Preimage and Collision Attacks on Reduced Ascon Using Algebraic Strategies

Qinggan Fu[1], Ye Luo[1], Qianqian Yang[2,3], and Ling Song[1,4](✉)

[1] College of Cyber Security, Jinan University, Guangzhou, China
fuqinggan@stu2018.jnu.edu.cn, roylaw456@gmail.com, songling.qs@gmail.com
[2] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[3] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China yangqianqian@iie.ac.cn
[4] National Joint Engineering Research Center of Network Security Detection and Protection Technology, Jinan University, Guangzhou, China

Abstract. Ascon, a family of algorithms that supports hashing and authenticated encryption, is the winner of the NIST Lightweight Cryptography Project. In this paper, we propose an improved preimage attack against 2-round Ascon-XOF-64 with a complexity of 2^{32} via a better guessing strategy. Furthermore, in order to find a good guessing strategy efficiently, we build a MILP model and successfully extend the attack to 3 rounds. The time complexity is 2^{53} when $IV = 0$, while for the real IV, the attack still works and the time complexity is 2^{51}. Additionally, we also investigate the resistance of Ascon-HASH against collision attacks. We introduce the linearization of the inverse of S-boxes and then propose a practical free-start collision attack on 3-round Ascon-HASH using a differential trail searched dedicatedly. Furthermore, We construct different 2-round connectors using the linearization of the inverse of S-boxes and successfully extend the collision attack to 4 rounds and 5 rounds of Ascon-HASH with complexities of 2^{21} and 2^{41} respectively. Although our attacks do not compromise the security of the full 12-round Ascon-XOF and Ascon-HASH, they provide some insights into Ascon's security.

Keywords: Ascon · Preimage attack · Collision attack · Guessing strategy · Linearization.

1 Introduction

With the increasing demand for a cryptographic primitive that provides both encryption and authentication and the rise of lightweight cryptography that is suitable to resource-constrained platforms, the National Institute of Standards and Technology (NIST) decided to solicit lightweight authenticated encryption by hosting a cryptographic competition in 2019. In February 2023, The NIST Lightweight Cryptography Team announced that they decided to standardize the Ascon family for lightweight cryptographic applications as it meets the needs of most use cases where lightweight cryptography is required. Therefore, Ascon [5] is the ultimate winning algorithm after several rounds of selection processes. The Ascon family consists of the authenticated ciphers Ascon-128 and Ascon-128a, hash functions Ascon-HASH and Ascon-HASHA, and extendable output functions Ascon-XOF and Ascon-XOFA. All schemes use the 320-bit permutation which is repeated for 12 times. AEAD schemes and hashing schemes in Ascon are based on duplex construction and sponge construction [1], respectively.

In recent years, there are a majority of valuable research results on Ascon, including the results of the underlying permutation [6] [10] [13] [8] and AEAD [6] [10] [12]. However, compared to permutation and authentication encryption, which have been extensively analyzed, relatively few analyses focus on Ascon-HASH and Ascon-XOF. Dobraunig et al., the designers of Ascon, proposed a preimage attack on 2-round Ascon-XOF with a 64-bit output (aka Ascon-XOF-64) based on state linearization strategy in [7]. In addition, Zong et al. [18] performed a 2-round collision attack on Ascon-HASH with a non-practical time complexity of 2^{125} and a collision attack on 2-round Ascon-XOF with a practical time complexity of 2^{15}. Gerault et al. [10] proposed an improved collision attack on 2-round Ascon-HASH with a time complexity of 2^{103}, which is based on a differential trail with a higher probability than the previous one. Yu et al. [17] performed a practical collision attack on 2-round Ascon-HASH based on some critical observations on the round function of Ascon, its complexity is $2^{62.6}$. With the widespread use of Ascon, analyzing its security against various attacks has become increasingly important.

Table 1. Summary of attacks on Ascon-XOF and Ascon-HASH

Primitive	Type	Size	Rounds	Time	Reference
Ascon-XOF	Preimage	64	2/12	2^{39}	[7]
Ascon-XOF	**Preimage**	**64**	**2/12**	$\mathbf{2^{32}}$	**Section3**
Ascon-XOF	**Preimage**	**64**	**3/12**	$\mathbf{2^{53}}$	**Section4**
Ascon-XOF*	**Preimage**	**64**	**3/12**	$\mathbf{2^{51}}$	**Section4**
Ascon-HASH	Collision	256	2/12	2^{125}	[18]
Ascon-HASH	Collision	256	2/12	2^{103}	[10]
Ascon-HASH	Collision	256	2/12	$2^{62.6}$	[17]
Ascon-HASH	**FS Collision**	**256**	**3/12**	$\mathbf{2^{14}}$	**Section5**
Ascon-HASH	**FS Collision**	**256**	**4/12**	$\mathbf{2^{21}}$	**Section5**
Ascon-HASH	**FS Collision**	**256**	**5/12**	$\mathbf{2^{41}}$	**Section5**

* means that the IV is a real IV and round constants and initialization are both considered. FS is the abbreviation of Free-Start.

1.1 Our Contribution.

In this paper, we aim to analyze the security of hashing modes Ascon-HASH and Ascon-XOF of Ascon. On the one hand, we work on improving the time complexity of the 2-round preimage attack and extending the attack to 3 rounds. On the other hand, we also investigate the resistance of Ascon-HASH against collision attacks. The previous attacks on Ascon-XOF and Ascon-HASH and our new results are summarized in Table 1. Our contributions are summarized as follows.

Improved preimage attack on 2-round Ascon-XOF The main idea of our improved preimage attack is to derive 32 linear equations for the inputs of round 1 and the outputs after the substitution layer in round 2 by guessing 32 input bits of round 2. By applying Gauss-Jordan elimination to those linear equations, we can fully recover the message corresponding to the target hash value. Furthermore, our improved preimage attack can reduce the complexity from 2^{39} summarized in [7] to 2^{32}.

Improved preimage attack on 3-round Ascon-XOF On the one hand, we propose a preimage attack against 3-round Ascon-XOF, where IV is set to 0 and round constants and initialization are ignored. For the first time, we transform the manual selection of guess bits in Ascon preimage attacks into an automated optimization problem and solve it through a dedicated MILP model. Specifically, we perform a preimage attack on 3-round Ascon-XOF by just guessing 53 state bits of the input in round 1, which means that we can find a preimage of 3-round Ascon-XOF with a complexity of 2^{53}. On the other hand, we also propose a more realistic preimage attack against 3-round Ascon-XOF, where the IV is a real IV and round constants and initialization are both considered. We can obtain 13 linear equations by just guessing 51 state bits of the input in round 1 based on another dedicated MILP model, which means that we can find a preimage of 3-round Ascon-XOF with a complexity of 2^{51}.

Improved collision attack on 3-round Ascon-HASH We propose a free-start collision attack on 3-round Ascon-HASH based on a new 3-round differential trail searched dedicatedly, the main idea of which is to construct a 2-round connector so that the differential propagation probability of the last two rounds is 1. Moreover, the 2-round connector, playing an essential role in our attack, is constructed by using the linearization of the inverse of S-boxes. As a result, in the free-start setting, the complexity of our proposed collision attack on 3-round Ascon-HASH is 2^{14}.

Collision attacks on 4-round and 5-round Ascon-HASH We construct different 2-round connectors using the linearization of the inverse of S-boxes and successfully extend the collision attack to 4 rounds and 5 rounds of Ascon-HASH with complexities of 2^{21} and 2^{41} respectively. The essential factors in the success of these attacks are differential trails searched dedicatedly and 2-round connectors based on the linearization of the inverse of S-boxes.

1.2 Organization.

The rest of paper is organized as follows. In Section 2, we give some preliminaries and briefly describe the Ascon. In Section 3, we present our improved preimage attack against 2-round Ascon-XOF. In Section 4, we describe our improved preimage attacks against 3-round Ascon-XOF. In Section 5, we describe our 3-round, 4-round and 5-round collision attacks against Ascon-HASH respectively. Finally, the paper is concluded in Section 6.

2 Preliminaries

In this section, we will describe some definitions of preimage attack and collision attack and the details of Ascon required for our attacks.

2.1 A Brief Description of Ascon

Ascon [4] proposed by Dobrauning et al., includes a permutation-based AEAD and hashing schemes using a sponge duplex construction (see Fig.1). Its core components are the two 320-bit permutations p^a and p^b with a and b rounds, respectively. In the hash modes, both a and b are set to 12 (b is set to 8 for Ascon-HASHA and Ascon-XOFA), the details are illustrated in Fig.1. For the description of the round transformations, the 320-bit state S is split into five 64-bit words X_i, $S = X_0||X_1||X_2||X_3||X_4$ (as shown in Fig.2). The 320-bit initial state of Ascon-HASH and Ascon-XOF is defined by a constant IV which specifies the algorithm parameters, including key k and round numbers a and the value of $a-b$ and the rate r, each written as an 8-bit integer (with $h = l = 256$ for Ascon-HASH and $h = 0$ for unlimited output in Ascon-XOF). Then, it is followed by the maximal output length of h bits as a 32-bit integer. Especially, key is set to 0 in the hash modes. The a-round permutation p^a is applied to initialize the state

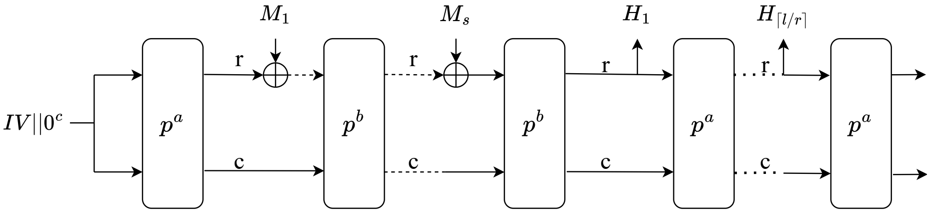

Fig. 1. Ascon's Hashing Modes

S. Ascon-HASH and Ascon-XOF process the message M in blocks of r bits. The padding process is as follows: it appends a single 1 and the smallest of 0s to M such that the length of the padded message is an integer multiple of r bits. The padded message is split into s blocks of r bits and described as Formula 1.

$$M_1, .., M_s \leftarrow r\text{-bit blocks of } M||1||0^{r-1-(|M| \bmod \mathrm{r})} \tag{1}$$

The permutations iteratively apply a round transformation p, which in turn consists of three functions: the addition of constants p_C, the substitution layer p_S, and the linear diffusion layer p_L ($p = p_L \circ p_S \circ p_C$). When representing the *i-th* round in a layer, we append the number to the subscript; for instance, the substitution of the 1-*th* round is written as $p_{S,1}$. The output state after p_S layer is denoted as $Y^r = Y_0^r||Y_1^r||Y_2^r||Y_3^r||Y_4^r$ while the input state to the permutation at *r-th* round is represented as $X^r = X_0^r||X_1^r||X_2^r||X_3^r||X_4^r$. The bit at round r, row i and column j will be denoted as $S_i^r[j]$. For instance, $X_i^r[j]$ represents the *j-th* bit of word i at the round *r-th* for $j = 0, ..., 63$.

Addition of Constants (p_C) The addition of constants p_C adds the round constant c_r to the $X_2[7, ..., 0]$ at each round where $X_2[7, ..., 0]$ denotes 8 consecutive bits of X_2 (see Fig.2). The added constant changes depending on the round.

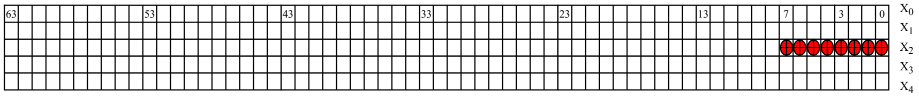

Fig. 2. Ascon's constant addition layer

Substitution Layer (p_S) The substitution layer p_S updates the state S through 64 parallel applications of the 5-bit S-box (see Fig.3). The algebraic normal form (ANF) of the S-box is shown in Equation 2.

$$\begin{aligned}
Y_0[j] &= X_4[j]X_1[j] \oplus X_3[j] \oplus X_2[j]X_1[j] \oplus X_2[j] \oplus X_1[j]X_0[j] \oplus X_1[j] \oplus X_0[j] \\
Y_1[j] &= X_4[j] \oplus X_3[j]X_2[j] \oplus X_3[j]X_1[j] \oplus X_3[j] \oplus X_2[j]X_1[j] \oplus X_2[j] \oplus X_1[j] \oplus X_0[j] \\
Y_2[j] &= X_4[j]X_3[j] \oplus X_4[j] \oplus X_2[j] \oplus X_1[j] \oplus 1 \\
Y_3[j] &= X_4[j]X_0[j] \oplus X_4[j] \oplus X_3[j]X_0[j] \oplus X_3[j] \oplus X_2[j] \oplus X_1[j] \oplus X_0[j] \\
Y_4[j] &= X_4[j]X_1[j] \oplus X_4[j] \oplus X_3[j] \oplus X_1[j]X_0[j] \oplus X_1[j]
\end{aligned} \tag{2}$$

Linear Diffusion Layer (p_L) Each row of the 320-bit state consists of 64 bits, which is diffused by a linear function $\sum_i(X_i)$, as shown in Equation 3 (see Fig.4).

