Cryptanalysis of Ascon

Christoph Dobraunig[1], Maria Eichlseder[1], Florian Mendel[1], and Martin Schläffer[2]

[1] IAIK, Graz University of Technology, Austria
firstname.lastname@iaik.tugraz.at
[2] Infineon Technologies AG, Austria
martin.schlaeffer@gmail.com

Abstract. We present a detailed security analysis of the CAESAR candidate Ascon. Amongst others, cube-like, differential and linear cryptanalysis are used to evaluate the security of Ascon. Our results are practical key-recovery attacks on round-reduced versions of Ascon-128, where the initialization is reduced to 5 out of 12 rounds. Theoretical key-recovery attacks are possible for up to 6 rounds of initialization. Moreover, we present a practical forgery attack for 3 rounds of the finalization, a theoretical forgery attack for 4 rounds finalization and zero-sum distinguishers for the full 12-round Ascon permutation. Besides, we present the first results regarding linear cryptanalysis of Ascon, improve upon the results of the designers regarding differential cryptanalysis, and prove bounds on the minimum number of (linearly and differentially) active S-boxes for the Ascon permutation.

Keywords: Ascon, CAESAR initiative, cryptanalysis, authenticated encryption

1 Introduction

The CAESAR competition [20] is an ongoing cryptographic competition, where numerous authenticated encryption schemes are challenging each other with the goal of finding a portfolio of ciphers, suitable for different use-cases. Currently, more than 45 ciphers are still participating in the competition. In the near future, this portfolio will be further reduced to focus the attention of the crypto community on a few candidates. Therefore, analyzing the security of the candidate ciphers is of great importance to enable the committee to judge them adequately.

Ascon is a submission by Dobraunig et al. [11] to the CAESAR competition. In the submission document, the designers discuss the design rationale for the cipher and give first cryptanalytic results, in particular on the differential properties of the Ascon permutation. Since the cipher was only recently presented, results of external cryptanalysis are scarce so far. Jovanovic et al. [15] prove the security of Ascon's mode of operation under idealness assumptions for the permutation.

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-16715-2_20

Our contribution. We present a detailed security analysis of the CAESAR candidate Ascon-128. Based on the low algebraic degree of Ascon, we are able to construct a zero-sum distinguisher with complexity 2^{130} for the full 12-round Ascon permutation in Section 3. In Section 4, we use similar algebraic properties to construct a distinguisher based on cube testers. We also use cube-like techniques to obtain a key-recovery attack for a round-reduced version of Ascon with 5-round initialization with practical complexity. Theoretical key-recovery attacks are possible for up to 6 rounds of initialization. Moreover, in Section 5, we present the first results on linear cryptanalysis, and improve the results by the designers on differential cryptanalysis. Our results include linear and differential characteristics obtained with heuristic search, as well as a computer-aided proof of security bounds against linear and differential cryptanalysis (minimum number of active S-boxes). Using our results on linear-differential analysis, we present a practical forgery attack for 3 rounds of the finalization and a theoretical forgery attack for 4-round finalization. Our results are summarized in Table 1.

Table 1. Results for Ascon-128. Attacks performed on the initialization or finalization.

type	rounds	time	method	source
permutation distinguisher	12 / 12	2^{130}	zero-sum	Section 3
key recovery	6 / 12	2^{66}	cube-like	Section 4.4
	5 / 12	2^{35}		
	5 / 12	2^{36}	differential-linear	Section 5.4
	4 / 12	2^{18}		
forgery	4 / 12	2^{101}	differential	Section 5.3
	3 / 12	2^{33}		

2 Ascon

Ascon is a submission by Dobraunig et al. [11] to the ongoing CAESAR competition. It is based on a sponge-like construction with a state size of 320 bits (consisting of five 64-bit words $x_0, \ldots, x_4$). Ascon comes in two flavors, Ascon-128 and Ascon-96, with different security levels and parameters, as summarized in Table 2. The analysis in this paper is focused on Ascon-128. In the following, we give a brief overview about the mode of operation and the permutation of Ascon. For a complete description, we refer to the design document [11].

Mode of operation. Ascon's mode of operation is based on MonkeyDuplex [8]. As illustrated in Fig. 1, the encryption is partitioned into four phases: initialization, processing associated data, processing the plaintext, and finalization. Those phases use two different permutations p^a and p^b. The stronger

Table 2. Parameters for Ascon [11].

name	bit size of					rounds	
	key	nonce	tag	data block		p^a	p^b
Ascon-128	128	128	128	64		12	6
Ascon-96	96	96	96	128		12	8

variant p^a is used for initialization and finalization, while p^b is used in the data processing phases.

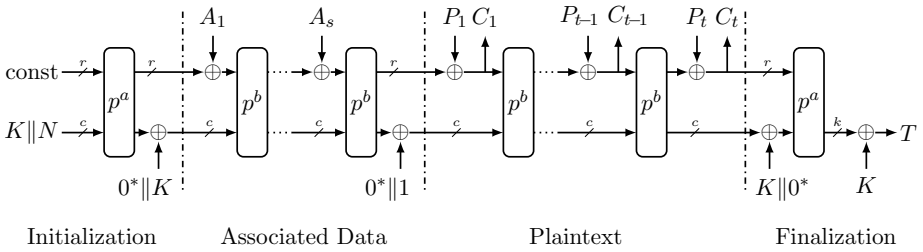


Fig. 1. The encryption of Ascon [11].

The initialization takes as input the secret key K and the public nonce N. The initialization ensures that we start with a random-looking state at the beginning of the data procession phase for every new nonce. In the subsequent processing of the associated data, r-bit blocks are absorbed by xoring them to the state, separated by invocations of p^b. If no associated data needs to be processed, the whole phase can be omitted. Plaintext is processed in r-bit blocks in a similar manner, with ciphertext blocks extracted from the state right after adding the plaintext. For domain separation between associated data and plaintext, a constant is xored to the secret part of the internal state. After all data is processed, the finalization starts and the k-bit tag T is returned.

