$-round test, i.e., where an $R'$-round characteristic with at most $B'$ active S-boxes exists, we start the slower $R$-round test. Here, we can take advantage of the girdle pattern definition, which requires that each round has $\geq S$ active S-boxes. Thus, we can test each pattern with girdle weight $S$ first for $R' = R - 1$ rounds with bound $B' = B - S$.

Then, the original task with bound $B$ for a girdle pattern in round $r$ of $R$ is only started if it succeded in both round $r$ and round $r - 1$ of $R'$ (if applicable, i.e., in the first round only $r$ and in the last round only $r - 1$ is relevant).

This is faster for three reasons: (a) the $R'$-round search space is much smaller; (b) the smaller bound $B'$ additionally reduces the cost of counting; and (c) the necessary number of tasks is $R'$ rather than $R$. Additionally, different tradeoffs are possible here, depending on the relative cost and number of $R'$-round tasks compared to $R$-round tasks. We refer to Subsection 5.1 for more details and an example.

## 4.3 Exploiting Rotational Symmetries in the Search Space

**Rotational Symmetry and Necklaces.** We observe that differential and linear characteristics for Ascon are *rotation-invariant*, in the sense that rotating each word of a characteristic by a fixed number of bit positions yields another equivalent valid characteristic with the same number of active S-boxes and probability. This implies that girdle patterns are similarly rotation-invariant, and allows us to significantly reduce the number of considered patterns in our partitioning. In particular, we only need to consider one representative pattern among all up to 64 equivalent, rotated variants.

In combinatorial terms, instead of fixed patterns, we are considering *necklaces* with 64 beads in 2 colors [Mor72]. An example for 4-bead necklaces, corresponding to a toy cipher with 4-bit words, is illustrated in Figure 4.

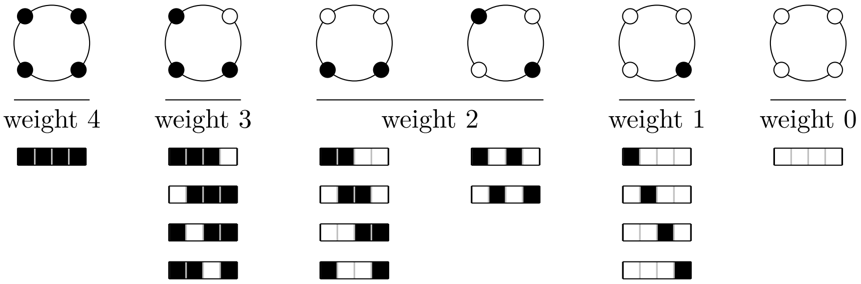


**Figure 4:** Necklaces with $n = 4$ beads in 2 colors, each corresponding to up to $n$ equivalent patterns of active S-boxes for $n$-bit words; fewer in case of rotation-symmetric patterns.

**Pólya Counting and Enumerating Necklaces.** Based on the Redfield-Pólya theorem [Red27, Pól37], the number of $k$-ary necklaces of length $n$ is

$$N_k(n) = \frac{1}{n} \sum_{d|n} \varphi(d) \cdot k^{\frac{n}{d}} \ ,$$

where $\varphi$ is the Euler totient function and the sum iterates over all divisors $d$ of $n$. The number of 2-ary necklaces of length $n$ with density (weight) $w$ is

$$\frac{1}{n} \sum_{j|\gcd(n,w)} \varphi(j) \binom{n/j}{d/j} \ .$$

These necklaces, including necklaces with fixed density, can be enumerated efficiently with amortized cost $\mathcal{O}(1)$ [FK86, SR99].

We provide an overview of the number of 64-bead necklaces with fixed weight in Table 5. The total number of necklaces with 64 beads in 2 colors is $2^{58} + 2^{26} + 2^{11} + 2^{4} + 2^{2}$ and thus very close to $2^{64}/64$, indicating that almost all necklaces correspond to exactly 64 equivalent patterns. Thus, by enumerating girdle patterns based on necklaces, we reduce the search space by a factor of $2^6$.

**Table 5:** Number of 64-bead necklaces with fixed weight. The numbers for weights $33, \dots, 64$ equal weights $31, \dots, 0$. The total number of 64-bead 2-color necklaces is $2^{58.0}$.

| Weight | Number of patterns | Weight | Number | Weight | Number | Weight | Number |
|---|---|---|---|---|---|---|---|
| 1 | $1 = 2^{0.00}$ | 9 | $2^{28.68}$ | 17 | $2^{44.29}$ | 25 | $2^{52.48}$ |
| 2 | $32 = 2^{5.00}$ | 10 | $2^{31.14}$ | 18 | $2^{45.68}$ | 26 | $2^{53.06}$ |
| 3 | $651 = 2^{9.35}$ | 11 | $2^{33.44}$ | 19 | $2^{46.95}$ | 27 | $2^{53.55}$ |
| 4 | $9\,936 = 2^{13.28}$ | 12 | $2^{35.58}$ | 20 | $2^{48.12}$ | 28 | $2^{53.96}$ |
| 5 | $119\,133 = 2^{16.86}$ | 13 | $2^{37.58}$ | 21 | $2^{49.19}$ | 29 | $2^{54.27}$ |
| 6 | $1\,171\,552 = 2^{20.16}$ | 14 | $2^{39.44}$ | 22 | $2^{50.16}$ | 30 | $2^{54.49}$ |
| 7 | $9\,706\,503 = 2^{23.21}$ | 15 | $2^{41.18}$ | 23 | $2^{51.03}$ | 31 | $2^{54.62}$ |
| 8 | $69\,159\,400 = 2^{26.04}$ | 16 | $2^{42.80}$ | 24 | $2^{51.80}$ | 32 | $2^{54.67}$ |

# 5 Bounds for 4-Round Ascon

In this section, we apply the ideas of Section 3 and Section 4 to obtain a parallelizable, symmetry-aware model of 4-round Ascon. We first summarize the overall strategy and then discuss further optimizations to decrease the overall runtime.