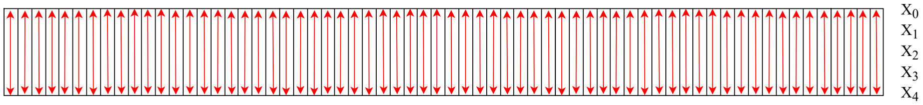

Fig. 3. Ascon's substitution layer

Here ⋙ denotes the right cyclic shift operation over the 64-bit word.

$$\begin{aligned} Y_0 &= \Sigma_0\left(X_0\right) = X_0 \oplus \left(X_0 \ggg 19\right) \oplus \left(X_0 \ggg 28\right) \\ Y_1 &= \Sigma_1\left(X_1\right) = X_1 \oplus \left(X_1 \ggg 61\right) \oplus \left(X_1 \ggg 39\right) \\ Y_2 &= \Sigma_2\left(X_2\right) = X_2 \oplus \left(X_2 \ggg 1\right) \oplus \left(X_2 \ggg 6\right) \\ Y_3 &= \Sigma_3\left(X_3\right) = X_3 \oplus \left(X_3 \ggg 10\right) \oplus \left(X_3 \ggg 17\right) \\ Y_4 &= \Sigma_4\left(X_4\right) = X_4 \oplus \left(X_4 \ggg 7\right) \oplus \left(X_4 \ggg 41\right) \end{aligned} \tag{3}$$

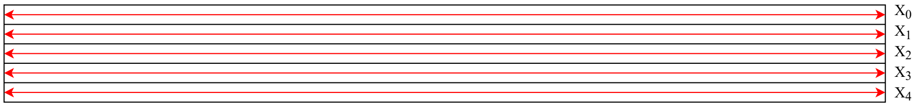

Fig. 4. Ascon's linear diffusion layer

2.2 Preimage Attack and Collision Attack

The definition of preimage attacks is as follows. Given a function H and a target value y, the goal of preimage attack is to find an input massage $x \in \{0,1\}^n$ such that $H(x) = y$.

Additionaly, the 2-round preimage attack configuration used by Dobrauning et al. [7] is described in Fig.5. Under the condition that the IV is set to zero and all-zero IV is taken as the input of the round permutation, the input state bits of $X_1[j], X_2[j], X_3[j], X_4[j]$ are fixed to zero in round 1. Since the message M is directly XORed to the first row of the input state in round 1, $X_0[j]$ of round 1 can be determined by the attacker. Based on these conditions, we can conclude two properties of Ascon's S-box as follows.

Property 1. [7] If $X_1[j], X_2[j], X_3[j], X_4[j]$ are fixed to zero, then, $Y_0[j], Y_1[j]$, $Y_3[j]$ are determined by $X_0[j]$.

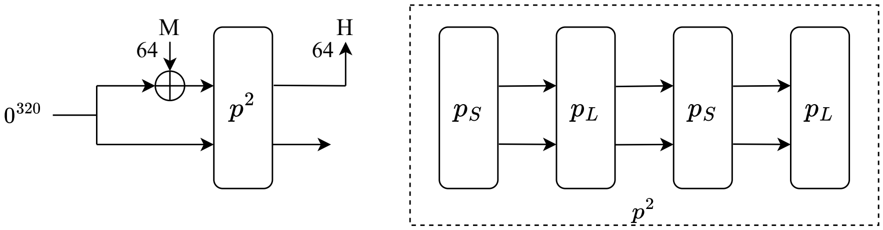

Fig. 5. 2-round preimage attack configuration

Property 2. [7] If $X_1[j], X_2[j], X_3[j], X_4[j]$ are fixed to zero, then, $Y_2[j] = 1$ and $Y_4[j] = 0$ always hold.

Furthermore, the definition of collision attacks is to find any pair of different input messages (M_1, M_2) ($M_1 \in \{0,1\}^n$, $M_2 \in \{0,1\}^n$) such that $H(IV, M_1) = H(IV, M_2)$. Moreover, the definition of free-start collision attacks is to find any pair of initial values (IV_1, IV_2) and a pair of messages (M_1, M_2) ($M_1 \in \{0,1\}^n$, $M_2 \in \{0,1\}^n$) such that $H(IV_1, M_1) = H(IV_2, M_2)$ but $IV_1 = IV_2$ and $M_1 = M_2$ do not hold simultaneously (see Fig.6).

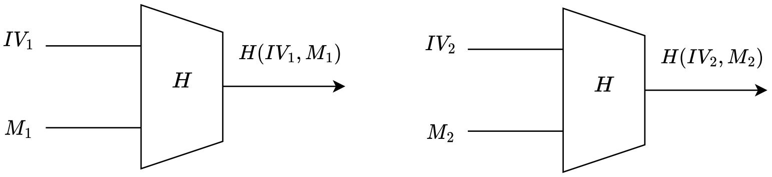

Fig. 6. Free-start collision attack configuration

3 Preimage Attacks on 2-round Ascon-XOF

In this section, we propose an improved preimage attack against 2-round Ascon-XOF (with a 64-bit output) with the same setting as the one used by Dobrauning et al. [7], where the IV is set to 0 and round constants and initialization are ignored. The main idea of the preimage attack on 2-round Ascon-XOF proposed by Dobrauning et al. [7] is to linear the state bits after the substitution layer of round 2 by guessing 39 consecutive input state bits of round 1. More specifically, they can obtain 25 linear equations after 2 rounds by guessing 39 consecutive input state bits. Then, by solving the 25 linear equations, they can determine the remaining 25 input state bits of round 1, which means that they can find a preimage of 2-round Ascon-XOF with a complexity of 2^{39}.

3.1 Our Improved Preimage Attack on 2-round ASCON-XOF

Here we perform a preimage attack against 2-round Ascon-XOF by just guessing 32 state bits of round 2 before the substitution layer, which means that we can find a preimage of 2-round Ascon-XOF with a complexity of 2^{32}. We will give more details about our attack in the following.

There is a linear layer from Y^1 to X^2. According to the property of Ascon's linear layer, $X_2^2[j] = 1$ and $X_4^2[j] = 0$ always hold due to Property 2. Then, $Y_0^2[j]$ can be calculated as given in Equation 4 based on Property 2 and ANF of S-box (The additions used in this paper are all modulo 64 additions).

$$Y_0^2[j] = X_1^2[j]X_0^2[j] \oplus X_0^2[j] \oplus X_3^2[j] \oplus 1 \tag{4}$$

Guessing each bit of X_0^2, we can obtain two linear equations (ignore constants) as shown in equation 5.

$$\begin{aligned} Y_0^2[j] &= X_1^2[j] \oplus X_3^2[j] \\ X_0^2[j] &= X_0^1[j] \oplus X_0^1[j+19] \oplus X_0^1[j+28] \end{aligned} \tag{5}$$

Y_0^2 can be obtained by applying $P_{L,2}^{-1}$ to H_0, therefore, analyzing Y_0^2 is sufficient to find a preimage. As shown in Fig.7, when guessing the last 32 bits of X_0^2, we

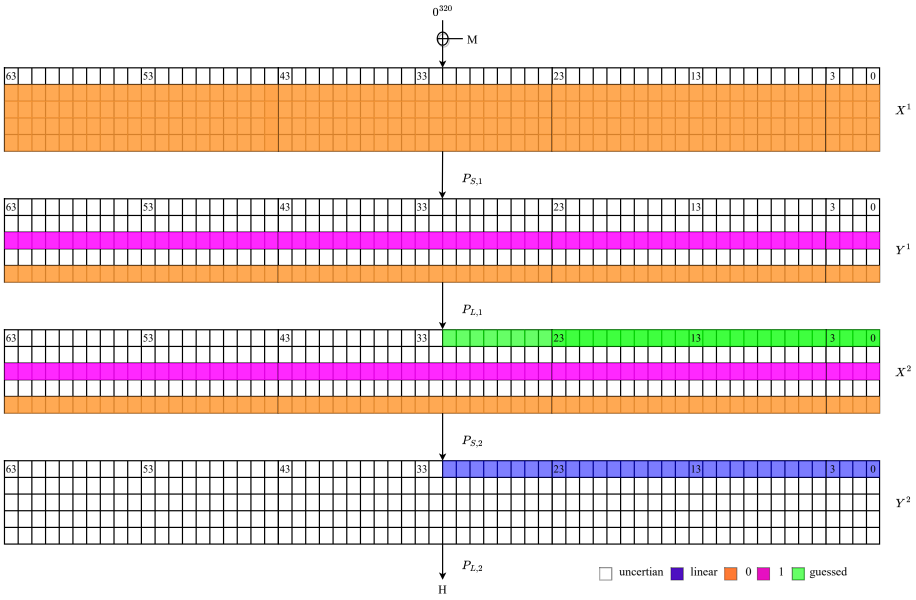

Fig. 7. 2-round preimage attack

can get 64 equations, denoted as E_P (64 equations are described in Appendix A). By solving the linear equation systems E_p, we can obtain a unique solution denoted as X^{pre}. By applying p^2 to X^{pre}, we can get a possible hash value H_0^{pre}, with a probability of 2^{-32} of matching the true hash value H_0. Therefore, we can obtain a preimage of H_0 with a probability of 1 by traversing 2^{32} possible values of $X_0^2[0], ..., X_0^2[31]$ and solving the corresponding equation systems. The process of finding a preimage for H_0 is as follows.

1. Calculate Y_0^2 from the hash value H_0 through $P_{L,2}^{-1}$.

2. Take a set of possible values in $X_0^2[31], ..., X_0^2[0] \in \{0, ..., 2^{32} - 1\}$. Then, generate 64 linear equations based on these values and Y_0^2.

3. Check whether the linear equation system is unsolvable. If it is unsolvable, return to Step 2 and perform the same process for other values of $X_0^2[31], ..., X_0^2[0]$. If the linear equation system is solvable, go to Step 4.

4. Applying Gauss-Jordan elimination to the constructed linear equation systems, obtain a unique solution, denoted as X^{pre}.

5. Applying 2-round Ascon-XOF to X^{pre}. Check whether the calculated hash value matches H_0. If the hash value does not match, go back to Step 2 and perform the same process for other values of $X_0^2[31], ..., X_0^2[0]$.

3.2 Complexity Analysis and Comparisons of Our Improved Preimage Attack

As shown in Fig.7, we can linearize 32 bits of $Y_0^2[31, ..., 0]$ by guessing 32 bits of $X_0^2[31, ..., 0]$. Therefore, our proposed preimage attack is valid because 64 variables of X_0^1 are all included in 64 linear equations (described in Appendix A). Therefore, our proposed preimage attack on 2-round Ascon-XOF just needs to guess 32 bits of X_0^2 and solve 2^{32} corresponding equation systems. For preimage attacks against 2-round Ascon-XOF, the summary of the results can be found in Table 2. As shown in Table 2, in [7], Dobrauning et al. perform a preimage attack on 2-round Ascon-XOF by guessing 39 bits of X_0^1, with a complexity of 2^{39}. Compared to the attack proposed by Dobrauning et al. [7], our proposed preimage attack on 2-round Ascon-XOF can reduce the complexity from 2^{39} to 2^{32}.

Table 2. Summary of preimage attacks against 2-round Ascon-XOF.

Primitive	#rounds	#guess bits	Complexity	Ref.
Ascon-XOF	2/12	39	2^{39}	[7]
Ascon-XOF	2/12	32	2^{32}	Section 3

4 Preimage Attacks on 3-round Ascon-XOF

In the previous section, we perform a preiamge attack on 2-round Ascon-XOF-64 based on a better guessing strategy. However, how to choose a good guessing strategy becomes very complicated when the number of rounds increases. Therefore, to solve this complex problem, for the first time, we transform the manual selection of guess bits in Ascon preimage attacks into an automated optimization problem and solve it through the MILP program. In this section, we propose a preimage attack on 3-round Ascon-XOF-64 based on a dedicated MILP model, where the IV is set to 0 and round constants and initialization are ignored. Additionally, we also propose a more realistic preimage attack on 3-round Ascon-XOF-64 using another dedicated MILP model, where the IV is a real IV and round constants and initialization are both considered.

4.1 Analysis of Preimage Attack on 3-round Ascon-XOF

The resistance to preimage attacks is a significant security property of hash functions because it ensures that the hash function is a one-way function, meaning that it is easy to compute the hash value of an input, but difficult to compute the input from the hash value. This property is important in many cryptographic applications, such as digital signatures, message authentication codes, and password storage, where it is necessary to ensure that an attacker cannot recover the original input from its hash value. Therefore, the resistance to preimage attacks is a critical metric in the security analysis of hash functions. Given a function f and a target value y, the goal of preimage attacks is to find an input $x \in \{0,1\}^n$ such that $f(x) = y$. A valid preimage attack is to find x such that $f(x) = y$ at a complexity cost less than 2^n. Therefore, the problem of finding a preimage is actually an optimization problem. In cryptanalysis, attack tools for solving optimization problems include MILP (Mixed Integer Linear Programming) [15] and SAT (Boolean Satisfiability Problem) [14], etc.

Currently, the preimage attacks on Ascon are based on guessing strategies and manual calculations to construct a linear equation system, and then find the preimage by solving the equation systems. The drawback of this method is that it is easy to ignore better solutions, resulting in higher complexity. For instance, the preimage attacks on Ascon-XOF proposed by Dobraunig et al. rely on manually selecting guess bits, constructing a linear equation system and solving the linear system to obtain the preimage [7]. Therefore, we transform the manual selection of guess bits in Ascon preimage attacks into an automated optimization problem and solve it through the MILP program. To achieve this transformation, we need to solve three major challenges.

1. How to establish the relationship between the linearization conditions of the linearized bits in the output layer and the guessed bits in the input layer.

2. If the state bit that needs to be guessed contains several variables, the situation where the state bit is judged to have been guessed is very complex.

How to choose a compromise method to model this complex situation and achieve good results requires careful consideration.