Table 3. The S-box of Ascon [11].

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\mathcal{S}(x)$	4	11	31	20	26	21	9	2	27	5	8	18	29	3	6	28
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$\mathcal{S}(x)$	30	19	7	14	0	13	17	24	16	12	1	25	22	10	15	23

Permutation. Ascon uses the two permutations p^a and p^b. Both iteratively apply the same round function p: a rounds for p^a, and b rounds for p^b. The round transformation p consists of a constant addition to x_2, followed by the application of a nonlinear substitution layer and a linear layer.

The substitution layer uses a 5-bit S-box (Table 3), which is affine equivalent to the Keccak [2] χ mapping. The Ascon S-box is applied 64 times in parallel on the state. Each bit of the 5 64-bit words $(x_0, \ldots, x_4)$ contributes one bit to each of the 64 S-boxes, where x_0 always serves as most significant bit.

The linear layer is derived from the Σ-function of SHA-2 [19]. The Σ-function is applied to each of the 5 state-words and uses different rotation values for each word:

$$\begin{aligned}
\Sigma_0(x_0) &= x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28) \\
\Sigma_1(x_1) &= x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39) \\
\Sigma_2(x_2) &= x_2 \oplus (x_2 \ggg \ 1) \oplus (x_2 \ggg \ 6) \\
\Sigma_3(x_3) &= x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17) \\
\Sigma_4(x_4) &= x_4 \oplus (x_4 \ggg \ 7) \oplus (x_4 \ggg 41)
\end{aligned}$$

3 Zero-sum distinguishers

In this section, we apply zero-sum distinguishers used in the analysis of Keccak [1,6,7] to Ascon. Zero-sum distinguishers have been used to show non-ideal properties of round-reduced versions for the Keccak permutation. With the help of zero-sum distinguishers, Boura et al. have have been able to distinguish the full 24-round Keccak permutation from a random permutation. Since the core of the Ascon S-box corresponds to the Keccak S-box, we are able to construct distinguishers for the full 12 rounds (or up to 20 rounds) of the Ascon permutation.

Algebraic model of Ascon. As the name zero-sum distinguishers suggests, we search for a set of inputs and corresponding outputs of an n-bit permutation which sum to zero over $\mathbb{F}_2^n$. To create this set of input-output pairs, we start in the middle of the permutation and compute outwards. Furthermore, we keep a set of $320 - d$ bits constant and vary the other d bits through all possible assignments. Thus, we get 2^d possible intermediate states. For all these 2^d intermediate states, we calculate the respective outputs. If the degree of the function determining the output bits is strictly smaller than d, the resulting outputs will sum to zero over $\mathbb{F}_2^n$ [1,6]. After that, we calculate the input values of the permutation using the 2^d intermediate states. Again, if the degree of the inverse function is smaller than d, the inputs sum to zero over $\mathbb{F}_2^n$. The result is a zero-sum distinguisher, or rather, a family of zero-sum distinguishers.

To apply the technique to Ascon, we have to bound the degree of multiple rounds of the Ascon permutation and its inverse. The algebraic degree of one

Ascon S-box is 2, with respect to $\mathbb{F}_2$, and can be easily determined from its algebraic normal form (ANF):

$$
\begin{aligned}
y_0 &= x_4x_1 + x_3 + x_2x_1 + x_2 + x_1x_0 + x_1 + x_0 \\
y_1 &= x_4 + x_3x_2 + x_3x_1 + x_3 + x_2x_1 + x_2 + x_1 + x_0 \\
y_2 &= x_4x_3 + x_4 + x_2 + x_1 + 1 \\
y_3 &= x_4x_0 + x_4 + x_3x_0 + x_3 + x_2 + x_1 + x_0 \\
y_4 &= x_4x_1 + x_4 + x_3 + x_1x_0 + x_1
\end{aligned}
$$

Here, x_0, x_1, x_2, x_3, x_4, and y_0, y_1, y_2, y_3, y_4 represent the input, and output of an S-box, with x_0/y_0 representing the most significant bit. The S-boxes in one substitution layer are applied in parallel to the state, and the linear layer and constant addition do not increase the algebraic degree. Consequently, the overall degree of one Ascon permutation round is 2, and the degree of r rounds is at most 2^r.

To determine the degree of the inverse permutation, we use the ANF of the inverse Ascon S-box:

$$
\begin{aligned}
x_0 &= y_4y_3y_2 + y_4y_3y_1 + y_4y_3y_0 + y_3y_2y_0 + y_3y_2 + y_3 + y_2 + y_1y_0 + y_1 + 1 \\
x_1 &= y_4y_2y_0 + y_4 + y_3y_2 + y_2y_0 + y_1 + y_0 \\
x_2 &= y_4y_3y_1 + y_4y_3 + y_4y_2y_1 + y_4y_2 + y_3y_1y_0 + y_3y_1 + y_2y_1y_0 \\
&\quad + y_2y_1 + y_2 + 1 + x_1 \\
x_3 &= y_4y_2y_1 + y_4y_2y_0 + y_4y_2 + y_4y_1 + y_4 + y_3 + y_2y_1 + y_2y_0 + y_1 \\
x_4 &= y_4y_3y_2 + y_4y_2y_1 + y_4y_2y_0 + y_4y_2 + y_3y_2y_0 + y_3y_2 + y_3 \\
&\quad + y_2y_1 + y_2y_0 + y_1y_0
\end{aligned}
$$

The algebraic degree of the ANF of the inverse Ascon S-box is 3. Therefore, the degree for an r-round inverse Ascon permutation is at most 3^r.