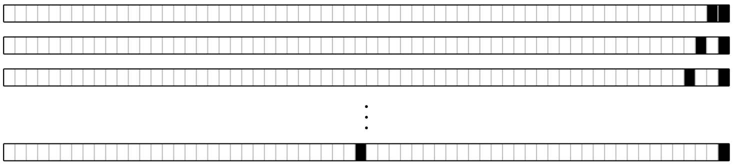
## 5.1 Optimized Strategy for 4 Rounds

For 4 rounds, the best known result is a differential characteristic with 44 active S-boxes and probability $2^{-107}$ as well as a linear characteristic with 43 active S-boxes and squared correlation $2^{-98}$, both obtained with heuristic tools [DEMS15, DEM15b]. On the other hand, no bounds have been proven so far; thus, the only currently known bound is by combining those for $1+3$ rounds, resulting in a total of $1+15 = 16$ differentially or $1+13 = 14$ linearly active S-boxes. In our pattern-based strategy of Section 4, the target bound $B$ has a strong impact on the overall runtime, as it defines the maximum girdle weight of $S \leq \lfloor B/R \rfloor$ for $R$ rounds and thus the number of patterns to enumerate. We thus focus on proving $\geq B$ active S-boxes where $B$ is a multiple of $R = 4$, which is equivalent to the unsatisfiability of $B-1$ S-boxes and a maximum girdle weight of $\lfloor B/R \rfloor - 1$. Within our computational budget, the maximum bound we can cover is 36 active S-boxes with $S \leq 8$; we refer to Subsection 5.4 for a runtime estimate for bound 40 with $S \leq 9$.

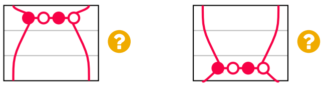
Using the girdle necklace patterns of Section 4, we partition the search into subproblems as follows to prove $\geq 36$ differentially or linearly active S-boxes over 4 rounds of Ascon:

1. **Girdle weight $S$:** For a 4-round characteristic with less than 36 active S-boxes, the girdle weight can be at most $\lfloor 35/4 \rfloor = 8$, so we consider weights $S \in \{1, 2, \ldots, 8\}$.

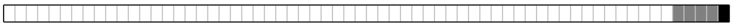
2. **Girdle necklace patterns:** These girdle weights correspond to a total of more than $2^{26}$ necklaces of weight $\leq 8$ (see Table 5). For example, the 32 necklace patterns representing $\binom{64}{2} = 2016$ girdle patterns for $S = 2$ are



3. **Pooled pre-filtering of patterns based on 3 rounds:** Before testing each pattern for bound $B$ over 4 rounds, we perform a faster pre-filtering test for bound $B - S$ over 3 rounds. We only test each pattern for the first or last round (see next step):

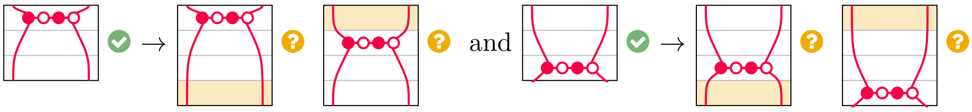


To reduce the overhead further, we do not test each pattern individually, but combine them in small pools of 4 patterns in one SAT task. Here, we combine patterns that share most active bit positions and only differ in a few. For example, we can combine the first 4 patterns for $S = 2$ and require that exactly 1 of the 4 gray bits is active:



4. **Girdle rounds:** Taking only the successful patterns for 3 rounds with the reduced bound, we start the test for 4 rounds. Since the number of surviving patterns is very

small (see Subsection 5.2), we slightly adapt the strategy described at the end of Subsection 4.2: Instead of testing the pattern for each of the 3 rounds and then only starting the 4-round test for round $r$ if both $r$ and $r-1$ were successful for 3 rounds, we trade more 4-round tests for fewer 3-round tests and only test every second round $r$ out of the 3 rounds, taking successful results for round $r$ and $r+1$ out of 4 rounds:



## 5.2 Results for 4 Rounds

We successfully proved the differential and linear models UNSAT for $R = 4$ and $B = 35$, thus proving a maximum probability or squared correlation for characteristics of $\leq 2^{-72}$. The total runtime for the differential model was over 600 CPU days executed in half a week of wall-clock time on up to 176 Intel Xeon cores. The linear model has a similar runtime. Table 6 provides a runtime overview. The runtime is dominated by the round-reduced search for $R' = 3$, since only very few patterns satisfy the reduced bound of $B' = B - S$.