3. When considering the actual IV, The conditions of each state bit to be linearized or guessed are not regular, so the expression of each state bit needs to be calculated, but it is very difficult to manually calculate the expression of each state bit.

In response to the first challenge, we observe the properties of the ANF of Ascon's S-box and establish the relationship between the linearization conditions of the linearized bits in the output layer and the guessed bits in the input layer from top to bottom. For instance, $Y_0^3[j]$ can be expressed as the equation 6.

$$Y_0^3[j] = (X_4^3[j] \oplus X_2^3[j] \oplus X_0^3[j] \oplus 1)X_1^3[j] \oplus X_3^3[j] \oplus X_2^3[j] \oplus X_0^3[j] \tag{6}$$

As shown in equation 6, there are two situations where $Y_0^3[j]$ is linearized. In the first situation, $Y_0^3[j]$ is linearized when $X_1^3[j]$ is guessed and $X_0^3[j]$, $X_3^3[j]$ and $X_4^3[j]$ are linearized ($X_2^3[j]$ is always linear when IV is zero). In the second situation, $Y_0^3[j]$ is linearized when $X_0^3[j]$, $X_2^3[j]$ and $X_4^3[j]$ are guessed and $X_1^3[j]$ and $X_3^3[j]$ are both linearized. By combining these two situations, we can establish the relationship between the conditions for Y_0^3 to be linearized and X^3. Moreover, If we continue to establish the relationship between the conditions imposed on X^3 by linearization of Y_0^3 and input value X_0^1, then the relationship between the linearization conditions of Y_0^3 and input value X_0^1 is established. More modeling details can be found in Algorithm 1.

The second challenge can be illustrated by an example. As for $X_1^3[j]$, $X_1^3[j]$ can be expressed as the equation 7.

$$X_1^3[j] = Y_1^2[j] \oplus Y_1^2[j+61] \oplus Y_1^2[j+39] \tag{7}$$

In fact, if $X_1^3[j]$ needs to be guessed, we need to consider the situation where $Y_1^2[j]$, $Y_1^2[j+61]$, and $Y_1^2[j+39]$ are guessed. The number of values in $\{Y_1^2[j]$, $Y_1^2[j+61]$, $Y_1^2[j+39]\}$ that have not been guessed denoted as M. If M is greater than or equal to 2 when we need to guess $X_1^3[j]$, the number of linear equations needs to be increased by 1. If M is equal to 1, guessing $X_1^3[j]$ is equivalent to guessing the values in $\{Y_1^2[j]$, $Y_1^2[j+61]$, $Y_1^2[j+39]\}$ that have not been guessed. If M is equal to 0, $X_1^3[j]$ is equivalent to having been guessed. As described above, the situation where the state bit containing several variables needs to be guessed is complex. In order to reduce the complexity of modeling and find better solutions, for all state bits that need to be guessed, our strategy is that they are considered to be guessed only if all the variables they contain have been guessed. For instance, the conditions for $X_1^3[j]$ to be guessed is whether all three values $\{Y_1^2[j]$, $Y_1^2[j+61]$, $Y_1^2[j+39]\}$ it contains have been guessed.

The third challenge can be described as follows. We need to know the expression of the state bit when it needs to be guessed or linearized. In other words, we need to determine which items in the expression for this state bit need to be guessed or linearized. However, the expression of each round's state bit under the actual IV may be irregular, so we need to calculate the specific expression for each state bit. Since the state of Ascon is 320 bits, manually calculating the expression for each state bit would require enormous computations. Therefore, we write a program by Sagemath to obtain the expression for each state bit under the actual IV.

4.2 Our Preimage Attack on 3-round Ascon-XOF under Zero IV

In this section, we propose a preimage attack on 3-round Ascon-XOF (with a 64-bit output), where the IV is set to 0 and round constants and initialization are ignored. Specifically, we perform a preimage attack on 3-round Ascon-XOF by just guessing 53 state bits of X_0^1, which means that we can find a preimage of 3-round Ascon-XOF with a complexity of 2^{53}. Indeed, our proposed preimage attack on 3-round Ascon-XOF under 0 IV is based on MILP, more details about our MILP model can be found in Algorithm 1. Next, we will provide a detailed description to our MILP modeling process.

Based on the ANF of Ascon's S-box, $Y_0^3[j]$ can be expressed as the equation 6.As shown in Equation 6, there are two situations where $Y_0^3[j]$ is linearized. In the first situation, $Y_0^3[j]$ is linearized when $X_1^3[j]$ is guessed and $X_0^3[j]$, $X_3^3[j]$ and $X_4^3[j]$ are linearized ($X_2^3[j]$ is always linear). In the second situation, $Y_0^3[j]$ is linearized when $X_0^3[j]$, $X_2^3[j]$ and $X_4^3[j]$ are guessed and $X_1^3[j]$ and $X_3^3[j]$ are both linearized. We will provide more details about our MILP program using the first situation as an example. As shown in Fig.8, X_1^3 is obtained by applying $P_{L,2}$ to Y_1^2. Therefore, $X_1^3[j]$ can be expressed as the equation 7. As shown in equation 7, $X_1^3[j]$ is guessed when $Y_1^2[j]$, $Y_1^2[j+61]$ and $Y_1^2[j+39]$ are all guessed. Similarly, $X_1^3[j]$ is linearized when $Y_1^2[j]$, $Y_1^2[j+61]$ and $Y_1^2[j+39]$ are all linearized.

At the same time, $Y_1^2[j]$ can be simplified as shown in equation 8 because X_2^2 always equals 1 and X_4^2 always equals 0 (see Fig.8).

$$\begin{aligned}
Y_0^2[j] &= X_1^2[j]X_0^2[j] \oplus X_3^2[j] \oplus X_0^2[j] \oplus 1 \\
Y_1^2[j] &= X_3^2[j]X_1^2[j] \oplus X_0^2[j] \oplus 1 \\
Y_3^2[j] &= X_3^2[j]X_0^2[j] \oplus X_3^2[j] \oplus X_1^2[j] \oplus X_0^2[j] \oplus 1 \\
Y_4^2[j] &= X_1^2[j]X_0^2[j] \oplus X_3^2[j] \oplus X_1^2[j]
\end{aligned} \tag{8}$$

Therefore, $Y_1^2[j]$ is guessed when $X_3^2[j]$, $X_1^2[j]$ and $X_0^2[j]$ are all guessed. At the same time, $Y_1^2[j]$ is linearized when $X_3^2[j]$ or $X_1^2[j]$ is guessed. When $Y_0^2[j]$, $Y_3^2[j]$, and $Y_4^2[j]$ need to be guessed or linearized, the situation is similar to that of $Y_1^2[j]$. Especially, $Y_2^2[j]$ is equal to $X_1^2[j]$. Therefore, $Y_2^2[j]$ is guessed when $X_1^2[j]$ is guessed. At the same time, $Y_2^2[j]$ itself is linear. $X_0^2[j]$ is a linear combination of different bits of Y_0^1, as shown in Equation 3. Therefore, $X_0^2[j]$ is guessed when $Y_0^1[j]$, $Y_0^1[j+19]$ and $Y_0^1[j+28]$ are all guessed. When $X_1^2[j]$ and $X_3^2[j]$ need to be guessed, the situation is similar to that of $X_0^2[j]$. Moreover, based on Property 1, if $X_0^1[j]$ is guessed then $Y_3^1[j]$, $Y_1^1[j]$ and $Y_0^1[j]$

are all guessed. Following the steps described above, the relationship between the linearization conditions of Y_0^3 and input value X_0^1 can be established.

Algorithm 1 Establishing a MILP model of 3-round Ascon preimage attack

```
Input: N: Index set of guessed bits in $X_0^1$; Intermediate state variables: $Y^1$, $X^2$, $Y^2$, $X^3$
Output: M: Number of bits of $Y_0^3$ to be linearized
1: According to equation 6, convert the linearization conditions of $Y_0^3[j]$ into constraint conditions on $\{X_0^3[j], X_1^3[j], X_2^3[j], X_3^3[j], X_4^3[j]\}$ $(j \in \{0, 1, ..., 63\})$, and record it as $setX^3$
2: According to equation 3, convert the constraint conditions of $X^3(setX^3)$ into constraint conditions on $\{Y_0^2[j], Y_1^2[j], Y_2^2[j], Y_3^2[j], Y_4^2[j]\}$, and record it as $setY^2$
3: According to equation 2, convert the constraint conditions of $Y^2(setY^2)$ into constraint conditions on $\{X_0^2[j], X_1^2[j], X_2^2[j], X_3^2[j], X_4^2[j]\}$, and record it as $setX^2$
4: According to equation 3, convert the constraint conditions of $X^2(setX^2)$ into constraint conditions on $\{Y_0^1[j], Y_1^1[j], Y_2^1[j], Y_3^1[j], Y_4^1[j]\}$, and record it as $setY^1$
5: According to equation 2, convert the constraint conditions of $Y^1(setY^1)$ into constraint conditions on $X_0^1[j]$
6: Add additional constraints: $len(N) + M \leqslant 64$ and the linearized $Y_0^3[j]$ can not be a constant
7: Set the objective function: Maximize M
8: According to the conditional inequality obtained from step 1 to 6, solve the model using the MILP optimizer
9: A feasible solution is found, save it to a file
```

The results of our preimage attack on 3-round Ascon-XOF are shown in Table 3, and the linear expressions for the linearized bits of Y_0^3 are shown in Appendix B. Our proposed preimage attack on 3-round Ascon-XOF is valid because the remaining 11 bits that have not been guessed are all included in the 11 linear equations. Therefore, we can obtain a preimage of H with a probability of 1 by traversing 2^{53} possible values of guessing bits and solving the corresponding equation systems.

Table 3. Results of preimage attacks against 3-round Ascon-XOF.

Item	Index	#Total bits
Guess bits of X_0^1	1 2 4 5 6 7 8 9 11 12 13 14 15 16 18 19 21 22 24 25 26 27 28 29 30 31 33 34 35 37 38 40 41 43 44 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63	53
Linear bits of Y_0^3	2 5 12 21 24 27 30 34 37 52 63	11

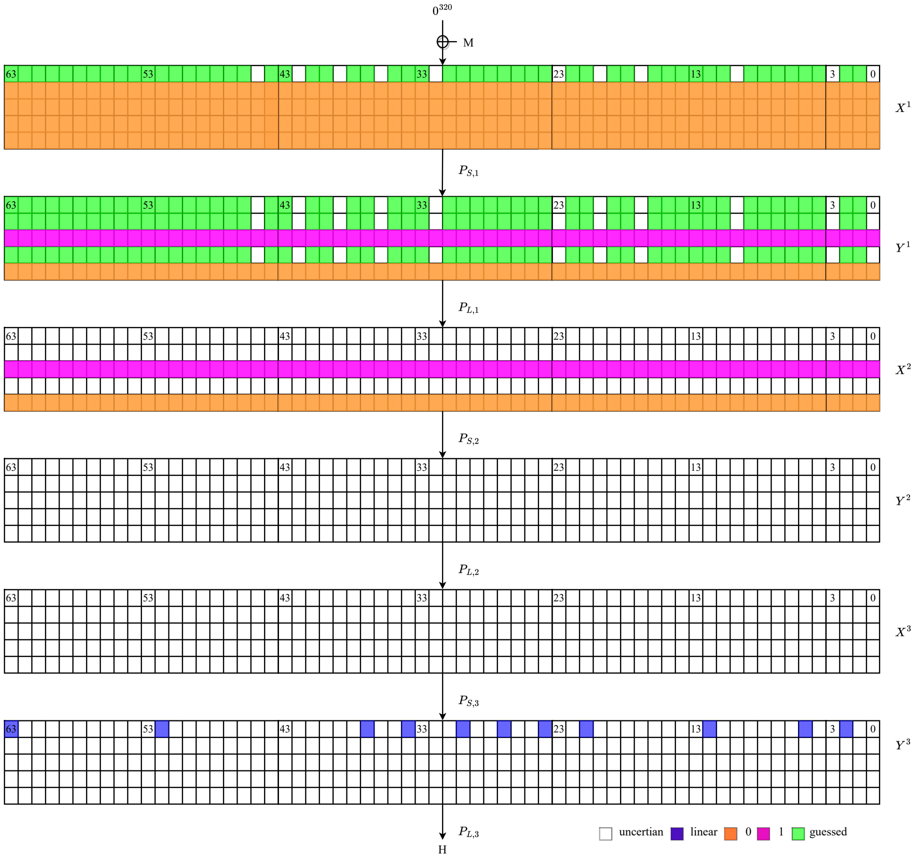

Fig. 8. 3-round preimage attack

4.3 Our Preimage Attack on 3-round Ascon-XOF under Real IV

In this section, we perform a preimage attack on 3-round Ascon-XOF, where the IV is a real value and round constants and initialization are both considered. In fact, the steps of the MILP model of our preimage attack on 3-round Ascon-XOF under a real IV are similar to that of our preimage attack on 3-round Ascon-XOF with zero IV. The difference is that the conditions of each state bit that needs to be guessed or linearized need to follow the expressions obtained by our Sagemath program. Therefore, the MILP modeling process will be omitted here. We can obtain 13 linear equations of Y_0^3 with respect to X_0^1 by just guessing 51 state bits of X_0^1, which means that we can find a preimage of 3-round Ascon-XOF with a complexity of 2^{51}. The results of our preimage attack on 3-round Ascon-XOF under real IV are described in Table 4. The linear expressions for the linearized bits of Y_0^3 are shown in Appendix C, and the initial state S_1^0

Table 4. Results of preimage attacks against 3-round ASCON-XOF under real IV.