Basic distinguisher for 12 rounds. To create a zero-sum distinguisher for the 12-round Ascon permutation that is used for the cipher's initialization and finalization, we target the intermediate state after round 5. Thus, we attack 5 backward (inverse) rounds and 7 forward rounds. An upper bound for the degree of the 7-round permutation is $2^7 = 128$, while for the 5 inverse rounds, an upper bound is $3^5 = 243$. So we choose $d = 244$, fix $320 - 244 = 76$ bits of the intermediate state and vary the remaining 244 bits to create a set of 2^{244} intermediate states. For all these states, we calculate 7 rounds forward and 5 rounds backward. The sum of all the resulting input and output values over $\mathbb{F}_2^n$ is zero. A similar attack is possible for $11 = 4+7$ rounds (with $d = \max\{81, 128\} + 1 = 129$) and for $13 = 5 + 8$ rounds (with $d = \max\{243, 256\} + 1 = 257$).

Improvement using Walsh spectrum analysis. The complexity of the 12-round distinguisher can be further improved by analyzing the permutation's

Walsh spectrum and applying the techniques by Boura and Canteaut [6]: If the Walsh spectrum of a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is 2^ℓ-divisible, then for any $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$, we have

$$\deg(G \circ F) \leq n - \ell + \deg(G).$$

As Boura and Canteaut show, the Walsh spectrum of the Keccak S-box is 2^3-divisible. The affine linear preprocessing and postprocessing that the Ascon S-box adds compared to the Keccak S-box does not change this number. The same holds true for the inverse S-box. The Ascon nonlinear layer applies this S-box 64 times in parallel. The Walsh spectrum of a parallel composition is the multiplication of the individual Walsh spectra [6]. Thus, the Walsh spectrum of the complete nonlinear layer is divisible by $2^{3 \cdot 64} = 2^{192}$. Let p denote one round of the Ascon permutation, and p^{-1} its inverse. A closer bound on the degree of 5 rounds of the inverse permutation, p^{-5}, is then obtained by

$$\deg(p^{-5}) = \deg(p^{-4} \circ p^{-1}) \leq 320 - 192 + \deg(p^{-4}) \leq 320 - 192 + 81 = 209.$$

Thus, $d = \max\{209, 128\} + 1 = 210$ is sufficient for $12 = 5 + 7$ rounds of the Ascon permutation.

Adding a free round in the middle. Additionally, as Boura and Canteaut [6] observe, an additional round can be added to the attack (almost) for free as follows: The original attack requires an intermediate state where $n - d$ bits are fixed to a constant, while d bits loop through all possible valuations. Now, we set d to be a multiple of the 5-bit S-box size and furthermore, choose the d variable bits such that they always include complete S-boxes. Then, the inputs (and consequently outputs) of some S-boxes are constant, while the other S-boxes have their inputs (and consequently outputs) loop through all possible values. If we look at the output of the nonlinear layer after this intermediate step, we observe it adheres to the same pattern as the input: $n - d$ bits are fixed and d bits enumerate through all their possible values. We can now use the original intermediate step as the starting point for the backwards rounds, and the output of the nonlinear layer as the starting point for the forward rounds (plus an additional, free linear layer). This way, we can extend the previous attacks by one round each, with the only additional cost of choosing d as a multiple of 5. We get zero-sum distinguishers on 12, 13, and 14 rounds with $d = 130$, 210, and 260, respectively.

More rounds. Finally, the results of Boura et al. [7, Theorem 2] are also directly applicable to our previous results to distinguish up to 20 permutation rounds with $d = 319$ (using 9 backward rounds with degree ≤ 318 and 11 forward rounds with degree ≤ 317, no free middle round possible).

Using a zero-sum distinguisher, we can show non-random properties for the full 12-round permutation of Ascon. However, the designers already state [11]

that the permutation is not ideal and are aware of such distinguishers. The non-ideal properties of the permutation do not seem to affect the security of Ascon. In particular, the complexity of 2^{130} is above the cipher's claimed security level.

4 Cube attacks

Recently, Dinur et al. [9] published various cube and cube-like attacks on several keyed primitives using the Keccak permutation. Those cube-like attacks include cube testers, which can serve as distinguishers, and also cube-like attacks to recover the secret key. In this section, we apply two attacks presented by Dinur et al. [9] to Ascon.

4.1 Brief description of cube attacks

The cube attack is an algebraic attack developed by Dinur and Shamir [10]. This algebraic attack builds on the fact that for most ciphers, each output bit can be represented as a polynomial over $\mathbb{F}_2^n$ in algebraic normal form (ANF). The variables x_i of this polynomial may be single bits of plaintext, key-bits, or constants. Dinur and Shamir made the following observation: If a carefully chosen set of plaintext bits is varied over all possible values and the other bits are kept constant, the sum of one bit of the output (cube sum) might be the result of a linear polynomial (called superpoly) consisting solely of bits of the secret key. By gathering many of these linear polynomials, the secret key can be found.

To perform such a cube attack on a cipher, two things have to be done. First, an attacker has to find such cubes (variables to vary and the resulting linear key relations). This is done in an offline preprocessing phase. Here, the attacker determines the cubes by selecting the cube variables randomly and check if the resulting superpoly is linear and contains the key. This preprocessing phase has to be carried out once for each cipher. In an online phase, the attacker uses the knowledge of the cubes to recover the secret key of his target. To perform the attack, the attacker has to be able to choose the plaintext according to his needs and obtain the corresponding ciphertext outputs.

4.2 Cube attack on Ascon

Now we want to investigate the potential threat of cube attacks to Ascon. If we look at the different phases of Ascon, the only phase where a nonce-respecting adversary can easily keep some inputs of the permutation constant and deterministically influence others is the initialization. In this scenario, the key is kept secret and the attacker has the ability to choose the nonce according to his needs.