**Table 6:** Runtime overview for the $R' = 3$-round tests over all girdle patterns of weight $S$ in round 1 or 3 with $\leq B'$ active S-boxes, $P = 4$. #T is the number of CPU cores used in parallel for the *Kissat* solver, mostly *Intel Xeon E5-2669* or *E5-4669 v4 @ 2.20GHz*.

| $S$ | $B'$ | #T | Differential | | | Linear | | |
|---|---|---|---|---|---|---|---|---|
| | | | $t_{\max}$ | $t_{\min}$ | $\bar{t}$ | $t_{\max}$ | $t_{\min}$ | $\bar{t}$ |
| 1 | 34 | 1 | 00:00:01 | 00:00:01 | 00:00:01 | 00:00:01 | 00:00:01 | 00:00:01 |
| 2 | 33 | 1 | 00:00:28 | 00:00:28 | 00:00:28 | 00:00:07 | 00:00:07 | 00:00:07 |
| 3 | 32 | 4 | 00:06:06 | 00:05:53 | 00:06:00 | 00:01:06 | 00:00:52 | 00:00:57 |
| 4 | 31 | 24 | 00:09:27 | 00:07:40 | 00:08:34 | 00:04:46 | 00:03:31 | 00:04:23 |
| 5 | 30 | 72 | 00:26:09 | 00:21:42 | 00:26:09 | 00:21:44 | 00:12:23 | 00:19:23 |
| 6 | 29 | 72 | 03:39:16 | 03:12:08 | 03:22:10 | 03:30:45 | 02:52:25 | 03:11:25 |
| 7 | 28 | 176 | 11:21:01 | 09:35:43 | 10:24:31 | 11:06:31 | 05:55:59 | 09:25:24 |
| 8 | 27 | 176 | 64:31:08 | 61:13:50 | 62:26:36 | 62:43:03 | 59:15:14 | 60:31:35 |

**Counting model and pool size.** In order to reduce the number of solver invocations, we combine multiple girdle patterns into pools of size $P$. Each pool encodes a problem instance which is satisfiable if at least one of the girdle patterns is part of a valid characteristic for the given bound. We encode a given pool by building a truth table of size $2^x$, where $x$ is the number of differing S-box activities across the given patterns. Similar to the encoding of the permutation layer in Subsection 3.1, we can exclude invalid assignments by adding complementary clauses while fixing the truth assignments for S-boxes with equal activity. In Table 7, we summarize the performance of the pooling approach using different counter implementations, showing that pooling several solutions reduces the runtime. While our final runtimes in Table 6 are based on pools of size 4 using the sequential counter, the totalizer-based encodings might reduce the runtime further.

**Filtered results for 3 rounds.** In Table 8, we give an overview of the number of surviving patterns after the filter for $R' = 3$ rounds with bound $B' = B - S$. We note that in our experiments, characteristics are not only counted when the round with the girdle pattern is the actual girdle round with the fewest S-boxes, but also when there is another round with fewer S-boxes, marked in Table 8 as *All*. We also included the count of actual surviving girdle patterns marked as *Girdle*. The numbers are very similar for the differential case (8464 in total with 6083 girdle patterns) and linear case (8112 in total with 6161 girdle patterns). Clearly, differential characteristics favor a lower weight in the first round, while linear characteristics favor a lower weight in the last round. This is to be expected due to the linear layer, which has a much stronger diffusion in backward than in forward direction.

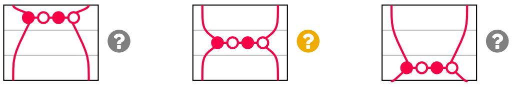
## 5.3 Strategy and Results for 6 Rounds

We can build on the 3-round and 4-round results to derive bounds for 6 rounds. With our patterns up to weight $S \leq 8$, we can prove at least $6 \cdot 9 = 54$ active S-boxes for 6 rounds. For this purpose, we can reuse the small set of filtered patterns for $R' = 3$ rounds, according to similar arguments as in Subsection 4.2 – another big advantage of the partitioning and pre-filtering approach. However, we need to extend the current set of patterns for $r \in \{1, 3\}$ to also cover $r = 2$. A 6-round characteristic with $< 54$ active S-boxes would need $< 27$ either in its first 3 or its last 3 rounds and thus a girdle weight of $\leq 8$ in that half, so it must have appeared among our pre-filtered candidate list. A preliminary test of some patterns for 6 rounds shows runtimes of less than 30 minutes, in some cases only seconds, so we can expect a feasible runtime.

In summary, using the girdle necklace patterns of Section 4, we partition the search as follows to prove $\geq 54$ differentially or linearly active S-boxes over 6 rounds of Ascon:

1. **Girdle weight $S$ and necklace patterns:** For a 6-round characteristic with less than 54 active S-boxes, the girdle weight can be at most $\lfloor 53/6 \rfloor = 8$, so we consider weights $S \in \{1, 2, \ldots, 8\}$. The girdle necklace patterns are the same as for 4 rounds.

2. **Pooled pre-filtering of patterns based on 3 rounds:** Either the first 3 or the last 3 of the 6 rounds must contain $< 27$ S-boxes and a girdle pattern. We thus re-use our previous 3-round patterns, but add a test for $r = 2$ to the existing $r \in \{1, 3\}$:



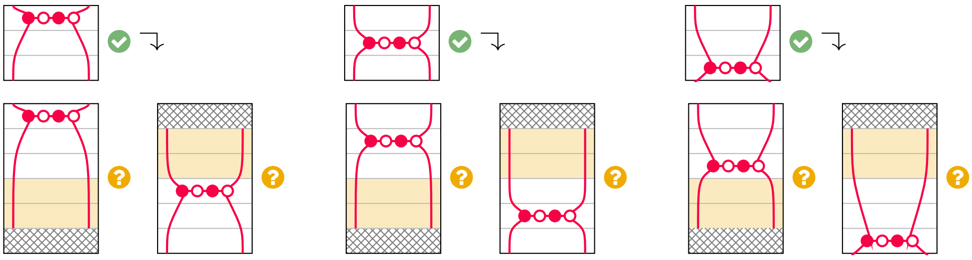
3. **Girdle rounds:** We take each successful pattern for 3 rounds with the reduced bound of $< 27$ active S-boxes as a candidate for either the last or first 3 out of 6 rounds. Instead of testing the full 6 rounds with bound 54, we test only 5 rounds with bound 53, which is a necessary condition. If no such 5-round solution exists, then any 6-round characteristic has at least 54 active S-boxes:

**Table 7:** Runtime comparison of cardinality encodings for $R' = 3$ rounds with $\leq 27$ active S-boxes, fixing the S-box activity in round $r \in \{1, 3\}$ to one of $P$ different pooled necklaces of weight 8, as described in Section 4. Results with *Kissat* on an *Intel E5-4669 @ 2.20GHz*.

| Encoding | Pool size $P = 1$ | | Pool size $P = 4$ | |
|---|---|---|---|---|
| | Time ($r = 1$) | Time ($r = 3$) | Time ($r = 1$) | Time ($r = 3$) |
| Seq. Ctr. [Sin05] | 1:14:08 | 1:18:40 | 0:58:04 | 0:37:55 |
| Sort. Net [Bat68] | 1:30:24 | 1:26:50 | 1:19:47 | 0:43:25 |
| Card. Net [ANOC09] | 1:10:25 | 1:13:30 | 0:57:31 | 0:37:53 |
| Totalizer [BB03] | 0:52:46 | 1:10:36 | 0:43:12 | 0:34:10 |
| mTotalizer [OLH+13] | 0:48:28 | 1:23:08 | 0:39:45 | 0:38:20 |
| kmTotalizer [MJML14] | 0:47:13 | 1:26:55 | 0:38:37 | 0:44:39 |

**Table 8:** Overview of filtered results after 3 rounds for the differential and linear model with girdle weight $S \leq 8$ in round $r \in \{1, 2, 3\}$.

| $S$ | Differential | | | | | | Linear | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $r = 1$ | | $r = 2$ | | $r = 3$ | | $r = 1$ | | $r = 2$ | | $r = 3$ | |
| | All | Girdle | All | Girdle | All | Girdle | All | Girdle | All | Girdle | All | Girdle |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 2 | 32 | 32 | 6 | 6 | 0 | 0 | 0 | 0 | 5 | 5 | 32 | 32 |
| 3 | 574 | 574 | 50 | 48 | 1 | 1 | 2 | 2 | 51 | 49 | 603 | 603 |
| 4 | 1388 | 1388 | 283 | 272 | 0 | 0 | 0 | 0 | 257 | 248 | 1293 | 1293 |
| 5 | 1175 | 1175 | 1037 | 753 | 1 | 1 | 7 | 3 | 867 | 603 | 1084 | 1084 |
| 6 | 660 | 627 | 1822 | 737 | 3 | 2 | 19 | 16 | 1687 | 815 | 650 | 615 |
| 7 | 258 | 156 | 901 | 219 | 20 | 15 | 43 | 29 | 861 | 340 | 303 | 208 |
| 8 | 86 | 16 | 106 | 17 | 60 | 43 | 80 | 42 | 154 | 38 | 113 | 35 |
| Total | 4174 | 3969 | 4205 | 2052 | 85 | 62 | 151 | 92 | 3882 | 2098 | 4079 | 3871 |



**Results.** Table 8 already lists the number of girdle patterns with weight $S \leq 8$ in $r \in \{1, 2, 3\}$ that permit 3-round characteristics with $B' = B - S = 35 - S \geq 27$ active S-boxes. Thus, either the first 3 or last 3 rounds of any 6-round characteristic with $< 54$ active S-boxes must appear in this list.

We denote the check of any pattern (either fixed for the first or second half) as an individual task. To avoid unbalanced work distribution, the tasks are not pre-batched as for the 4-round experiment, but rather each task is scheduled individually. Once a core has finished its task, it will be assigned the next one from the queue. This ensures that all CPU cores are used within a margin of the maximum runtime of an individual task.

The totel runtime for all 5-round checks was about 2 CPU months each for the differential and linear checks, where the most expensive tasks are those with the pattern in round $r = 1$ in the differential case and $r = 5$ in the linear case (almost 2 CPU hours per pattern on average), while the cheapest tasks are in $r = 4$ for the differential and $r = 2$ for the linear case (about 20 seconds per pattern on average). All 5-round tasks are UNSAT, so 6-round characteristics must have at least 54 active S-boxes or a probability $\leq 2^{-108}$. We expect that this bound is by far not tight.

## 5.4 Extending the Bounds

**Higher bounds.** The bound of at least 36 active S-boxes, corresponding to a differential probability or squared correlation of at most $2^{-72}$, is likely not tight. The best known solution has 44 differentially or 43 linearly active S-boxes. Based on the considerations in Subsection 5.1, the next interesting bound for our model is 40 active S-boxes, based on a maximum girdle weight of $S \leq 9$. Based on Table 5, this number is more than 6 times higher than for $S \leq 8$. Additionally, the counter circuit is more complex and the runtime per subproblem is also higher. We evaluated the runtime based on $2^{18}$ patterns with Totalizer [BB03] encoding and estimate that this bound takes $38 \cdot 176 = 6688$ CPU days (perfectly parallelizable) to prove, which is feasible but beyond our computational budget. Of course, in case the bound is actually satisfiable, the runtime may be lower.