Item	Index	#Total bits
Guess bits of X_0^1	0 1 2 4 5 7 8 9 10 11 12 13 16 18 19 20 21 22 23 24 26 27 29 30 31 32 33 34 35 37 38 39 41 42 43 44 46 47 48 49 51 52 54 55 56 57 58 59 60 61 63	51
Linear bits of Y_0^3	5 13 16 19 27 30 35 44 47 49 52 57 58	13

of our preimage attack of 3-round Ascon-XOF is described in Appendix D. Our proposed preimage attack on 3-round Ascon-XOF under real IV is valid because the remaining 13 bits that have not been guessed are all included in the 13 linear equations. Therefore, we can obtain a preimage of the given hash value with a probability of 1 by traversing 2^{51} possible values of guessing bits and solving the corresponding equation systems.

5 Collision Attacks on Reduced Ascon-HASH

When performing collision attacks on SPN-based primitive, we can attack for more rounds if we can linearize the nonlinear layers at an acceptable cost. Linearization of nonlinear layers requires a lot of degrees of freedom. Compared to the collision attack on Keccak proposed by Qiao et al. [16], Ascon's state only has 64 bits of degrees of freedom, making linearization strategy using in [16] less effective in Ascon's collision attacks. In order to effectively use the linearization strategy, we consider the free-starting collision setting and investigate the resistance of Ascon-HASH against collision attacks in this setting. By comprehensively analyzing the properties of Ascon and its S-box, we found that a 2-round connector constructed by applying the linearization of the inverse of S-boxes can extend collision attacks against Ascon-HASH for more rounds under the free-start collision setting. Therefore, in this section, we propose free-start collision attacks on reduced Ascon-HASH based on new differential trails searched by CP [9], followed by the details of constructing 2-round connectors, which play an essential role in our attacks.

5.1 S-Box Linearization

The critical observation is that the internal state of Ascon-HASH is much larger than the digest size, providing a majority of freedom degrees to attackers to launch collision attacks. One can select some subsets of the available spaces with special properties to achieve deterministic differential propagation. As for Ascon, we intend to choose the subsets that can make the S-Box linear, *i.e.*, the expression of the S-box can be re-written as a linear transformation when the inputs are restricted to such subsets. When considering the entire 2^5 input space of the S-box, the S-box is nonlinear. However, affine subspaces with a size equal to or less than 4, as shown in [16], could be found so that the S-box can be linearized. Since the S-box is the only nonlinear part of the Ascon round function. Therefore, the entire round function will become linear when the inputs of all S-boxes are restricted to such subspaces. Formally, we give the following definition and observation.

Definition 1 *(Linearizable affine subspace [16]). Linearizable affine subspaces are affine input subspaces on which S-box substitution can be re-written as a linear transformation. If V denotes a linearizable affine subspace of an S-box operation $S(\cdot)$, $\forall x \in V, S(x) = A \cdot x + b$ where A is a matrix and b is a constant vector.*

For instance, when the input is limited to subset {01001, 11001, 01010, 11010} ({09, 19, 0A, 1A} in hex), the corresponding output set of Ascon's S-box is {00101, 01100, 01000, 00001}({05, 0C, 08, 01} in hex), and its corresponding input difference and output difference are 10 and 09 (in hex) respectively, the corresponding S-box can be re-written as a linear transformation:

$$y = \begin{pmatrix} 1\,0\,0\,0\,1 \\ 0\,0\,1\,0\,0 \\ 1\,0\,0\,0\,0 \\ 1\,0\,0\,1\,1 \\ 1\,1\,0\,0\,0 \end{pmatrix} \cdot x + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{9}$$

Where x and y are bit vector representations of input and output values of the Ascon S-box with the least significant bit on top. As shown in Equation 9, when the input is restricted to a linearizable affine subspace, the S-box can be re-written as a linear transformation, denoted as $y = A \cdot x + b$.

Definition 2 *(The linearization of the inverse of the S-box). When the input is restricted to a linearizable affine subspace, the inverse of the S-box can also be re-written as a linear transformation, expressed as $x = A^{-1} \cdot y + b'$. Where A^{-1} represents the inverse matrix of A, b' represents a constant vector.*

Specifically, Equation 10 denotes the constraints of the input of the inverse of the S-box. When the input is restricted to a linearizable affine subspace, the inverse of the S-box can also be re-written as a linear transformation.

$$\begin{cases} y_0 = 0 \\ y_3 = 0 \\ y_1 \oplus y_4 = 1 \end{cases} \tag{10}$$

Where $[y_0, y_1, y_2, y_3, y_4]^T$ is the vector representation of y.

Observation 1 *[16] Given a 5-bit input difference δ_{in} and a 5-bit output difference δ_{out} with $\mathrm{DDT}(\delta_{in}, \delta_{out}) \neq 0$, denote the solution set $V = \{x : \mathrm{S}(x) + \mathrm{S}(x + \delta_{in}) = \delta_{out}\}$ and $\mathrm{S}(V) = \{\mathrm{S}(x) : x \in V\}$, we have*

1. *if* $\mathrm{DDT}\left(\delta_{in}, \delta_{out}\right) = 2$ *or 4,* V *is a linearizable affine subspace.*
2. *if* $\mathrm{DDT}\left(\delta_{in}, \delta_{out}\right) = 8$*, there are six 2-dimensional subsets* $W_i \subset V, i = 0, 1, ...5$ *such that* $W_i (i = 0, 1, ...5)$ *are linearizable affine subspaces.*

As is well known that there is a one-to-one corresponding relationship between linearizable affine subspaces and entries with 2 or 4 in DDT [11]. As for the DDT entries with value 8, we can deduce 6 2-dimensional linearizable affine subspaces from the 3-dimensional subset, meaning that linearizing the S-box requires at least 3 degrees of freedom. For instance, the 3-dimensional subset with input difference and output difference being 10 and 09 is {09, 19, 0A, 1A, 0D, 1D, 0E, 1E} and the six 2-dimensional linearizable affine subspaces from it are

$$\begin{aligned}
&\{09, 19, 0A, 1A\},\\
&\{09, 19, 0D, 1D\},\\
&\{09, 19, 0E, 1E\},\\
&\{0A, 1A, 0D, 1D\},\\
&\{0A, 1A, 0E, 1E\},\\
&\{0D, 1D, 0E, 1E\}.
\end{aligned} \tag{11}$$

When projected to the whole Ascon state, the direct product of affine subspaces of each S-box forms affine subspaces of the entire state. Therefore, the entire round function becomes linear when all the S-boxes in the round function are linearized. This will be the method that we are to handle the S-box layer of the first round of the 2-round connector.

5.2 A 2-round Connector based on the Linearization of the Inverse of S-boxes

The important observation is that the internal state of Ascon-HASH is much larger than the digest size, providing a larger number of freedom degrees to attackers to mount collision attacks. Therefore, we perform a collision attack on 3-round Ascon-HASH based on this observation. Firstly, We establish a 2-round connector between the permutations of Ascon. The main idea of our 2-round connector is to transform the problem into solving a system of linear equations. Two rounds of Ascon permutation can be expressed as $P_{L3} \circ P_{S3} \circ P_{L2} \circ P_{S2}$ (omitting the P_C). The layer P_{S2} can be linearized by the method discussed in Section 5.1, *i.e.*, given both the input differences and output differences of P_{S2}, the operations $P_{L2} \circ P_{S2}$ can become linear when the input values are restricted to the linearizable affine subspace. At the same time, the differences at layer P_{S3} can also propagate with a probability of 1 when the inputs are limited to a specific set.

Next, we will show how the layers P_{S2} and P_{S3} can propagate with probability 1 simultaneously. In the first step, we can add linear equations to the input values (denoted as X^3) of round 3 to achieve deterministic difference propagation in round 3 (from X^3 to Y^3). Since the output difference of ΔY^3 is given, if we restrict the input value of X^3 to the specific set based on the method described in Section 5.1, we can achieve deterministic difference propagation of round 3. As for round 3, there are 44 active S-boxes. Therefore, there are a total of 134 constraint equations for the input value according to the DDT of Ascon's S-box and the 3-round differential trail shown in Fig.9, denoted as:

$$A \cdot X^3 = b_2 \tag{12}$$

Where A denotes a matrix of 134 rows and 320 columns, X^3 denotes a vector of 320 rows and 1 column, representing the input value of round 3, and b_2 denotes a vector of 134 rows and 1 column. In addition, there is a linear layer in the transformation from the output of round 2 (Y^2) to the input of round 3 (X^3), denoted as L_2. Indeed, the transformation of the L_2 can be considered as a matrix computation, written as $L_2 \cdot Y^2 = X^3$. Where L_2 denotes a matrix of 320 rows and 320 columns, Y^2 denotes a vector of 320 rows and 1 column.

Moreover, in round 2 (from X^2 to Y^2), there are 33 active S-boxes. According to [2,3], for any pair of $(\delta_{in},\delta_{out})$, the solution set $V = \{x : \mathrm{S}(x) + \mathrm{S}\left(x + \delta_{\text{in}}\ \right) = \delta_{\text{out}}\}$ forms an affine subspace. In other words, V can be deduced from the set $\{0,1\}^5$ by setting up i constraints when the size of V is 2^{5-i}. Since the number of DDT(0C,08) is 8, there are six 2-dimensional subsets $W_i \subset V, i = 0, 1, ...5$ such that $W_i(i = 0, 1, ...5)$ are linearizable affine subspces according to the property described in Section 5.1. We can randomly select one from these linearizable affine subspaces, which can be deduced from the set $\{0,1\}^5$ by setting up i constraints that turn to be binary linear equations.

In general, these constraints of input values can be written in the form of linear equations. Therefore, The constraints on the input values of the round 2 can be denoted as:

$$A_1 \cdot X^2 = b_1 \tag{13}$$

Where A_1 denotes a matrix of 104 rows and 320 columns, X^2 denotes a vector of 320 rows and 1 column, representing the input value of round 2, and b_1 denotes a vector of 104 rows and 1 column. Given the input difference and output difference of round 2, all active S-boxes in round 2 can be re-written as a linear transformation under the corresponding i linear constraints on the input values, denoted as:

$$L_0^{-1} \cdot Y^2 + b_0' = X^2 \tag{14}$$

Here, L_0^{-1} denotes a matrix that can be deduced while the input difference and output difference of round 2 are given and b_0' represents a constant vector. By applying $L_2 \cdot Y^2 = X^3$ into Equation 12, we can obtain Equation 15 as follows:

$$A \cdot L_2 \cdot Y^2 = b_2 \tag{15}$$

Moreover, by substituting Equation 14 into Equation 13, we can obtain Equation 16 as follows:

$$A_1 \cdot L_0^{-1} \cdot Y^2 = b_1' \tag{16}$$

In fact, the solutions of the linear equation systems composed of Equation 15 and Equation 16 will be the solutions of the 2-round connectors. By restricting the input values of the 2-round connector to the solution sets, the input difference can propagate to the output difference with a probability of 1. For simplicity, we use E_S to denote the system of linear equations.

5.3 Free-start Collision Attack on 3-round Ascon-HASH

Zong et al. [18] proposed a collision attack on 2-round Ascon-HASH with a complexity of 2^{125} by using a differential trail with a probability of 2^{-199}. Gerault et al. [10] obtained an improved differential trail with a probability of 2^{-156} by using CP, decreasing the complexity of the collision attack of 2-round Ascon-HASH from 2^{125} to 2^{103}. Yu et al. [17] performed a practical collision attack on 2-round Ascon-HASH based on some critical observations on the round function of Ascon, its complexity is $2^{62.6}$. The commonality between these algorithms is that both the input difference and the output difference of the differential trail exist only in the rate part, and then the output difference is canceled by XORing the message pair in the next block.

However, in the free-start setting, the input difference may exist in the IV of the hash function, based on which we can mount a collision attack on Ascon-HASH with the input difference in the IV but not the rate part. We perform a collision attack on 3-round Ascon-HASH based on a new 3-round differential trail. The details of the 3-round differential used in our attack are described in Fig.9.

As shown in Fig.9, the first round has 5 active S-boxes with a probability of 2^{-13}, and the second round has 33 active S-boxes with a probability of 2^{-90}. The total probability of the 3-round differential trail is 2^{-237}. Furthermore, according to the method described in Section 5.2, round 2 and round 3 of the 3-round differential trail can be constructed as a 2-round connector, through which the input difference of the round 2 can propagate to the output of round 3 deterministically. Therefore, the 3-round differential trail used by our attack is built by a 2-round connector and a 1-round differential trail with a probability of 2^{-13}. The differential trail used by our collision attack on 3-round Ascon-HASH is summarized in Table 5. We adjust the value of Y^2 to satisfy the conditions of the 2-round connector so that round 2 and round 3 can propagate the difference with a probability of 1. The configuration of our collision attack on 3-round Ascon-HASH is shown in Fig.10.