As evaluated in Section 3, the degree of a 5-round initialization of Ascon is at most 32. Thus, if we search for cubes of 31 variables, the resulting superpoly is definitely linear or constant. Considering 6 rounds of the initialization, we

have to look for cubes with at most 63 variables, for 7 rounds with at most 127 variables and so on. So it is likely that a practical cube attack on 6 rounds is already hard to achieve. However, we have not searched for cubes, but instead performed cube-like attacks on Ascon to recover the secret key in Section 4.4.

4.3 Distinguishers using cube testers

Below, we describe a cube tester for 6 rounds of the Ascon permutation with the property that the generated output bits sum to zero over $\mathbb{F}_2$. Moreover, this cube tester has a practical complexity of only 2^{33}, although the expected degree for 6 rounds of the Ascon permutation is about 64. To achieve this, we have to take a closer look at the internal structure of Ascon.

The permutation of Ascon starts with the substitution layer. In this layer, the 5-bit S-box is applied 64 times in parallel to the internal state of Ascon. Each of the five 64-bit words of the internal state contributes exactly one bit to each instantiation of a 5-bit S-box. So if all cube variables lie within the same word of the state, they do not appear together in one term after the application of the S-box layer. Hence, after 5 more rounds, at most 32 variables of one state-word appear together in one term. As a consequence, selecting a cube of 33 variables of the same state-word definitely results in an empty superpoly and all 2^{33} generated outputs sum to zero.

This distinguisher can be used to distinguish the key-stream generated by Ascon-128 in a nonce-misuse scenario, where the attacker can keep the nonce constant while varying the plaintext. For Ascon-128, 64-bit blocks of plaintext are xored with the state-word x_0. Thus, the attacker can vary 33 bits of the first plaintext block, while keeping the remaining 31 bits and the bits of a second plaintext block constant. The resulting 2^{33} second ciphertext blocks will sum to zero. However, the designers of Ascon strictly forbid nonce reuse, and no security claims are made for such a scenario.

Similar cube testers can be applied to reduced versions of Ascon with only 6 rounds (instead of 12 rounds) of initialization. Then, an attacker with control over the nonce can observe the first key-stream block. In contrast to the nonce-misuse scenario, attacks on round-reduced versions of Ascon in a nonce-respecting scenario give insight in the expected security of Ascon and are therefore of more value. Next, we will show how to extend the observations made in this section to a key-recovery attack on round-reduced versions of Ascon.

4.4 Key recovery using cube-like attacks

Dinur et al. [9] published a key recovery attack where the superpoly does not necessarily have to be a linear function of the secret key bits, but can also be non-linear. Such attacks are also possible for round-reduced versions of Ascon, with the initialization reduced to 5 or 6 out of 12 rounds. The attack on 5 rounds has practical complexity and has been implemented. We will discuss the working principle of the attack by means of a 5-round version of Ascon-128. For a 6-round initialization, the attack works similarly. The attack itself is divided into

two steps, each with an online and an offline phase, and relies on the following two observations.

Observations. The first observation has already been discussed in the context of cube testers: If all cube variables are located within one state-word, they do not appear in the same term of the output polynomial after one application of the substitution layer.

To discuss the second observation, we have to take a look at the ANF of the S-box and consider the positions of the initial values. During the initialization, the constant C is written to x_0, the first word K_1 of the key to x_1, the second key word K_2 to x_2, the first word N_1 of the nonce to x_3, and the second nonce word N_2 to x_4. We use the ANF of the S-box to get the relations for the state words $x_0, \ldots, x_4$ after the first call of the substitution layer. The index i represents the corresponding bit position of the 64-bit word.

$$
\begin{aligned}
x_0[i] &= N_2[i]K_1[i] + N_1[i] + K_2[i]K_1[i] + K_2[i] + K_1[i]C[i] + K_1[i] + C[i] \\
x_1[i] &= N_2[i] + N_1[i](K_2[i] + K_1[i]) + N_1[i] + K_2[i]K_1[i] + K_2[i] + K_1[i] + C[i] \\
x_2[i] &= N_2[i]N_1[i] + N_2[i] + K_2[i] + K_1[i] + 1 \\
x_3[i] &= N_2[i]C[i] + N_2[i] + N_1[i]C[i] + N_1[i] + K_2[i] + K_1[i] + C[i] \\
x_4[i] &= N_2[i]K_1[i] + N_2[i] + N_1[i] + K_1[i]C[i] + K_1[i]
\end{aligned}
$$

Observe that $N_2[i]$ is only combined nonlinearly with key bit $K_1[i]$, and $N_1[i]$ only with $K_1[i]$ and $K_2[i]$. As demonstrated by Dinur et al. [9], we can make use of this fact to build a so-called borderline cube. For instance, we select $N_2[0..15]$ as our cube variables. The rest of the nonce is kept constant. After round 1, our cube variables only appear with $K_1[0..15]$ in one term and definitely not together with the other bits of the secret key. After 4 more rounds, all of the cube variables may appear together in one term, possibly combined with a selection of the key bits $K_1[0..15]$, but never together with the rest of the key bits. Thus, the cube sum depends on $K_1[0..15]$, but it does not depend on $K_1[16..63]$, or $K_2[0..63]$. This fact leads to the following attack.

Step 1. In the first step, we recover the key-word K_1 in 16-bit chunks. Therefore, we select 4 different borderline cubes with 16 variables in N_2 and probe the online oracle with each of these 4 sets. So we get 4 sums of key-stream blocks, each dependent on 16 different key bits of K_1. In the upcoming offline phase, we use the fact that the sum of the outputs (key-stream blocks) only depends on 16 key bits. So we set the rest of the key bits to a constant and calculate cube sums for every possible 16-bit key part. If such a cube sum corresponds to the cube sum received in the online phase, we get a key candidate. In our experiments, we only received one key candidate per 16-bit block on average. Therefore, we only have one key candidate on average for K_1.