**Constrained characteristics.** Referring to Table 2, when the Ascon permutation is used in any of the keyed or unkeyed schemes of the Ascon family, the attacker has only very limited access to the input and output state. They can only introduce differences through the 64 or 128 bits of the outer part, corresponding to the first one or two state words for the different variants, or via the nonce in initialization in the two bottom words. For example, a 64-bit rate limits the possible transitions in the first S-box to either $\texttt{00} \to \texttt{00}$ or $\texttt{10} \to \{\texttt{09}, \texttt{0b}, \texttt{18}, \texttt{1a}\}$ and thus further restricts the search space. However, these constraints are only of limited use for improving the runtime of our model and it is thus not clear whether they can be used to achieve higher bounds. One advantage when using the results for a 6-round bound could be the potentially smaller number of surviving solutions.

**Probability.** As we expect that our current bounds are not tight, we did not evaluate the more expensive probabilistic model to bound the maximum probability instead of the number of active S-boxes. In the currently best known characteristics, there is a significant gap between the S-box-bound and the probability bound; for example, for 3 rounds, the best known characteristic has 15 active S-boxes and a probability of $2^{-40}$ [DEMS15], which we can prove to be optimal using the probabilistic version of the SAT model by Sun et al. [SWW21].

**Other cipher designs.** Our partitioning approach is generally useful for primitives with large states and weak alignment. Such designs are growing increasingly popular and include the permutations of prominent lightweight sponge-based designs (e.g., Xoodyak and Keccak/Ketje/Keyak), LS-designs as proposed for side-channel resistence (e.g., Robin/Fantomas), or recent designs like the low-latency cipher Speedy. Most of these are also rotation-invariant and can thus apply the necklace technique; for higher-dimensional state layouts, this could be represented using multi-color necklaces.

## 5.5 Implications for the Ascon AEAD and Hashing Family

Our bounds of $\geq 36$ active S-boxes for 4 rounds and $\geq 54$ for 6 rounds imply at least 72 active S-boxes for 8 rounds and 108 for 12 rounds. This is more than enough to conclude the resistance of the initialization, data processing, and finalization phases against differential and linear attacks for most keyed and unkeyed modes based on Table 2.

**Ascon-128 and Ascon-128a.** Potential differential attack vectors for these authenticated ciphers include introducing a difference via the nonce $N$ (e.g., to determine the key based on the partially observed output difference after the initialization) or via the message blocks (e.g., to cancel the difference with the next block and produce a collision-based forgery, or to predict the partial output difference after the finalization for a difference-based forgery). The initialization and finalization with our bound of $\geq 108$ active S-boxes and a probability $\leq 2^{-216}$ for their 12 rounds provide ample security margin for 128-bit security against these attack vectors. For the message processing phase, our bounds imply $\geq 54$ active S-boxes or probability $\leq 2^{-108}$ for the 6 rounds in Ascon-128 and $\geq 72$ or probability $\leq 2^{-144}$ for the 8 rounds of Ascon-128a. This means there are no characteristics that could be exploited within the message limit of at most $2^{64}$ blocks per key. We emphasize again that the 6-round bound is almost certainly not tight; we expect no collision-producing characteristics with probability $\geq 2^{-128}$.

**Ascon-Hash and Ascon-Xof.** Unkeyed modes are difficult to evaluate based on probabilities, since the attacker sees the internal state and can manipulate message blocks to satisfy some of the differential conditions directly. For this reason, they might potentially be able to find solutions for characteristics with probability $< 2^{-128}$ with sufficiently low computational complexity to create collisions faster than brute force. As an approximation, we can assume that an attacker can use one degree of freedom (i.e., one input bit in the outer part) to deterministically satisfy a differential condition with probability $2^{-1}$. Note that this is a pessimistic simplified assumption – likely, the attacker will not be able to fully use all degrees of freedom in practice, as this is more difficult in sponges than, say, in Davies-Meyer Merkle-Damgård hash functions like MD5 [WY05], SHA-1 [SBK+17], and SHA-2 [MNS13, DEM15a]. Under this assumption, an attacker could use the freedom of the 64-bit rate plus $< 2^{128}$ probabilistic tries to find a message that satisfies a characteristic of probability $> 2^{-192}$. Our bound of at least 108 active S-boxes for 12 rounds, while almost certainly very far from tight, already implies a probability of $\leq 2^{-216}$, which gives confidence in the resistance of Ascon-Hash and Ascon-Xof against this attack vector. Currently, the best known attacks only cover up to 2 rounds [ZDW19, GPT21].

# 6 Conclusion

Gaining trust based on provable bounds as well as heuristic results is important for the 10 finalists in the NIST Lightweight Cryptography standardization project. Like several of the finalists, Ascon is based on a large, weakly aligned permutation, which contributes to its lightweight implementation cost and is well-suited for both authenticated encryption and hashing. On the downside, proving bounds for such designs is more difficult than for smaller AES-like block cipher designs and requires substantial effort to optimize the model or proof strategy. For this reason, so far, there have been no satisfactory bounds to support the security of Ascon against differential and linear attacks – instead, this trust is primarily built on results from custom heuristic search tools, which suggest a much better resistance than could be proven to date.

In this paper, we proved that any single differential or linear characteristic over 4 rounds has at least 36 active S-boxes (compared to a weak bound of only 16 that could be derived based on previous results). This implies at least 54 active S-boxes for 6 rounds (with our extended model, compared to previous bounds of 30 differentially or 26 linearly), 72 for 8 rounds, and 108 for 12 rounds – more than enough to conclude the resistance of the initialization, data processing, and finalization phases against differential and linear attacks for the keyed and unkeyed modes.

To achieve this result, we proposed a new search strategy of dividing the search space into a large number of subproblems based on 'girdle patterns', and show how to exploit the rotational symmetry of Ascon using necklace theory to reduce the search space drastically. One of the advantages of this approach is the predictable total runtime, which permits to precisely evaluate the impact of different variations and optimize the model accordingly. Among others, we evaluated different variants of integer counters and observed that the best choice depends strongly on the target sum. We also showed that optimizations like pre-filtering and pooling reduce the overall runtime.