As shown in Fig.10, we apply the 3-round differential trail described above to the permutation (denoted as p^3) in the initialization phase. After absorbing a pair of messages with a difference equal to the difference in the output of the 3-round differential trail in the absorbing phase, we can get collisions in the squeezing phase. As the required freedom of degree in step S1 is 2^{13}, therefore, 2^{13} degrees of freedom is enough for our collision attack to find Y^2 and Y^{2*} that satisfy the 3-round differential trail. For the attack process, we define a Boolean function:

$$f : F_2^{13} \rightarrow F_2. \tag{17}$$

If we construct $((IV, M_1), (IV^*, M_1^*))$ based on Y^2 that satisfies $f(Y^2) = 1$, the pair will lead to a collision. The function $f(Y^2)$ can be constructed as follows:

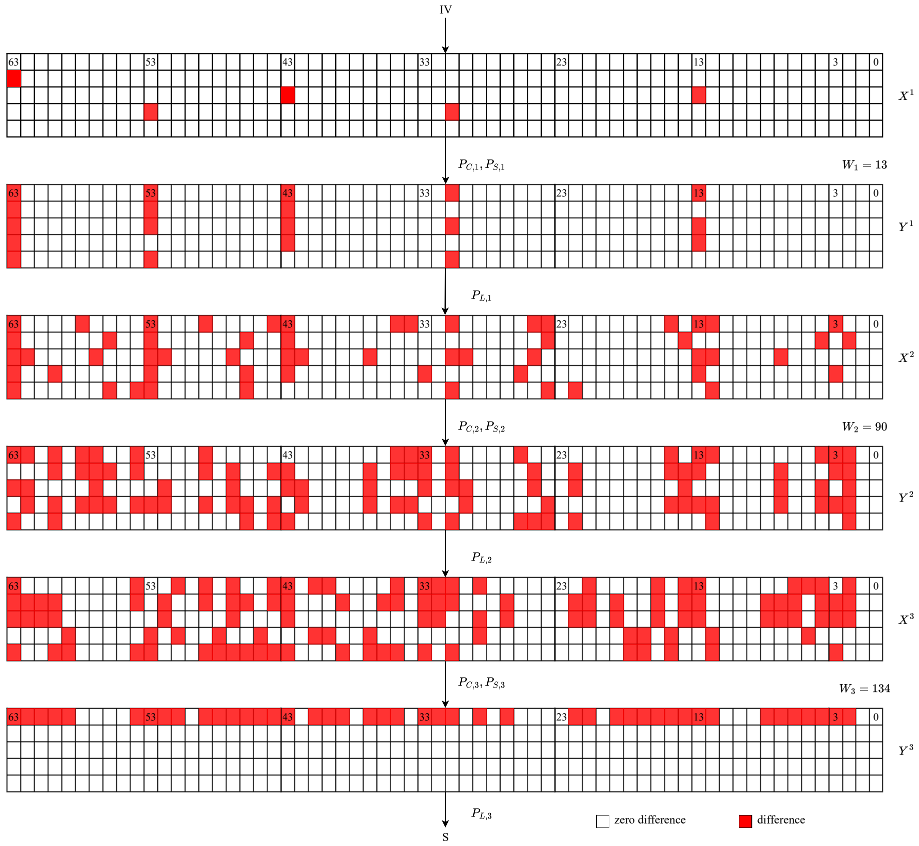

Fig. 9. 3-round differential trail of Ascon-HASH

S1 Compute Y^1, Y^{1*} by applying the operation $P_L^{-1}(P_S^{-1}(Y^2)), P_L^{-1}(P_S^{-1}(Y^{2*}))$ on (Y^2, Y^{2*}) that satisfies $A \cdot L_2 \cdot Y^2 = b_2$ and $A_1 \cdot L_0^{-1} \cdot Y^2 = b_1'$.

S2 Compute X^1, X^{1*} by applying the operation $P_S^{-1}(Y^1), P_S^{-1}(Y^{1*})$ and check whether $\Delta X^1 = X^1 \oplus X^{1*}$ holds.

S3 Compute Y^3, Y^{3*} by applying the operation $P_S(P_L(Y^2)), P_S(P_L(Y^{2*}))$ on (Y^2, Y^{2*}) and check whether $\Delta Y^3 = Y^3 \oplus Y^{3*}$ holds.

S4 If (Y^2, Y^{2*}) satisfies all steps, $f(Y^2)$ return 1; otherwise return 0.

Analysis of Degree of Freedom. The degree of freedom of the solution space of E_S is a crucial factor for the success of our collision attack on 3-round Ascon-HASH. In other words, if the degree of freedom of the solution space is larger

Table 5. Our differential trail on 3-round Ascon-HASH

Round(r)	Probability	Input difference(ΔX^r)	Output difference(ΔY^r)
1	2^{-13}	0000000000000000 8000000000000000 0000080000002000 0020000080000000 0000000000000000	8020080080002000 8020080000000000 8020080080002000 8000080000002000 8020000080000000
2	2^{-90}	8422180c8300b008 8120480001004014 c2308c20c2003080 9020480204002008 8160400081401000	d642000e8400b01c 1742982e8240e08c c2108c22c2404094 5772d42c4100f09c 9000580287401004
3	2^{-134}	804a9b0ba0216074 f052d89b886961dc f0108803286961dc 0829232020069020 9863f8ba80069008	f87bfbbba86ff1fc 0000000000000000 0000000000000000 0000000000000000 0000000000000000

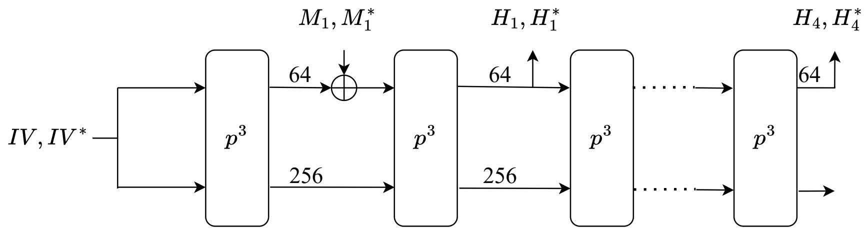

Fig. 10. Our attack configuration on 3-round Ascon-HASH

than the weight of the 3-round differential trail, it indicates that there is at least one pair of messages having the same digest. As for round 1 of our 3-round differential trail, each active S-box needs to be linearized, with constraint equations added to its input values. The degree of freedom of round 2 can be calculated as $\sum_{i=0}^{63} D_i^{(2)}$, where $D_i^{(2)}$ is the degree of freedom of the 5-bit input space of the *i-th* S-box in round 2. The definition of $D_i^{(2)}$ is as follows. If DDT$(\delta_{in}, \delta_{out})$ is equal to 4 or 8, $D_i^{(2)}$ is equal to 2. If DDT$(\delta_{in}, \delta_{out})$ is equal to 2, $D_i^{(2)}$ is equal to 1. If DDT$(\delta_{in}, \delta_{out})$ is equal to 0, $D_i^{(2)}$ is equal to 5, where δ_{in} and δ_{out} denote the input and output differences of the *i-th* S-box.

Another reduction in degrees of freedom is due to constraints on the input values of the S-boxes in round 3. Each active S-box in round 3 can deterministically propagate the input difference to the output difference by adding constraint equations to its input values. Therefore, the degree of freedom of round 3 can be calculated as $\sum_{i=0}^{63} D_i^{(3)}$, where $D_i^{(3)}$ is the degree of freedom of the 5-bit input space of the *i-th* S-box in round 3. If $\mathrm{DDT}(\delta_{in}, \delta_{out})$ is equal to 8, $D_i^{(3)}$ is equal to 3. If $\mathrm{DDT}(\delta_{in}, \delta_{out})$ is equal to 4, $D_i^{(3)}$ is equal to 2. If $\mathrm{DDT}(\delta_{in}, \delta_{out})$ is equal to 2, $D_i^{(3)}$ is equal to 1. if $\mathrm{DDT}(\delta_{in}, \delta_{out})$ is equal to 0, $D_i^{(3)}$ is equal to 5. For the *i-th* S-box in round 3, we add (5 - $D_i^{(3)}$) equations to E_S. As a result, The degree of freedom of the E_S can be calculated as follows:

$$D = \sum_{i=0}^{63} D_i^{(2)} - \sum_{i=0}^{63} (5 - D_i^{(3)}) \tag{18}$$

According to the 3-round differential trail used in our collision attack, the degree of freedom D equals 82. Since $D(82)$ is larger than the weight(13) of the 3-round differential trail used in our attack, the 3-round Ascon-HASH collision attack we proposed can theoretically obtain collisions.

Complexity Analysis. According to the 3-round differential trail described in Table 5, the weight of round 1 of the 3-round differential trail is 13, which is also the weight of the 3-round differential trail. Therefore, in the classical computer setting, our proposed 3-round collision attack on Ascon-HASH using 2-round connectors has a hash complexity of 2^{13}. The results are shown in Table 1 and the details of the attack process are as follows:

1. Generate a total of 2^{13} pairs of messages Y^2 and $Y^{2*} = Y^2 \oplus \mathit{\Delta} Y^2$ that satisfies the conditions of $A \cdot L_2 \cdot Y^2 = b_2$ and $A_1 \cdot L_0^{-1} \cdot Y^2 = b_1'$.

2. Compute Y^1, Y^{1*} through the operation $P_L^{-1}(P_S^{-1}(Y^2)), P_L^{-1}(P_S^{-1}(Y^{2*}))$ on (Y^2, Y^{2*}). Because the probability of the 2-round connector is 1, there are 2^{13} pairs of messages (Y^1, Y^{1*}) satisfying $Y^1 \oplus Y^{1*} = \mathit{\Delta} Y^1$.

3. Compute X^1, X^{1*} through the operation $P_S^{-1}(Y^1), P_S^{-1}(Y^{1*})$.

4. With a probability of 2^{-13}, a pair of messages (X^1, X^{1*}) will satisfy the constraint for the first round. Thus, we will have, on average, one message pair that satisfies the input difference $\mathit{\Delta}_{in}$.

5. Apply a random message block M_1 and $M_1^* = M_1 \oplus \mathit{\Delta}_{out}$ to the $(IV,\ IV^*)$ selected at the end of Step 4 and $((IV, M_1),\ (IV^*, M_1^*))$ will leads to a collision.

The complexity of the attack process above is $2 \cdot 2^{13} = 2^{14}$ hash function calls. As shown in Table 1, our proposed collision attack is more superior. Compared to the collision attack proposed by Zong et al. [18], the collision attack proposed by Gerault et al. [10] and the collision attack proposed by Yu et al. [17], our proposed collision attack on Ascon-HASH can attack one more round and the corresponding complexity is just 2^{14}.

5.4 Free-start Collision Attack on 4-round Ascon-HASH

Here we propose a free-start collision attack against 4-round Ascon-HASH, which is performed based on a new 4-round differential trail searched by CP with a complexity of 2^{21}. Its main idea is to construct a 2-round connector by applying the linearization of the inverse of S-boxes so that the differential propagation probability of the last two rounds is 1.

A 2-round Connector for 4-round Collision Attack. Similar to the 3-round free-start collision attack described in 5.3, our proposed 4-round free-start collision attack can also construct a 2-round connector in the last two rounds. Therefore, the differences in the layers P_{S4} and P_{S3} can propagate with probability 1 simultaneously.

Since the output difference $\varDelta Y^4$ is given, if we restrict the input values of X^4 to the specific set by adding constraint equations, we can achieve deterministic difference propagation of round 4 (from X^4 to Y^4). As for round 4, there are 44 active S-boxes and a total of 133 constraint equations for the input values, which can be written as:

$$A_4 \cdot X^4 = b_4 \tag{19}$$

Where A_4 denotes a matrix of 133 rows and 320 columns, X^4 denotes a vector of 320 rows and 1 column, representing the input values of round 4, and b_4 denotes a vector of 133 rows and 1 column.

As for round 3 (33 active S-boxes), each active S-box can be re-written as a linear transformation under the corresponding linear constraints on the input values. All equations constraining the input values (X^3) of active S-boxes in round 3 can be merged, denoted as:

$$A_3 \cdot X^3 = b_3 \tag{20}$$

Where A_3 denotes a matrix of 113 rows and 320 columns, X^3 denotes a vector of 320 rows and 1 column, representing the input values of round 3, and b_3 denotes a vector of 113 rows and 1 column. The linear transformations of all active S-boxes in round 3 can also be merged, denoted as:

$$L_3^{-1} \cdot Y^3 + b_3' = X^3 \tag{21}$$

Here, L_3^{-1} denotes a matrix that can be deduced while the input difference and output difference of round 3 are given and b_3' represents a constant vector. Additionally, there is a linear layer in the transformation from the output of round 3 (denoted as Y^3) to the input of round 4 (X^4), denoted as L_{34}. By applying $L_{34} \cdot Y^3 = X^4$ into Equation 19, we can obtain Equation 22 as follows:

$$A_4 \cdot L_{34} \cdot Y^3 = b_4 \tag{22}$$

Moreover, by substituting Equation 21 into Equation 20, we can obtain Equation 23 as follows:

$$A_3 \cdot L_3^{-1} \cdot Y^3 = b_3'' \tag{23}$$

In fact, the solutions of the linear equation systems composed of Equation 22 and Equation 23 will be the solutions of the 2-round connector. By restricting the input values of the 2-round connector to the solution sets, the input difference of the 2-round connector can propagate to the output difference of the 2-round connector with a probability of 1. For simplicity, we use E_M to denote the linear equation systems.

Our Collision Attack on 4-round Ascon-HASH. Here we perform a collision attack on 4-round Ascon-HASH based on a new 4-round differential trail. The 4-round differential trail used by our free-start collision attack on 4-round Ascon-HASH is summarized in Table 6 and Appendix E. The first round has 1 active S-box, with a probability of 2^{-2} and the second round has 7 active S-boxes, with a probability of 2^{-18}. The third round has 33 active S-boxes, with a probability of 2^{-110} and the fourth round has 44 active S-boxes, with a probability of 2^{-133}. Therefore, the total probability of the 4-round differential trail is 2^{-263}.