Step 2. In the second step, we recover K_2 in 16-bit chunks. To do so, we use $N_1[i]$ to create our borderline cubes. In contrast to the step before, we

have a dependency of the output on bits of K_1, too. So we have to repeat the offline phase for every guess of K_1 received in the previous step. The rest of the procedure works in the same manner as for the recovery of K_1. Again, we only received one key guess for K_2 on average in our implementation of the attack.

The complexity of the described attack depends on the number of key candidates for K_1 and K_2. Since the attack on 5 rounds is practical and we have implemented it, we can state that we only have one key candidate on average. So we estimate that the time complexity is about $8 \cdot 2^{32}$. The attack works similarly for reduced versions of Ascon with only 6 initialization rounds. Here, we need borderline cubes of size 32. If we make the optimistic assumption that we only have one key guess for each recovered key word, the estimated time complexity for the 6 round attack is $4 \cdot 2^{64}$.

5 Differential and linear cryptanalysis

Differential [5] and linear [18] cryptanalysis are two standard tools for cryptanalysis. New designs are typically expected to come with some kind of arguments of security against these attacks. For this reason, the designers of Ascon provided security arguments for the individual building blocks (S-box, linear layer), and included first practical results on the differential analysis of Ascon in the design document. In this section, we show some improvements over the existing differential characteristics and present the first linear characteristics for Ascon, including computer-aided proofs on the minimum number of active S-boxes for 3-round characteristics. In addition, we use the combination of differential and linear characteristics to perform practical key-recovery attacks on round-reduced versions of Ascon.

5.1 Linear and differential bounds

Beside using heuristic search techniques to find actual characteristics for Ascon (see Section 5.2), we have also used complete search tools (MILP and SAT) to prove bounds on the best possible linear and differential characteristics. The results are given in this section.

Linear programming. We have first modelled the problem of minimizing the number of active S-boxes in differential characteristics for round-reduced versions of the Ascon permutation as a mixed integer linear program (MILP). The model for R rounds uses the following variables:

- $x_{r,w,b} \in \{0,1\}$ specifies whether bit b of word w of the S-box input in round r is different between the two messages, where $b = 0, \ldots, 63$ and $w = 0, \ldots, 4$.
- $y_{r,w,b} \in \{0,1\}$ specifies whether bit b of word w of the S-box output in round r is different between the two messages, where $b = 0, \ldots, 63$ and $w = 0, \ldots, 4$.
- $d_{r,b} \in \{0,1\}$ specifies if S-box b of round r is active, $b = 0, \ldots, 63$.
- $u_{r,w,b} \in \{0,1,2\}$ is a helper for the linear layer model in word w of round r.

The optimization objective is to minimize the number of active S-boxes,

$$\min \sum_{r=1}^{R} \sum_{b=0}^{63} d_{r,b}.$$

The S-box is modelled only by specifying its branch number, and linking it with the S-box activeness for each $r = 1, \ldots, R$ and $b = 0, \ldots, 63$:

$$d_{r,b} \leq \sum_{w=0}^{63} x_{r,w,b} \leq 5d_{r,b}, \quad \sum_{w=0}^{63} (x_{r,w,b} + y_{r,w,b}) \geq 3d_{r,b}, \quad d_{r,b} \leq \sum_{w=0}^{63} y_{r,w,b} \leq 5d_{r,b}$$

The linear layer is modelled explicitly for $r = 1, \ldots, R$ and $b = 0, \ldots, 63$:

$$\begin{aligned}
y_{r,0,b} + y_{r,0,b+19} + y_{r,0,b+28} + x_{r+1,0,b} &= 2 \cdot u_{r,0,b} \\
y_{r,1,b} + y_{r,1,b+61} + y_{r,1,b+39} + x_{r+1,1,b} &= 2 \cdot u_{r,1,b} \\
y_{r,2,b} + y_{r,2,b+1} + y_{r,2,b+6} + x_{r+1,2,b} &= 2 \cdot u_{r,2,b} \\
y_{r,3,b} + y_{r,3,b+10} + y_{r,3,b+17} + x_{r+1,3,b} &= 2 \cdot u_{r,3,b} \\
y_{r,4,b} + y_{r,4,b+7} + y_{r,4,b+41} + x_{r+1,4,b} &= 2 \cdot u_{r,4,b}
\end{aligned}$$

Finally, at least one S-box needs to be active:

$$\sum_{w=0}^{4} x_{0,w,0} \geq 1$$

The model for linear cryptanalysis is essentially identical, except for different rotation values. This MILP can then be solved using an off-the-shelf linear optimization tool, such as CPLEX. Unfortunately, it turns out that the highly combinatorial nature of the problem is not well suited for linear solvers, and that SAT solvers are a better fit for this type of problem.

SAT solvers. For SAT solvers, we can model essentially the same description by using an extended modelling language, as is used by Satisfiability Modulo Theory (SMT) solvers. We used the constraint solver `STP` by Ganesh et al. [13] to translate a bitvector-based CVC model to conjunctive normal form (CNF). This CNF model can then be solved using a parallel SAT solver, such as Biere's `Treengeling` [3]. Instead of an optimization problem, the problem has to be phrased in terms of satisfiability; i.e., the questions is whether solutions below a specific bound exist.