We emphasize that these bounds are almost certainly not tight. Furthermore, an attacker is limited to constrained characteristics with potentially even higher bounds in the sponge and duplex constructions of the Ascon family. Still, even these non-tight bounds are sufficient to support trust in the used permutation variants. The usual caveats for provable bounds with respect to potential trail clustering and independence assumptions apply, but there is currently no indication that these significantly impact the results. For the hashing modes, a detailed evaluation of the unkeyed setting could help shed light on the question to which extent an attacker can exploit message-modification-style techniques based on the few degrees of freedom available with the 64-bit rate.

## Acknowledgments

Additional funding was provided by a generous gift from Google. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding parties.

## References

[AK18] Ralph Ankele and Stefan Kölbl. Mind the gap – A closer look at the security of block ciphers against differential cryptanalysis. In Carlos Cid and Michael J. Jacobson Jr., editors, *Selected Areas in Cryptography – SAC 2018*, volume 11349 of *LNCS*, pages 163–190. Springer, 2018. doi:10.1007/978-3-030-10970-7_8.

[ANOC09] Roberto Asin, Robert Nieuwenhuis, Albert Oliveras, and Enric Rodriguez Carbonell. Cardinality networks and their applications. In Oliver Kullmann, editor, *Theory and Applications of Satisfiability Testing – SAT 2009*, volume 5584 of *LNCS*, pages 167–180. Springer, 2009. doi:10.1007/978-3-642-02777-2_18.

[Bat68] Kenneth E. Batcher. Sorting networks and their applications. In *American Federation of Information Processing Societies – AFIPS 1968*, volume 32 of *AFIPS Conference Proceedings*, pages 307–314. Thomson Book Company, Washington D.C., 1968. doi:10.1145/1468075.1468121.

[BB03] Olivier Bailleux and Yacine Boufkhad. Efficient CNF encoding of Boolean cardinality constraints. In Francesca Rossi, editor, *Principles and Practice of Constraint Programming – CP 2003*, volume 2833 of *LNCS*, pages 108–122. Springer, 2003. doi:10.1007/978-3-540-45193-8_8.

[BCN$^+$21] Miquel Bofill, Jordi Coll, Peter Nightingale, Josep Suy, Felix Ulrich-Oltean, and Mateu Villaret. SAT encodings for pseudo-Boolean constraints together with at-most-one constraints. *CoRR*, abs/2110.08068, 2021. URL: https://arxiv.org/abs/2110.08068, arXiv:2110.08068.

[BDPV07] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge functions. Ecrypt Hash Workshop 2007, 2007. URL: https://keccak.team/files/SpongeFunctions.pdf.

[BDPV08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 181–197. Springer, 2008. doi:10.1007/978-3-540-78967-3_11.

[BDPV11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the sponge: Single-pass authenticated encryption and other applications. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography – SAC 2011*, volume 7118 of *LNCS*, pages 320–337. Springer, 2011. doi:10.1007/978-3-642-28496-0_19.

[BFFH20] Armin Biere, Katalin Fazekas, Mathias Fleury, and Maximillian Heisinger. CaDiCaL, Kissat, Paracooba, Plingeling and Treengeling entering the SAT

Competition 2020. In Tomas Balyo, Nils Froleyks, Marijn Heule, Markus Iser, Matti Järvisalo, and Martin Suda, editors, *SAT Competition 2020 – Solver and Benchmark Descriptions*, volume B-2020-1 of *Department of Computer Science Report Series B*, pages 51–53. University of Helsinki, 2020.

[BFL10] Charles Bouillaguet, Pierre-Alain Fouque, and Gaëtan Leurent. Security analysis of SIMD. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography – SAC 2010*, volume 6544 of *LNCS*, pages 351–368. Springer, 2010. doi:10.1007/978-3-642-19574-7_24.

[BS90] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology – CRYPTO '90*, volume 537 of *LNCS*, pages 2–21. Springer, 1990. doi:10.1007/3-540-38424-3_1.

[BSS15] Tomás Balyo, Peter Sanders, and Carsten Sinz. HordeSat: A massively parallel portfolio. In Marijn Heule and Sean A. Weaver, editors, *Theory and Applications of Satisfiability Testing – SAT 2015*, volume 9340 of *LNCS*, pages 156–172. Springer, 2015. doi:10.1007/978-3-319-24318-4_12.

[CAE14] CAESAR committee. CAESAR: Competition for authenticated encryption: Security, applicability, and robustness, 2014. URL: https://competitions.cr.yp.to/caesar-submissions.html.

[DEM15a] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Analysis of SHA-512/224 and SHA-512/256. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015*, volume 9453 of *LNCS*, pages 612–630. Springer, 2015. doi:10.1007/978-3-662-48800-3_25.

[DEM15b] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Heuristic tool for linear cryptanalysis with applications to CAESAR candidates. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015*, volume 9453 of *LNCS*, pages 490–509. Springer, 2015. doi:10.1007/978-3-662-48800-3_20.

[DEM16] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Cryptanalysis of Simpira v1. In Roberto Avanzi and Howard M. Heys, editors, *Selected Areas in Cryptography – SAC 2016*, volume 10532 of *LNCS*, pages 284–298. Springer, 2016. doi:10.1007/978-3-319-69453-5_16.