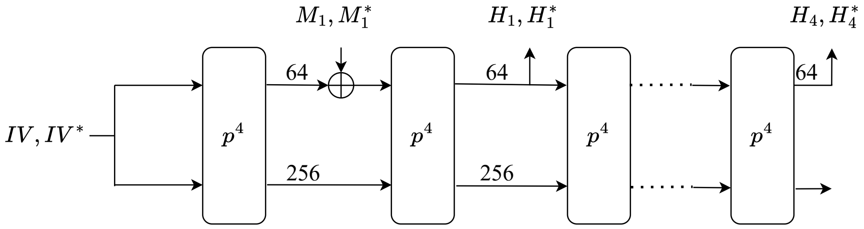

Fig. 11. Our attack configuration on 4-round Ascon-HASH

According to the method described in section 5.2, round 3 and round 4 can be constructed as a 2-round connector, through which the input difference of round 3 can propagate to the output difference of round 4 with a probability of 1. Therefore, the 4-round differential trail used by our attack is built by a 2-round connector and a 2-round differential trail with a probability of 2^{-20}. We adjust the value of Y^3 to satisfy the conditions of the 2-round connector so that the differences of round 3 and round 4 can propagate with a probability of 1.

The configuration of our free-start collision attack on 4-round Ascon-HASH is shown in Fig.11. We apply the 4-round differential trail described above to the permutation (denoted as p^4) in the initialization phase. After absorbing a pair of messages with a difference equal to the difference in the output of the 4-round differential trail in the absorbing phase, we can get collisions in the squeezing phase.

As the required freedom of the degree in Step S1 is 2^{20}, therefore, 2^{20} degrees of freedom is enough for our collision attack to find Y^3 and Y^{3*} that satisfy the

Table 6. Our differential trail on 4-round Ascon-HASH

Round(r)	Probability	Input difference(ΔX^r)	Output difference(ΔY^r)
1	2^{-2}	0000000000000000	0000000020000000
		0000000000000000	0000000000000000
		0000000020000000	0000000020000000
		0000000000000000	0000000020000000
		0000000000000000	0000000000000000
2	2^{-18}	0000000020000402	0000000030881000
		0000000000000000	0000000020880402
		0000000030800000	0000000030881000
		0000000020081000	0000000010880402
		0000000000000000	0000000020081402
3	2^{-110}	0a81000030881612	0290000128040005
		0041100920c82412	0c10140b39c6002b
		00000000280e3840	00c0100139ce161a
		02810000108c2e47	0040100108421221
		0410040a2148042a	0c411403314c3a6f
4	2^{-133}	02d0a00301042517	fbf3b877a1eea7df
		7ae33852a1ee215b	0000000000000000
		68a31841a1ce254f	0000000000000000
		8110802500008684	0000000000000000
		d3c1303600a88291	0000000000000000

4-round differential trail. For the attack process, we define a Boolean function:

$$f : F_2^{20} \to F_2. \tag{24}$$

If we construct $((IV, M_1), (IV^*, M_1^*))$ based on Y^3 that satisfies $f(Y^3) = 1$, then the pair leads to a collision. The function $f(Y^3)$ can be constructed as follows:

S1 Compute Y^2, Y^{2*} by applying the operation $P_L^{-1}(P_S^{-1}(Y^3)), P_L^{-1}(P_S^{-1}(Y^{3*}))$ on (Y^3, Y^{3*}) that satisfies $A_4 \cdot L_{34} \cdot Y^3 = b_4$ and $A_3 \cdot L_3^{-1} \cdot Y^3 = b_3''$.

S2 Compute X^2, X^{2*} by applying the operation $P_S^{-1}(Y^2), P_S^{-1}(Y^{2*})$ on (Y^2, Y^{2*}) and check whether $\Delta X^2 = X^2 \oplus X^{2*}$ holds.

S3 Compute X^1, X^{1*} by applying the operation $(P_L^{-1}(P_S^{-1}(X^2)), (P_L^{-1}(P_S^{-1}(X^{2*}))$ on (X^2, X^{2*}) and check whether $\Delta X^1 = X^1 \oplus X^{1*}$ holds.

S4 Compute Y^4, Y^{4*} by applying the operation $P_S(P_L(Y^3)), P_X(P_L(Y^{3*}))$ on (Y^3, Y^{3*}) and check whether $\Delta Y^4 = Y^4 \oplus Y^{4*}$ holds.

S5 If(Y^3, Y^{3*}) satisfies all steps, $f(Y^3)$ return 1; otherwise return 0.

Analysis of Degree of Freedom. The degree of freedom of the solution space of E_M is a crucial factor for the success of our collision attack on 4-round Ascon-HASH. If the degree of freedom of the solution space is larger than the weight of the 4-round differential trail, it indicates that there is at least one pair of messages having the same digest. As for round 3 of our 4-round differential trail, each active S-box needs to be linearized, with constraint equations added to its input values. The degree of freedom can be calculated as $\sum_{i=0}^{63} F_i^{(3)}$ after the linearization of round 3, in which $F_i^{(3)}$ is the degree of freedom of the 5-bit input space of the *i-th* S-box in round 3. Here the definition of $F_i^{(3)}$ is the same as the definition of $\sum_{i=0}^{63} D_i^{(2)}$ described in Section 5.3.

Another decrease in the degree of freedom is due to the constraints on the input values of round 4. The definition of $\sum_{i=0}^{63} F_i^{(4)}$, the degree of freedom of 5-bit input values to S-boxes in round 4, is the same as the definition of $\sum_{i=0}^{63} D_i^{(3)}$ described in Section 5.3. For the *i-th* S-box in round 4, we add $(5-F_i^{(4)})$ equations to E_M. The degree of freedom of the final E_M can be calculated as follows:

$$N = \sum_{i=0}^{63} F_i^{(3)} - \sum_{i=0}^{63}(5 - F_i^{(4)})) \tag{25}$$

According to the 4-round differential trail used in our collision attack, the degree of freedom F equals 74. Since $F(74)$ is larger than the weight (20) of the 4-round differential trail used in our attack, our proposed collision attack against 4-round Ascon-HASH can get collisions in theory.

Complexity Analysis. According to the 4-round differential trail described in Table 6, the weight of round 1 and round 2 of the 4-round differential trail is 20. Therefore, our proposed 4-round collision attack using a 2-round connector has a complexity of $2 \cdot 2^{20} = 2^{21}$.

5.5 Free-start Collision Attack on 5-round Ascon-HASH

Here we propose a free-start collision attack on 5-round Ascon-HASH based on a new 5-round differential trail searched dedicatedly, followed by the details of constructing a 2-round connector, which plays an essential role in our attack. Its time complexity is 2^{41}.

A 2-round Connector for 5-round Collision Attack. Similar to the 3-round free-start collision attack described in 5.3, our proposed 5-round free-start collision attack can also construct a 2-round connector in the last two rounds. Therefore, the differences in the layers P_{S5} and P_{S4} can propagate with probability 1 simultaneously.

Since the output difference ΔY^5 is given, if we restrict the input values of X^5 to the specific set by adding constraint equations, we can achieve deterministic difference propagation of round 5 (from X^5 to Y^5). As for round 5, there are

41 active S-boxes and a total of 133 constraint equations for the input values, which can be written as:

$$A_5 \cdot X^5 = b_5 \tag{26}$$

Where A_5 denotes a matrix of 133 rows and 320 columns, X^5 denotes a vector of 320 rows and 1 column, representing the input values of round 5, and b_5 denotes a vector of 133 rows and 1 column.

As for round 4 (40 active S-boxes), each active S-box can be re-written as a linear transformation under the corresponding linear constraints on the input values. All equations constraining the input values (X^4) of active S-boxes in round 4 can be merged, denoted as:

$$A_6 \cdot X^4 = b_6 \tag{27}$$

Where A_6 denotes a matrix of 138 rows and 320 columns, X^4 denotes a vector of 320 rows and 1 column, representing the input values of round 4, and b_6 denotes a vector of 138 rows and 1 column. The linear transformations of all active S-boxes in round 4 can also be merged, denoted as:

$$L_4^{-1} \cdot Y^4 + b_6' = X^4 \tag{28}$$

Here L_4^{-1} denotes a matrix that can be deduced while the input difference and output difference of round 4 are given and b_6' represents a constant vector. Additionally, there is a linear layer in the transformation from the output of round 4 (denoted as Y^4) to the input of round 5 (X^5), denoted as L_{45}. By applying $L_{45} \cdot Y^4 = X^5$ to Equation 26, we can obtain Equation 29 as follows:

$$A_6 \cdot L_{45} \cdot Y^4 = b_5 \tag{29}$$

Moreover, by substituting Equation 28 into Equation 27, we can obtain Equation 30 as follows:

$$A_6 \cdot L_6^{-1} \cdot Y^4 = b_6'' \tag{30}$$

In fact, the solutions of the linear equation systems composed of Equation 29 and Equation 30 will be the solutions of the 2-round connector. By restricting the input values of the 2-round connector to the solution sets, the input difference of the 2-round connector can propagate to the output difference of the 2-round connector with a probability of 1. For simplicity, we use E_N to denote the linear equation systems.

Our Collision Attack on 5-round Ascon-HASH. Here we perform a collision attack on 5-round Ascon-HASH based on a new 5-round differential trail. The 5-round differential trail used by our free-start collision attack on 5-round Ascon-HASH is summarized in Table 7 and Appendix F. The first round has 1 active S-box, with a probability of 2^{-2} and the second round has 3 active S-boxes, with a probability of 2^{-6}. The third round has 11 active S-boxes, with a probability of 2^{-32}, the fourth round has 40 active S-boxes, with a probability

Table 7. Our differential trail on 5-round Ascon-HASH

Round(r)	Probability	Input difference(ΔX^r)	Output difference(ΔY^r)
1	2^{-2}	0000000000000000	0000000000080000
		0000000000080000	0000000000000000
		0000000000080000	0000000000000000
		0000000000000000	0000000000000000
		0000000000000000	0000000000000000
2	2^{-6}	0080000000080001	0080000000080001
		0000000000000000	0080000000080001
		0000000000000000	0000000000000000
		0000000000000000	0000000000000000
		0000000000000000	0000000000000000
3	2^{-32}	0000200008080000	0400200000480008
		0480100002490009	040020000a080009
		0000000000000000	0480100002490009
		0000000000000000	0480100002490009
		0000000000000000	0000000008000000
4	2^{-127}	00812000444a0001	a4d382044d6bb36b
		2415300048400001	2481a004045a2209
		a2d218400364a40d	0006080041653561
		86c5b2440a49936d	048628444d24250c
		0004000008100000	a2c0b2440251936c
5	2^{-133}	0405004470131a84	5d851fed7cd77b97
		0884146034c23100	0000000000000000
		0405142060d23b04	0000000000000000
		55010b8d48154a97	0000000000000000
		59841be93c045013	0000000000000000

of 2^{-127} and the fifth round has 41 active S-boxes, with a probability of 2^{-133}. Therefore, the total probability of the 5-round differential trail is 2^{-300}.

According to the method described in Section 5.2, round 4 and round 5 can be constructed as a 2-round connector, through which the input difference of round 4 can propagate to the output difference of round 5 with a probability of 1. Therefore, the 5-round differential trail used by our attack is built by a 2-round connector and a 3-round differential trail with a probability of 2^{-40}. We adjust the value of Y^4 to satisfy the conditions of the 2-round connector so that the differences of round 4 and round 5 can propagate with a probability of 1.

The configuration of our free-start collision attack on 5-round Ascon-HASH is shown in Fig.12. We apply the 5-round differential trail described above to the permutation (denoted as p^5) in the initialization phase. After absorbing a pair of messages with a difference equal to the difference in the output of the 5-round

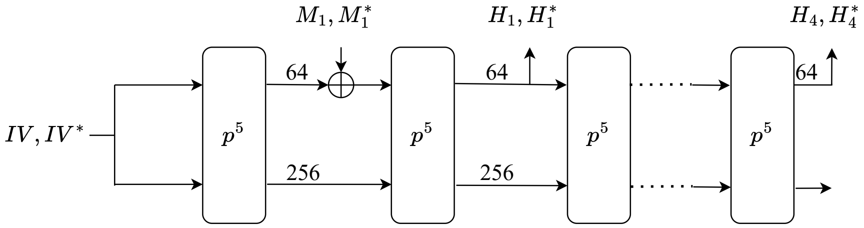

Fig. 12. Our attack configuration on 5-round Ascon-HASH

differential trail in the absorbing phase, we can get collisions in the squeezing phase.

As the required freedom of the degree in Step S1 is 2^{40}, therefore, 2^{40} degrees of freedom is enough for our collision attack to find Y^4 and Y^{4*} that satisfy the 5-round differential trail. For the attack process, we define a Boolean function:

$$f : F_2^{40} \to F_2. \tag{31}$$

If we construct $((IV, M_1), (IV^*, M_1^*))$ based on Y^4 that satisfies $f(Y^4) = 1$, then the pair leads to a collision. The function $f(Y^4)$ can be constructed as follows:

S1 Compute Y^3, Y^{3*} by applying the operation $P_L^{-1}(P_S^{-1}(Y^4)), P_L^{-1}(P_S^{-1}(Y^{4*}))$ on (Y^4, Y^{4*}) that satisfies $A_6 \cdot L_{45} \cdot Y^4 = b_5$ and $A_6 \cdot L_4^{-1} \cdot Y^4 = b_6''$.