Modelling the S-box only in terms of its branch number is not very effective for obtaining tight bounds. As a trade-off between the all-too-simplistic branch number model and the complex complete differential description of the S-box (differential distribution table), we chose the following approximation. The linear preprocessing and postprocessing part of the S-box can easily be modelled

exactly for both differential and linear cryptanalysis. The nonlinear core (equivalent to the Keccak S-box) is approximated, i.e., the model allows a few transitions that are not possible according to the differential or linear distribution table. For the differential model, we use the following word-wise constraint in terms of input difference words $a_0, \ldots, a_4 \in \mathbb{F}_2^{64}$ and output difference words $b_0, \ldots, b_4 \in \mathbb{F}_2^{64}$:

$$b_i = a_i \oplus ((a_{i+1} \vee a_{i+2}) \wedge t_i), \qquad t_i \in \mathbb{F}_2^{64}, \quad i = 0, \ldots, 4.$$

For the linear model with word-wise linear input mask $a_0, \ldots, a_4 \in \mathbb{F}_2^{64}$ and output mask $b_0, \ldots, b_4 \in \mathbb{F}_2^{64}$, the constraints are similar:

$$a_i = b_i \oplus ((b_{i-1} \vee b_{i-2}) \wedge t_i), \qquad t_i \in \mathbb{F}_2^{64}, \quad i = 0, \ldots, 4.$$

With this model, we can easily prove that the 3-round Ascon permutation has at least 15 differentially active S-boxes (probability $\leq 2^{-30}$), and at least 13 linearly active S-boxes (bias $\leq 2^{-14}$, complexity $\geq 2^{28}$). The bounds on the number of active S-boxes are tight, but not necessarily those on the probability. Using these results, we can prove that the full 12-round initialization or finalization has at least 60 differentially active S-boxes (probability $\leq 2^{-120}$) and at least 52 linearly active S-boxes (bias $\leq 2^{-53}$, complexity $\geq 2^{106}$). These bounds are almost certainly not tight, but we were not able to derive bounds for more than 3 rounds using SAT solvers. This motivates the use of heuristic search tools to find explicit characteristics.

5.2 Differential and linear characteristics

In Table 4, we present an overview of our best differential and linear characteristics for different round numbers of the Ascon permutation. We have been able to improve the differential characteristic for 4 rounds of the Ascon permutation compared to the previous best results by the designers [11]. Since the designers included no results on linear cryptanalysis in the submission document, we provide the first linear analysis. When comparing the best differential characteristics with the best linear characteristics, we see that for more than two rounds of the Ascon permutation, the linear characteristics have fewer active S-boxes. This might indicate that Ascon is more vulnerable to linear cryptanalysis. Nevertheless, for 5 rounds of Ascon, the best found linear characteristic has more than 64 active S-boxes. Assuming the best possible bias for all active S-boxes, the attack complexity is already higher than 2^{128}.

5.3 Forgery attack on round-reduced Ascon

Usually, the characteristics from Section 5.2 cannot be directly used in an attack, since there might be additional requirements that the characteristic has to fulfill. In the case of an attack on the finalization of Ascon-128, suitable characteristics may only contain differences in stateword x_0 at the input of the permutation.

Table 4. Minimum number of active S-boxes for the Ascon permutation.

result	rounds	differential	linear
proof	1	1	1
	2	4	4
	3	15	13
heuristic	4	44	43
	≥ 5	> 64	> 64

The rest of the statewords have to be free of differences. For the output of the finalization, the only requirement is that there is some fixed difference pattern in x_3 and x_4. Knowledge about the expected differences in x_0, x_1, and x_2 at the output of the permutation is not required.

For round-reduced versions of Ascon, we have found suitable characteristics for a reduced 3-round finalization with a probability of 2^{-33} and for 4-round finalization with a probability of 2^{-101}. The used characteristic for the three round attack is given in Table 6 and the differential for the four round attack is given in Table 7 in Appendix A.

5.4 Differential-linear cryptanalysis

In differential-linear cryptanalysis, differential and linear characteristics are used together in an attack. This kind of analysis was introduced by Langford and Hellman [17]. Later on, it was demonstrated that this type of analysis is also suitable for cases where the differential and the linear part have a probability different from 1 [4,16]. Differential-linear cryptanalysis is especially useful if the combined success probability of one short differential characteristic and one short linear characteristic is better than the probability of a longer linear or differential characteristic. One reason for such a behavior might be a bad diffusion for fewer rounds. For the attack to work, the individual probabilities of the two used characteristics have to be relatively high. According to Dunkelman et al. [12], the bias at the output of such a differential-linear characteristic is about $2pq^2$, where q is the bias of the linear part and p the probability of the differential characteristic. This results in a data complexity of $\mathcal{O}(p^{-2}q^{-4})$.

Outline of the attack. For Ascon-128, we can use differential-linear characteristics as key-stream distinguisher. Like for cube-tester (Section 4.3), we can target either the initialization in a nonce-respecting scenario, or the processing of the plaintext in a nonce-misuse scenario. Here, we focus on the initialization. Therefore, differences are only allowed in the nonce (x_3, x_4), whereas the linear active bits have to be observable and therefore must be in x_0.

Analysis of the initialization. We start with the analysis of a 4-round initialization and create a differential-linear characteristic for it. For the differential

part, we place two differences in the same S-box of round 1. With probability 2^{-2}, we have one active bit at the output of this S-box. The linear layer ensures that 3 S-boxes are active in the second round. Those 3 S-boxes have the difference at the same bit-position of their input. All 3 active S-boxes of round 2 have the same output pattern of 2 active bits with probability 2^{-3}. Due to the linear layer, we then have differences on 11 S-boxes of round 3. For the linear characteristic, we use a characteristic with one active S-box in round 4 and 5 active S-boxes in round 3. The bias of the linear characteristic is 2^{-8}. In addition, we place the S-boxes in a way that the linear active S-boxes in round 3 do not overlap with the 11 S-boxes that have differences at their inputs. The bias of the generated differential-linear characteristic is $2pq^2 = 2^{-20}$. In practice, we are only interested in the bias of the output bit for the specific differences at the input. Due to the vast amount of possible combinations of differential and linear characteristics that achieve these requirements, we expect a much better bias.