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

[DEM+21] Christoph Dobraunig, Maria Eichlseder, Stefan Mangard, Florian Mendel, Bart Mennink, Robert Primas, and Thomas Unterluggauer. ISAP. Submission as a Finalist to the NIST Lightweight Crypto Standardization Process, 2021. URL: https://csrc.nist.gov/Projects/lightweight-cryptography/finalists.

[DEMS15] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Cryptanalysis of Ascon. In Kaisa Nyberg, editor, *Topics in Cryptology – CT-RSA 2015*, volume 9048 of *LNCS*, pages 371–387. Springer, 2015. doi:10.1007/978-3-319-16715-2_20.

[DEMS16] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2. Submission to CAESAR: Competition for Authenticated Encryption. Security, Applicability, and Robustness (Round 3 and "First Choice" for lightweight authenticated encryption in the final portfolio), 2016. URL: http://competitions.cr.yp.to/round3/asconv12.pdf.

[DEMS21a] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon. Submission as a Finalist to the NIST Lightweight Crypto Standardization Process, 2021. URL: https://csrc.nist.gov/Projects/lightweight-cryptography/finalists.

[DEMS21b] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *Journal of Cryptology*, 34(3):33, 2021. doi:10.1007/s00145-021-09398-9.

[DR01] Joan Daemen and Vincent Rijmen. The wide trail design strategy. In Bahram Honary, editor, *Cryptography and Coding – IMACC 2001*, volume 2260 of *LNCS*, pages 222–238. Springer, 2001. doi:10.1007/3-540-45325-3_20.

[ENP19] Maria Eichlseder, Marcel Nageler, and Robert Primas. Analyzing the linear keystream biases in AEGIS. *IACR Transactions on Symmetric Cryptology*, 2019(4):348–368, 2019. doi:10.13154/tosc.v2019.i4.348-368.

[FK86] Harold Fredricksen and Irving J. Kessler. An algorithm for generating necklaces of beads in two colors. *Discret. Math.*, 61(2-3):181–188, 1986. doi:10.1016/0012-365X(86)90089-0.

[GD21] Shibam Ghosh and Orr Dunkelman. Automatic search for bit-based division property. In Patrick Longa and Carla Ràfols, editors, *Progress in Cryptology – LATINCRYPT 2021*, volume 12912 of *LNCS*, pages 254–274. Springer, 2021. doi:10.1007/978-3-030-88238-9_13.

[GLDS96] William Gropp, Ewing L. Lusk, Nathan E. Doss, and Anthony Skjellum. A high-performance, portable implementation of the MPI message passing interface standard. *Parallel Comput.*, 22(6):789–828, 1996. doi:10.1016/0167-8191(96)00024-5.

[GPT21] David Gérault, Thomas Peyrin, and Quan Quan Tan. Exploring differential-based distinguishers and forgeries for ASCON. *IACR Transactions on Symmetric Cryptology*, 2021(3):102–136, 2021. doi:10.46586/tosc.v2021.i3.102-136.

[HFB20] Maximilian Heisinger, Mathias Fleury, and Armin Biere. Distributed cube and conquer with Paracooba. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020*, volume 12178 of *LNCS*, pages 114–122. Springer, 2020. doi:10.1007/978-3-030-51825-7_9.

[HKWB11] Marijn Heule, Oliver Kullmann, Siert Wieringa, and Armin Biere. Cube and conquer: Guiding CDCL SAT solvers by lookaheads. In Kerstin Eder, João Lourenço, and Onn Shehory, editors, *Hardware and Software: Verification and Testing Conference – HVC 2011*, volume 7261 of *LNCS*, pages 50–65. Springer, 2011. doi:10.1007/978-3-642-34188-5_8.

[HvM09] Marijn Heule and Hans van Maaren. Look-ahead based SAT solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 155–184. IOS Press, 2009. doi:10.3233/978-1-58603-929-5-155.

[IMMS18] Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *SAT*, pages 428–437, 2018. URL: https://doi.org/10.1007/978-3-319-94144-8_26, doi:10.1007/978-3-319-94144-8_26.

[Köl15] Stefan Kölbl. CryptoSMT: An easy to use tool for cryptanalysis of symmetric primitives, 2015. URL: https://github.com/kste/cryptosmt.

[Laf18] Frédéric Lafitte. CryptoSAT: a tool for SAT-based cryptanalysis. *IET Inf. Secur.*, 12(6):463–474, 2018. doi:10.1049/iet-ifs.2017.0176.

[Mat93] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *LNCS*, pages 386–397. Springer, 1993. doi:10.1007/3-540-48285-7_33.

[Mat94] Mitsuru Matsui. On correlation between the order of S-boxes and the strength of DES. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT 1994*, volume 950 of *LNCS*, pages 366–375. Springer, 1994. doi:10.1007/BFb0053451.

[MJML14] Ruben Martins, Saurabh Joshi, Vasco M. Manquinho, and Inês Lynce. Incremental cardinality constraints for MaxSAT. CoRR abs/1408.4628, 2014. URL: http://arxiv.org/abs/1408.4628.

[MM00] Fabio Massacci and Laura Marraro. Logical cryptanalysis as a SAT problem. *J. Autom. Reason.*, 24(1/2):165–203, 2000. doi:10.1023/A:1006326723002.

[MNS11] Florian Mendel, Tomislav Nad, and Martin Schläffer. Finding SHA-2 characteristics: Searching through a minefield of contradictions. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 288–307. Springer, 2011. doi:10.1007/978-3-642-25385-0_16.

[MNS13] Florian Mendel, Tomislav Nad, and Martin Schläffer. Improving local collisions: New attacks on reduced SHA-256. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 262–278. Springer, 2013. doi:10.1007/978-3-642-38348-9_16.