S2 Compute X^3, X^{3*} by applying the operation $P_S^{-1}(Y^3), P_S^{-1}(Y^{3*})$ on (Y^3, Y^{3*}) and check whether $\varDelta X^3 = X^3 \oplus X^{3*}$ holds.

S3 Compute X^2, X^{2*} through the operation $(P_S^{-1}(P_L^{-1}(X^3)), (P_S^{-1}(P_L^{-1}(X^{3*}))$ on (X^3, X^{3*}) and check whether $\varDelta X^2 = X^2 \oplus X^{2*}$ holds.

S4 Compute X^1, X^{1*} through the operation $(P_S^{-1}(P_L^{-1}(X^2)), (P_S^{-1}(P_L^{-1}(X^{2*}))$ on (X^2, X^{2*}) and check whether $\varDelta X^1 = X^1 \oplus X^{1*}$ holds.

S5 Compute Y^5, Y^{5*} by applying the operation $P_S(P_L(Y^4)), P_S(P_L(Y^{4*}))$ on (Y^4, Y^{4*}) and check whether $\varDelta Y^5 = Y^5 \oplus Y^{5*}$ holds.

S6 If(Y^4, Y^{4*}) satisfies all steps, $f(Y^4)$ return 1; otherwise return 0.

Analysis of Degree of Freedom. The degree of freedom of the solution space of E_N is a crucial factor for the success of our collision attack on 5-round Ascon-HASH. As for round 4 of our 5-round differential trail, each active S-box needs to be linearized, with constraint equations added to its input values. The degree of freedom can be calculated as $\sum_{i=0}^{63} N_i^{(4)}$ after the linearization of round 4, in which $N_i^{(4)}$ is the degree of freedom of the 5-bit input space of the *i-th* S-box in round 4. Here the definition of $N_i^{(4)}$ is the same as the definition of $\sum_{i=0}^{63} D_i^{(2)}$ described in Section 5.3.

Another decrease in the degree of freedom is due to the constraints on the input values of round 5. The definition of $\sum_{i=0}^{63} N_i^{(5)}$, the degree of freedom of 5-bit input values to S-boxes in round 5, is the same as the definition of $\sum_{i=0}^{63} D_i^{(3)}$ described in Section 5.3. For the *i-th* S-box in round 5, we add $(5-N_i^{(5)})$ equations to E_N. The degree of freedom of the final E_N can be calculated as follows:

$$N = \sum_{i=0}^{63} N_i^{(4)} - \sum_{i=0}^{63}(5 - N_i^{(5)})) \tag{32}$$

According to the 5-round differential trail used in our collision attack, the degree of freedom N equals 49. Since $N(49)$ is larger than the weight (40) of the 5-round differential trail used in our attack, our proposed collision attack against 5-round Ascon-HASH can get collisions in theory.

Complexity Analysis. According to the 5-round differential trail described in Table 7, the weight of round 1, round 2 and round 3 of the 5-round differential trail is 40. Therefore, our proposed 5-round collision attack using a 2-round connector has a complexity of $2 \cdot 2^{40} = 2^{41}$.

6 Conclusions

In this paper, we present a security analysis of hashing modes Ascon-HASH and Ascon-XOF of Ascon, the winner of the NIST Lightweight Cryptography Project. Firstly, we propose an improved preimage attack against 2-round Ascon-XOF-64 with a complexity of 2^{32} via a better guessing strategy. Compared to the attack proposed by Dobrauning et al., our proposed preimage attack on 2-round Ascon-XOF-64 can reduce the complexity from 2^{39} to 2^{32}. Secondly, in order to find a good guessing strategy efficiently, we build a MILP model and successfully extend the attack to 3 rounds of Ascon-XOF-64. The time complexity is 2^{53} when $IV = 0$, while for the real IV, the attack still works and the time complexity is 2^{51}. Thirdly, we also propose a practical free-start collision attack on 3-round Ascon-HASH based on a new 3-round differential trail searched by CP with a complexity of 2^{14}. Its main idea is to construct a 2-round connector using the linearization of the inverse of S-boxes. Last but not least, we construct different 2-round connectors using the linearization of the inverse of S-boxes and successfully extend the collision attack to 4 rounds and 5 rounds of Ascon-HASH with complexities of 2^{21} and 2^{41} respectively.

Although our attacks can not extend to the full 12-round Ascon-HASH and Ascon-XOF, we are convinced that our works will provide new insights into Ascon's security and future works on Ascon.

References

1. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Sponge functions. In: ECRYPT hash workshop. vol. 2007 (2007)
2. Daemen, J.: Cipher and hash function design strategies based on linear and differential cryptanalysis. Ph.D. thesis, Doctoral Dissertation, March 1995, KU Leuven (1995)
3. Dinur, I., Dunkelman, O., Shamir, A.: New attacks on keccak-224 and keccak-256. In: Fast Software Encryption: 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers. pp. 442–461. Springer (2012)
4. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1. 2. submission to round 3 of the caesar competition (2016)
5. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1. 2: submission to nist, may 2021. URL: https://csrc. nist. gov/CSRC/media/Projects/lightweightcryptography/documents/finalist-round/updated-spec-doc/ascon-spec-final. pdf
6. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Cryptanalysis of ascon. In: Topics in Cryptology—CT-RSA 2015: The Cryptographer's Track at the RSA Conference 2015, San Francisco, CA, USA, April 20-24, 2015. Proceedings. pp. 371–387. Springer (2015)
7. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Preliminary analysis of ascon-xof and ascon-hash (2019)
8. Erlacher, J., Mendel, F., Eichlseder, M.: Bounds for the security of ascon against differential and linear cryptanalysis. IACR Transactions on Symmetric Cryptology pp. 64–87 (2022)
9. Gerault, D., Minier, M., Solnon, C.: Constraint programming models for chosen key differential cryptanalysis. In: Principles and Practice of Constraint Programming: 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings 22. pp. 584–601. Springer (2016)
10. Gerault, D., Peyrin, T., Tan, Q.Q.: Exploring differential-based distinguishers and forgeries for ascon. Cryptology ePrint Archive (2021)
11. Guo, J., Jean, J., Nikolić, I., Qiao, K., Sasaki, Y., Sim, S.M.: Invariant subspace attack against midori64 and the resistance criteria for s-box designs. Cryptology ePrint Archive (2016)
12. Liu, M., Lu, X., Lin, D.: Differential-linear cryptanalysis from an algebraic perspective. In: Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part III. pp. 247–277. Springer (2021)
13. Makarim, R.H., Rohit, R.: Towards tight differential bounds of ascon: A hybrid usage of smt and milp. IACR Transactions on Symmetric Cryptology pp. 303–340 (2022)
14. Mouha, N., Preneel, B.: Towards finding optimal differential characteristics for arx: Application to salsa20. Cryptology ePrint Archive (2013)
15. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Information Security and Cryptology: 7th International Conference, Inscrypt 2011, Beijing, China, November 30–December 3, 2011. Revised Selected Papers 7. pp. 57–76. Springer (2012)
16. Qiao, K., Song, L., Liu, M., Guo, J.: New collision attacks on round-reduced keccak. In: Advances in Cryptology–EUROCRYPT 2017: 36th Annual International

Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30–May 4, 2017, Proceedings, Part III 36. pp. 216–243. Springer (2017)

17. Yu, X., Liu, F., Wang, G., Sun, S., Meier, W.: A closer look at the s-box: Deeper analysis of round-reduced ascon-hash. Cryptology ePrint Archive (2023)
18. Zong, R., Dong, X., Wang, X.: Collision attacks on round-reduced gimli-hash/ascon-xof/ascon-hash. Cryptology ePrint Archive (2019)

A Equation systems of preimage attack on 2-round Ascon-XOF

$$
\begin{aligned}
&Y_0^2[0] = X_0^1[61] \oplus X_0^1[39] \oplus X_0^1[10] \oplus X_0^1[17]\\
&Y_0^2[1] = X_0^1[62] \oplus X_0^1[40] \oplus X_0^1[11] \oplus X_0^1[18]\\
&Y_0^2[2] = X_0^1[63] \oplus X_0^1[41] \oplus X_0^1[12] \oplus X_0^1[19]\\
&Y_0^2[3] = X_0^1[0] \oplus X_0^1[42] \oplus X_0^1[13] \oplus X_0^1[20]\\
&Y_0^2[4] = X_0^1[1] \oplus X_0^1[43] \oplus X_0^1[14] \oplus X_0^1[21]\\
&Y_0^2[5] = X_0^1[2] \oplus X_0^1[44] \oplus X_0^1[15] \oplus X_0^1[22]\\
&Y_0^2[6] = X_0^1[3] \oplus X_0^1[45] \oplus X_0^1[16] \oplus X_0^1[23]\\
&Y_0^2[7] = X_0^1[4] \oplus X_0^1[46] \oplus X_0^1[17] \oplus X_0^1[24]\\
&Y_0^2[8] = X_0^1[5] \oplus X_0^1[47] \oplus X_0^1[18] \oplus X_0^1[25]\\
&Y_0^2[9] = X_0^1[6] \oplus X_0^1[48] \oplus X_0^1[19] \oplus X_0^1[26]\\
&Y_0^2[10] = X_0^1[7] \oplus X_0^1[49] \oplus X_0^1[20] \oplus X_0^1[27]\\
&Y_0^2[11] = X_0^1[8] \oplus X_0^1[50] \oplus X_0^1[21] \oplus X_0^1[28]\\
&Y_0^2[12] = X_0^1[9] \oplus X_0^1[51] \oplus X_0^1[22] \oplus X_0^1[29]\\
&Y_0^2[13] = X_0^1[10] \oplus X_0^1[52] \oplus X_0^1[23] \oplus X_0^1[30]\\
&Y_0^2[14] = X_0^1[11] \oplus X_0^1[53] \oplus X_0^1[24] \oplus X_0^1[31]\\
&Y_0^2[15] = X_0^1[12] \oplus X_0^1[54] \oplus X_0^1[25] \oplus X_0^1[32]\\
&Y_0^2[16] = X_0^1[13] \oplus X_0^1[55] \oplus X_0^1[26] \oplus X_0^1[33]\\
&Y_0^2[17] = X_0^1[14] \oplus X_0^1[56] \oplus X_0^1[27] \oplus X_0^1[34]\\
&Y_0^2[18] = X_0^1[15] \oplus X_0^1[57] \oplus X_0^1[28] \oplus X_0^1[35]\\
&Y_0^2[19] = X_0^1[16] \oplus X_0^1[58] \oplus X_0^1[29] \oplus X_0^1[36]\\
&Y_0^2[20] = X_0^1[17] \oplus X_0^1[59] \oplus X_0^1[30] \oplus X_0^1[37]\\
&Y_0^2[21] = X_0^1[18] \oplus X_0^1[60] \oplus X_0^1[31] \oplus X_0^1[38]\\
&Y_0^2[22] = X_0^1[19] \oplus X_0^1[61] \oplus X_0^1[32] \oplus X_0^1[39]\\
&Y_0^2[23] = X_0^1[20] \oplus X_0^1[62] \oplus X_0^1[33] \oplus X_0^1[40]\\
&Y_0^2[24] = X_0^1[21] \oplus X_0^1[63] \oplus X_0^1[34] \oplus X_0^1[41]\\
&Y_0^2[25] = X_0^1[22] \oplus X_0^1[0] \oplus X_0^1[35] \oplus X_0^1[42]\\
&Y_0^2[26] = X_0^1[23] \oplus X_0^1[1] \oplus X_0^1[36] \oplus X_0^1[43]\\
&Y_0^2[27] = X_0^1[24] \oplus X_0^1[2] \oplus X_0^1[37] \oplus X_0^1[44]\\
&Y_0^2[28] = X_0^1[25] \oplus X_0^1[3] \oplus X_0^1[38] \oplus X_0^1[45]\\
&Y_0^2[29] = X_0^1[26] \oplus X_0^1[4] \oplus X_0^1[39] \oplus X_0^1[46]\\
&Y_0^2[30] = X_0^1[27] \oplus X_0^1[5] \oplus X_0^1[40] \oplus X_0^1[47]\\
&Y_0^2[31] = X_0^1[28] \oplus X_0^1[6] \oplus X_0^1[41] \oplus X_0^1[48]\\
&X_0^2[0] = X_0^1[0] \oplus X_0^1[19] \oplus X_0^1[28]\\
&X_0^2[1] = X_0^1[1] \oplus X_0^1[20] \oplus X_0^1[29]\\
&X_0^2[2] = X_0^1[2] \oplus X_0^1[21] \oplus X_0^1[30]\\
&X_0^2[3] = X_0^1[3] \oplus X_0^1[22] \oplus X_0^1[31]\\
&X_0^2[4] = X_0^1[4] \oplus X_0^1[23] \oplus X_0^1[32]\\
&X_0^2[5] = X_0^1[5] \oplus X_0^1[24] \oplus X_0^1[33]\\
&X_0^2[6] = X_0^1[6] \oplus X_0^1[25] \oplus X_0^1[34]\\
&X_0^2[7] = X_0^1[7] \oplus X_0^1[26] \oplus X_0^1[35]\\
&X_0^2[8] = X_0^1[8] \oplus X_0^1[27] \oplus X_0^1[36]\\
&X_0^2[9] = X_0^1[9] \oplus X_0^1[28] \oplus X_0^1[37]\\
&X_0^2[10] = X_0^1[10] \oplus X_0^1[29] \oplus X_0^1[38]
\end{aligned}
$$