Practical evaluation of the bias. In the best case, we place differences in bit 63 of x_3 and x_4, and get a bias of 2^{-2} in bit 9 of x_0 on the output of the substitution layer of round 4. This is much better than the result of 2^{-20} that we obtained from the theoretical analysis. It is possible to combine multiple characteristics to also get to a bias of 2^{-2} in theory. However, we decided to reduce our differential-linear analysis to statistical tests, where we place differences at the input and try to measure a bias at the output bits. We think that this method is sufficient for practical attacks. For a 5-round initialization, we observe a bias of 2^{-10} on $x_0[16]$ (last substitution layer) for differences in $x_3[63]$, and $x_4[63]$. This bias can be improved to 2^{-9} if we only use nonces with the same sign of the difference (the concrete pairs for both $x_3[63]$ and $x_4[63]$ are either $(0, 1)$ or $(1, 0)$). In the case of a 6-round initialization, we were not able to observe a bias by using a set of 2^{36} inputs. The biases were averaged for randomly-chosen keys.

Observing key-dependency of the bias. As shown by Huang et al. [14], the bias observed at the output depends on the concrete values of secret and constant bits. They used this observation to recover the secret state of ICEPOLE in a nonce-misuse scenario. So we expect that a similar attack is possible on round-reduced versions of Ascon-128. In contrast to Huang et al., we want to recover the secret key directly and attack round-reduced versions of the initialization. This also transfers the attack to a nonce-respecting scenario. For a reduced initalization of 4 out of 12 rounds, we observed the bias patterns shown in Table 5. This table shows that the observable bias depends on the concrete values of two key bits which contribute to the same S-box as the used difference. Moreover, the bias is completely independent of the concrete value of the constant in x_0. This leads to the following straightforward attack.

Key-recovery attack on round-reduced Ascon. The target of this attack is a round-reduced version of Ascon-128, where the initialization is reduced to

Table 5. Bias of bit $x_0[i+1]$ in the S-box outputs of round 4 for differences in input bits $x_3[i]$ and $x_4[i]$ (2^{30} different inputs).

inputs $(x_1[i], x_2[i])$	key-bit pair	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
output $x_0[i+1]$	sign	$+1$	-1	$+1$	-1
	bias	$2^{-2.68}$	$2^{-3.68}$	$2^{-3.30}$	$2^{-2.30}$

4 out of 12 rounds. In this setting, the attacker has the ability to choose the nonce and is able to observe the resulting key stream. The attacker performs a sufficient amount of queries, with pairs of nonces which have differences in $x_3[63]$ and $x_4[63]$, and calculates the bias of $x_0[0]$ of the key-stream. With the help of Table 5, the attacker is able to recover two bits of the key by matching the expected bias with his calculated bias. Since the characteristics of Ascon are rotation-invariant within the 64-bit words, the same method can be used to recover the other key bits by placing differences in bits i and observing the bias at position $i+1 \mod 64$. Already 2^{12} samples per bit position i are sufficient to get stable results. This results in an expected time complexity of 2^{18} for the key-recovery attack on 4 rounds. However, in practice, we use the bias of all the bits and compute the correlation with the results of a precomputation (fingerprinting) phase to get better results. This way, we were also able to mount a key-recovery attack on the initialization of Ascon-128 reduced to 5 out of 12 rounds. In particular, we can reliably recover all key-bit pairs with values $(0,0)$ and $(1,1)$ with a low complexity of 2^{36}. However, we need to brute-force the other pairs, which results in an additional complexity of 2^{32} on average and 2^{64} in the worst case. Thus, the expected attack complexity is about 2^{36}. The complexities of both attacks on 4 and 5 rounds of the initialization have been practically verified.

Acknowledgments. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. The work has been supported in part by the Austrian Science Fund (project P26494-N15) and by the Austrian Research Promotion Agency (FFG) and the Styrian Business Promotion Agency (SFG) under grant number 836628 (SeCoS).

References

1. Aumasson, J.P., Meier, W.: Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. CHES rump session (2009)
2. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Keccak Specifications. Submission to NIST (Round 3) (2011), `http://keccak.noekeon.org`
3. Biere, A.: Lingeling, Plingeling and Treengeling entering the SAT Competition 2013. In: Balint, A., Belov, A., Heule, M., Järvisalo, M. (eds.) SAT Competition 2013. vol. B-2013-1, pp. 51–52 (2013), `http://fmv.jku.at/lingeling/`