[Mor72] C. Moreau. Sur les permutations circulaires distinctes. *Nouvelles annales de mathématiques : journal des candidats aux écoles polytechnique et normale*, 2e série, 11:309–314, 1872. URL: http://www.numdam.org/item/NAM_1872_2_11__309_0/.

[MP13] Nicky Mouha and Bart Preneel. Towards finding optimal differential characteristics for ARX: Application to Salsa20. IACR Cryptology ePrint Archive, Report 2013/328, 2013. URL: https://eprint.iacr.org/2013/328.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology – Inscrypt 2011*, volume 7537 of *LNCS*, pages 57–76. Springer, 2011. doi:10.1007/978-3-642-34704-7_5.

[Nat18] National Institute of Standards and Technology. Submission requirements and evaluation criteria for the lightweight cryptography standardization process, 2018. URL: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf.

[OLH+13] Toru Ogawa, Yangyang Liu, Ryuzo Hasegawa, Miyuki Koshimura, and Hiroshi Fujita. Modulo based CNF encoding of cardinality constraints and its application to MaxSAT solvers. In *International Conference on Tools with Artificial Intelligence – ICTAI 2013*, pages 9–17. IEEE Computer Society, 2013. doi:10.1109/ICTAI.2013.13.

[OMA95] Kazuo Ohta, Shiho Moriai, and Kazumaro Aoki. Improving the search algorithm for the best linear expression. In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO '95*, volume 963 of *LNCS*, pages 157–170. Springer, 1995. doi:10.1007/3-540-44750-4_13.

[Pól37] George Pólya. Kombinatorische Anzahlbestimmungen für Gruppen, Graphen und chemische Verbindungen. *Acta Mathematica*, 68:145–254, 1937. doi:10.1007/BF02546665.

[Red27] J. Howard Redfield. The theory of group-reduced distributions. *American Journal of Mathematics*, 49(3):433–455, 1927. doi:10.2307/2370675.

[RHSS21] Raghvendra Rohit, Kai Hu, Sumanta Sarkar, and Siwei Sun. Misuse-free key-recovery and distinguishing attacks on 7-round Ascon. *IACR Transactions on Symmetric Cryptology*, 2021(1):130–155, 2021. doi:10.46586/tosc.v2021.i1.130-155.

[SBK+17] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. The first collision for full SHA-1. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, volume 10401 of *LNCS*, pages 570–596. Springer, 2017. doi:10.1007/978-3-319-63688-7_19.

[SGL+17] Siwei Sun, David Gérault, Pascal Lafourcade, Qianqian Yang, Yosuke Todo, Kexin Qiao, and Lei Hu. Analysis of AES, SKINNY, and others with constraint programming. *IACR Transactions on Symmetric Cryptology*, 2017(1):281–306, 2017. doi:10.13154/tosc.v2017.i1.281-306.

[SHW+14] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014*, volume 8873 of *LNCS*, pages 158–178. Springer, 2014. doi:10.1007/978-3-662-45611-8_9.

[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*, volume 3709 of *LNCS*, pages 827–831. Springer, 2005. doi:10.1007/11564751_73.

[SLM09] João P. Marques Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 131–153. IOS Press, 2009. doi:10.3233/978-1-58603-929-5-131.

[SR99] Joe Sawada and Frank Ruskey. An efficient algorithm for generating necklaces with fixed density. In Robert Endre Tarjan and Tandy J. Warnow, editors, *Symposium on Discrete Algorithms – SODA '99*, pages 752–758. ACM/SIAM, 1999. URL: http://dl.acm.org/citation.cfm?id=314500.314910.

[SS21] Dominik Schreiber and Peter Sanders. Scalable SAT solving in the cloud. In Chu-Min Li and Felip Manyà, editors, *Theory and Applications of Satisfiability Testing – SAT 2021*, volume 12831 of *LNCS*, pages 518–534. Springer, 2021. doi:10.1007/978-3-030-80223-3_35.

[SWW18] Ling Sun, Wei Wang, and Meiqin Wang. More accurate differential properties of LED64 and Midori64. *IACR Transactions on Symmetric Cryptology*, 2018(3):93–123, 2018. doi:10.13154/tosc.v2018.i3.93-123.

[SWW21] Ling Sun, Wei Wang, and Meiqin Wang. Accelerating the search of differential and linear characteristics with the SAT method. *IACR Transactions on Symmetric Cryptology*, 2021(1):269–315, 2021. doi:10.46586/tosc.v2021.i1.269-315.

[Udo21] Aleksei Udovenko. MILP modeling of Boolean functions by minimum number of inequalities. IACR Cryptology ePrint Archive, Report 2021/1099, 2021. URL: https://eprint.iacr.org/2021/1099.

[WW11] Shengbao Wu and Mingsheng Wang. Security evaluation against differential cryptanalysis for block cipher structures. IACR Cryptology ePrint Archive, Report 2011/551, 2011. URL: https://eprint.iacr.org/2011/551.

[WY05] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 19–35. Springer, 2005. doi:10.1007/11426639_2.

[ZDW19] Rui Zong, Xiaoyang Dong, and Xiaoyun Wang. Collision attacks on round-reduced Gimli-Hash/Ascon-Xof/Ascon-Hash. IACR Cryptology ePrint Archive, Report 2019/1115, 2019. URL: https://eprint.iacr.org/2019/1115.

[ZHWW20] Hongluan Zhao, Guoyong Han, Letian Wang, and Wen Wang. MILP-based differential cryptanalysis on round-reduced Midori64. *IEEE Access*, 8:95888–95896, 2020. doi:10.1109/ACCESS.2020.2995795.