$$
\begin{aligned}
X_0^2[11] &= X_0^1[11] \oplus X_0^1[30] \oplus X_0^1[39]\\
X_0^2[12] &= X_0^1[12] \oplus X_0^1[31] \oplus X_0^1[40]\\
X_0^2[13] &= X_0^1[13] \oplus X_0^1[32] \oplus X_0^1[41]\\
X_0^2[14] &= X_0^1[14] \oplus X_0^1[33] \oplus X_0^1[42]\\
X_0^2[15] &= X_0^1[15] \oplus X_0^1[34] \oplus X_0^1[43]\\
X_0^2[16] &= X_0^1[16] \oplus X_0^1[35] \oplus X_0^1[44]\\
X_0^2[17] &= X_0^1[17] \oplus X_0^1[36] \oplus X_0^1[45]\\
X_0^2[18] &= X_0^1[18] \oplus X_0^1[37] \oplus X_0^1[46]\\
X_0^2[19] &= X_0^1[19] \oplus X_0^1[38] \oplus X_0^1[47]\\
X_0^2[20] &= X_0^1[20] \oplus X_0^1[39] \oplus X_0^1[48]\\
X_0^2[21] &= X_0^1[21] \oplus X_0^1[40] \oplus X_0^1[49]\\
X_0^2[22] &= X_0^1[22] \oplus X_0^1[41] \oplus X_0^1[50]\\
X_0^2[23] &= X_0^1[23] \oplus X_0^1[42] \oplus X_0^1[51]\\
X_0^2[24] &= X_0^1[24] \oplus X_0^1[43] \oplus X_0^1[52]\\
X_0^2[25] &= X_0^1[25] \oplus X_0^1[44] \oplus X_0^1[53]\\
X_0^2[26] &= X_0^1[26] \oplus X_0^1[45] \oplus X_0^1[54]\\
X_0^2[27] &= X_0^1[27] \oplus X_0^1[46] \oplus X_0^1[55]\\
X_0^2[28] &= X_0^1[28] \oplus X_0^1[47] \oplus X_0^1[56]\\
X_0^2[29] &= X_0^1[29] \oplus X_0^1[48] \oplus X_0^1[57]\\
X_0^2[30] &= X_0^1[30] \oplus X_0^1[49] \oplus X_0^1[58]\\
X_0^2[31] &= X_0^1[31] \oplus X_0^1[50] \oplus X_0^1[59]
\end{aligned}
$$

B Linear equations of Y_0^3 when IV is set to 0.

When guessing bits of X_0^1 are all guessed as 0, the expression for the linearized bits of Y_0^3 are as follows:

$$\begin{aligned}
&Y_0^3[63] = X_0^1[45] \oplus X_0^1[23] \oplus X_0^1[3]\\
&Y_0^3[52] = X_0^1[39] \oplus X_0^1[17]\\
&Y_0^3[37] = 0\\
&Y_0^3[34] = 0\\
&Y_0^3[30] = X_0^1[17] \oplus X_0^1[0]\\
&Y_0^3[27] = 0\\
&Y_0^3[24] = 0\\
&Y_0^3[21] = X_0^1[45] \oplus X_0^1[3]\\
&Y_0^3[12] = X_0^1[36] \oplus X_0^1[32]\\
&Y_0^3[5] = X_0^1[39]\\
&Y_0^3[2] = X_0^1[36]
\end{aligned}$$

When guessing bits of X_0^1 are all guessed as 1, the expression for the linearized bits of Y_0^3 are as follows:

$$\begin{aligned}
&Y_0^3[63] = X_0^1[23]\\
&Y_0^3[52] = X_0^1[39] \oplus X_0^1[23] \oplus 1\\
&Y_0^3[37] = X_0^1[42]\\
&Y_0^3[34] = X_0^1[39]\\
&Y_0^3[30] = X_0^1[17]\\
&Y_0^3[27] = X_0^1[32] \oplus X_0^1[23]\\
&Y_0^3[24] = X_0^1[20]\\
&Y_0^3[21] = X_0^1[45] \oplus X_0^1[17] \oplus 1\\
&Y_0^3[12] = X_0^1[36] \oplus X_0^1[17] \oplus 1\\
&Y_0^3[5] = X_0^1[10]\\
&Y_0^3[2] = 1
\end{aligned}$$

C Linear equations of Y_0^3 under Real IV.

When guessing bits of X_0^1 are all guessed as 0, the expression for the linearized bits of Y_0^3 are as follows:

$Y_0^3[58] = X_0^1[50] \oplus 1$
$Y_0^3[57] = 1$
$Y_0^3[52] = X_0^1[53] \oplus X_0^1[50] \oplus X_0^1[28] \oplus 1$
$Y_0^3[49] = X_0^1[50] \oplus X_0^1[25]$
$Y_0^3[47] = 0$
$Y_0^3[44] = X_0^1[36] \oplus 1$
$Y_0^3[35] = 1$
$Y_0^3[30] = X_0^1[36] \oplus X_0^1[28] \oplus X_0^1[15] \oplus X_0^1[6]$
$Y_0^3[27] = 0$
$Y_0^3[19] = X_0^1[25] \oplus X_0^1[17]$
$Y_0^3[16] = X_0^1[45]$
$Y_0^3[13] = X_0^1[62] \oplus X_0^1[53] \oplus X_0^1[14]$
$Y_0^3[5] = X_0^1[50] \oplus X_0^1[45] \oplus X_0^1[15] \oplus X_0^1[6] \oplus X_0^1[3] \oplus 1$

When guessing bits of X_0^1 are all guessed as 1, the expression for the linearized bits of Y_0^3 are as follows:

$Y_0^3[58] = X_0^1[50]$
$Y_0^3[57] = X_0^1[62] \oplus X_0^1[53] \oplus X_0^1[17] \oplus X_0^1[3]$
$Y_0^3[52] = X_0^1[62] \oplus X_0^1[14] \oplus X_0^1[6]$
$Y_0^3[49] = X_0^1[25] \oplus X_0^1[14]$
$Y_0^3[47] = 0$
$Y_0^3[44] = X_0^1[45] \oplus X_0^1[25]$
$Y_0^3[35] = X_0^1[35] \oplus X_0^1[40] \oplus X_0^1[17]$
$Y_0^3[30] = X_0^1[40] \oplus X_0^1[17] \oplus X_0^1[15] \oplus 1$
$Y_0^3[27] = X_0^1[25] \oplus X_0^1[3]$
$Y_0^3[19] = X_0^1[45] \oplus X_0^1[36] \oplus 1$
$Y_0^3[16] = X_0^1[45] \oplus X_0^1[17] \oplus X_0^1[14]$
$Y_0^3[13] = X_0^1[62] \oplus X_0^1[17] \oplus 1$
$Y_0^3[5] = X_0^1[50] \oplus X_0^1[45] \oplus X_0^1[3] \oplus 1$

D Initial state S_1^0 of 3-round Ascon-XOF under real IV.

Table 8. Initial state S_1^0 of 3-round Ascon-XOF under real IV.

Item	S_1^0 (Hex)
Initial state	572189c148318b05 af4d04273a5422d6 09a0c623eb455377 55d5f514195c3489 9d6294e4afc8e4d7

E 4-round differential trail of Ascon-HASH.

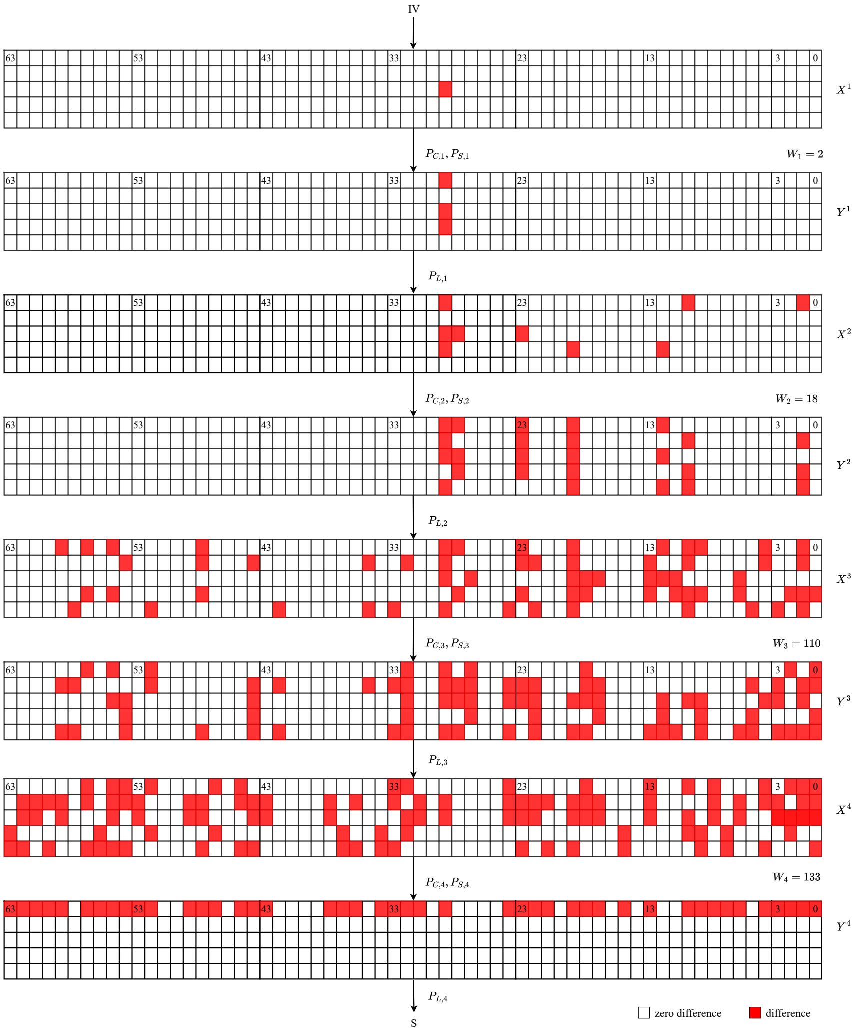

Fig. 13. 4-round differential trail of Ascon-HASH

F 5-round differential trail of Ascon-HASH.

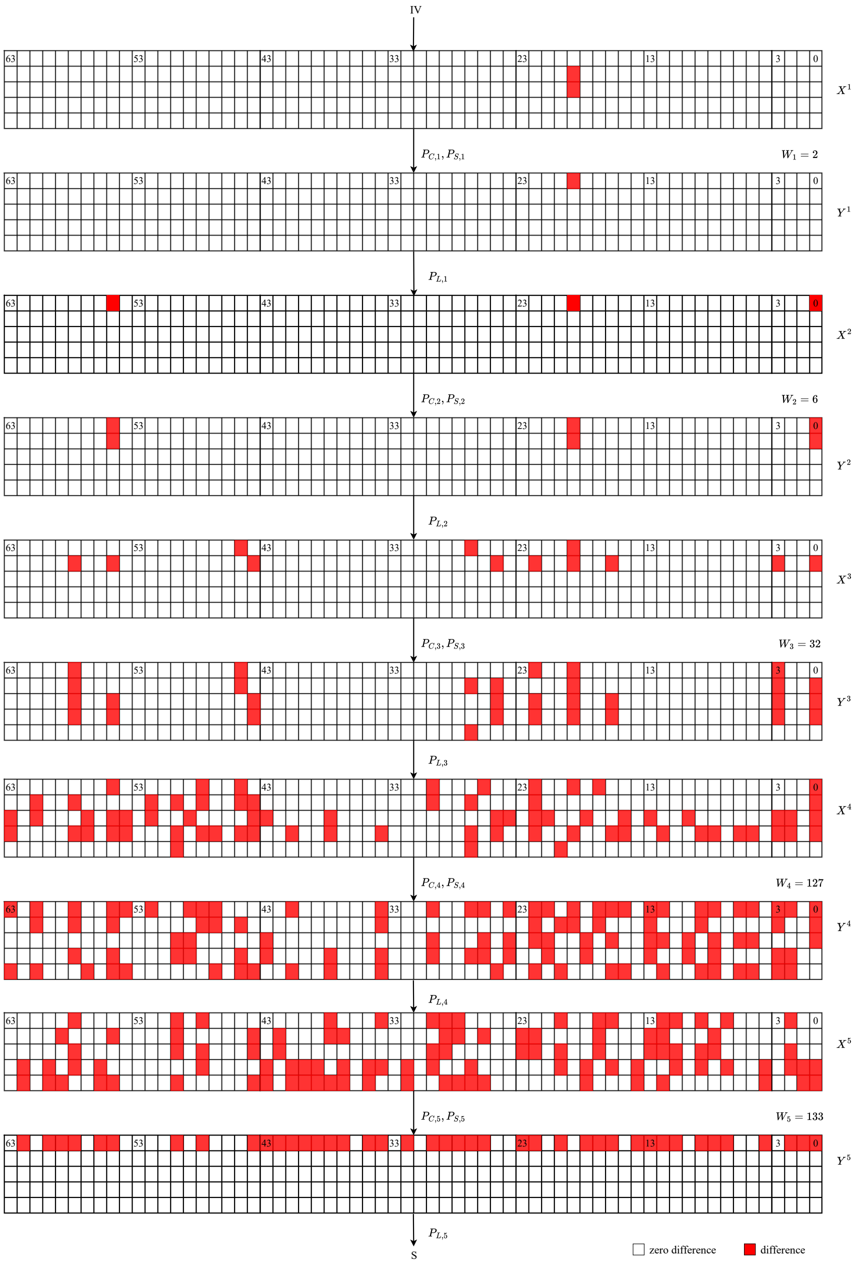

Fig. 14. 5-round differential trail of Ascon-HASH