4. Biham, E., Dunkelman, O., Keller, N.: Enhancing Differential-Linear Cryptanalysis. In: Zheng, Y. (ed.) Advances in Cryptology – ASIACRYPT 2002. LNCS, vol. 2501, pp. 254–266. Springer (2002)
5. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. In: Menezes, A., Vanstone, S.A. (eds.) Advances in Cryptology – CRYPTO 1990. LNCS, vol. 537, pp. 2–21. Springer (1990)
6. Boura, C., Canteaut, A.: A zero-sum property for the Keccak-f permutation with 18 rounds. In: IEEE International Symposium on Information Theory. pp. 2488–2492. IEEE (2010)
7. Boura, C., Canteaut, A., Cannière, C.D.: Higher-order differential properties of keccak and *Luffa*. In: Joux, A. (ed.) Fast Software Encryption – FSE 2011. LNCS, vol. 6733, pp. 252–269. Springer (2011)
8. Daemen, J.: Permutation-based Encryption, Authentication and Authenticated Encryption. DIAC – Directions in Authenticated Ciphers (2012)
9. Dinur, I., Morawiecki, P., Pieprzyk, J., Srebrny, M., Straus, M.: Cube Attacks and Cube-attack-like Cryptanalysis on the Round-reduced Keccak Sponge Function. IACR Cryptology ePrint Archive 2014, 736 (2014), `http://eprint.iacr.org/2014/736`
10. Dinur, I., Shamir, A.: Cube Attacks on Tweakable Black Box Polynomials. In: Joux, A. (ed.) Advances in Cryptology – EUROCRYPT 2009. LNCS, vol. 5479, pp. 278–299. Springer (2009)
11. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon. Submission to the CAESAR competition: `http://ascon.iaik.tugraz.at` (2014)
12. Dunkelman, O., Indesteege, S., Keller, N.: A Differential-Linear Attack on 12-Round Serpent. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) Progress in Cryptology – INDOCRYPT 2008. LNCS, vol. 5365, pp. 308–321. Springer (2008)
13. Ganesh, V., Dill, D.L.: A decision procedure for bit-vectors and arrays. In: Computer Aided Verification (CAV '07). Springer (2007), `https://sites.google.com/site/stpfastprover/`
14. Huang, T., Wu, H., Tjuawinata, I.: Practical State Recovery Attack on ICEPOLE, `http://www3.ntu.edu.sg/home/huangtao/icepole/icepole_attack.pdf`
15. Jovanovic, P., Luykx, A., Mennink, B.: Beyond $2^{c/2}$ Security in Sponge-Based Authenticated Encryption Modes. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014. LNCS, vol. 8873, pp. 85–104. Springer (2014), `http://dx.doi.org/10.1007/978-3-662-45611-8_5`
16. Langford, S.K.: Differential-linear cryptanalysis and threshold signatures. Ph.D. thesis, Stanford University (1995)
17. Langford, S.K., Hellman, M.E.: Differential-linear cryptanalysis. In: Desmedt, Y. (ed.) Advances in Cryptology – CRYPTO 1994. LNCS, vol. 839, pp. 17–25. Springer (1994)
18. Matsui, M., Yamagishi, A.: A New Method for Known Plaintext Attack of FEAL Cipher. In: Rueppel, R.A. (ed.) Advances in Cryptology – EUROCRYPT 1992. LNCS, vol. 658, pp. 81–91. Springer (1992)
19. National Institute of Standards and Technology: FIPS PUB 180-4: Secure Hash Standard. Federal Information Processing Standards Publication 180-4, U.S. Department of Commerce (March 2012), `http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf`
20. The CAESAR committee: CAESAR: Competition for authenticated encryption: Security, applicability, and robustness (2014), `http://competitions.cr.yp.to/caesar.html`

A Differentials to create forgery

Table 6 contains the differential characteristic and Table 7 contains the differential used for the forgery attacks of Section 5.3. One column corresponds to the five 64-bit words of the state, and the xor differences are given in hexadecimal notation (truncated in the last round).

Table 6. Differential characteristic to create forgery for round-reduced Ascon-128 with a 3-round finalization. The differential probability is 2^{-33}.

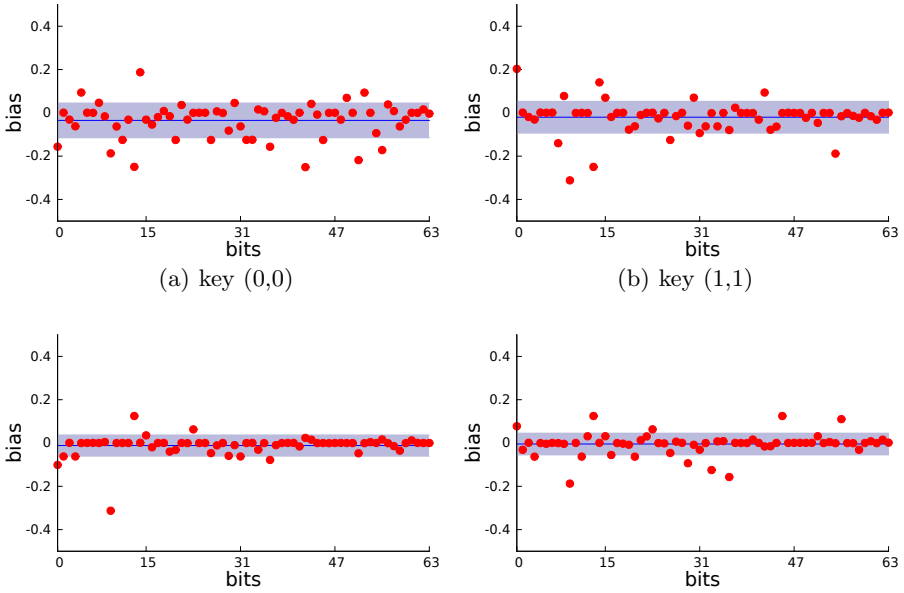
input difference	after 1 round	after 2 rounds	after 3 rounds
x_0 8000000000000000	8000100800000000	8000000002000080	????????????????
x_1 0000000000000000	8000000001000004	9002904800000000	????????????????
x_2 0000000000000000 $\rightarrow$	0000000000000000 $\rightarrow$	d200000001840006 $\rightarrow$	????????????????
x_3 0000000000000000	0000000000000000	0102000001004084	4291316c5aa02140
x_4 0000000000000000	0000000000000000	0000000000000000	090280200302c084

Table 7. Differential to create forgery for round-reduced Ascon-128 with a 4-round finalization. The differential probability is 2^{-101}.

input difference	after 4 rounds
x_0 8000000000000000	????????????????
x_1 0000000000000000	????????????????
x_2 0000000000000000 $\rightarrow$	????????????????
x_3 0000000000000000	280380ec6a0e9024
x_4 0000000000000000	eb2541b2a0e438b0

B Differential-linear key recovery attack on 4 rounds

Fig. 2 illustrates the observed bias in bit $x_0[i]$ in the key-stream for the differential-linear attack of Section 5.4, grouped by the values of the key-bit pair $(x_1[63], x_2[63])$.



(a) key (0,0) (b) key (1,1)

(c) key (0,1) (d) key (1,0)

Fig. 2. Biases for the differential-linear attack on the initialization of Ascon reduced to 4 (out of 12) rounds for the key-bit pair values $(0,0)$, $(0,1)$, $(1,0)$, $(1,1)$.