Preimage Attacks on up to 5 Rounds of SHA-3 Using Internal Differentials *

Zhongyi Zhang[1,2][0009−0008−0491−3006], Chengan Hou[1,2][0009−0009−5618−6979], and Meicheng Liu[1,2(✉)][0000−0002−5259−1848]

[1] State Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, People's Republic of China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, People's Republic of China
{zhangzhongyi0714,houchengan,liumeicheng}@iie.ac.cn

Abstract. In this paper, we study preimage resistance of the SHA-3 standard. We propose a squeeze meet-in-the-middle attack as a new preimage attack method for the sponge functions. This attack combines the squeeze attack and meet-in-the-middle attack, and is implemented by internal differentials. We analyze the inverse operation of the SHA-3 round function, and develop a new target internal differential algorithm as well as a linearization technique for the Sbox in the backward phase. In addition, we propose the concept of a value-difference distribution table (VDDT) to optimize the attack complexity. These techniques lead to faster preimage attacks on five (out of six) SHA-3 functions reduced to 4 rounds, and also bring preimage attacks on 5 rounds of four SHA-3 instances. The attack techniques are verified by performing practical preimage attack on a small variant of 4-round Keccak.

Keywords: Hash Function · SHA-3 · Preimage Attack · Internal Differentials · Linearization · Meet-in-the-Middle

1 Introduction

The Keccak hash function, designed by Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche [2], emerged victorious in the SHA-3 (Secure Hash Algorithm-3) competition conducted by the National Institute of Standards and Technology (NIST) in the United States. In 2015, NIST released the final version of the SHA-3 standard [9]. The SHA-3 family consists of four cryptographic hash functions, called SHA3-224, SHA3-256, SHA3-384 and SHA3-512, and two extendable-output functions (XOFs), called SHAKE128 and SHAKE256, capable of generating digests of variable lengths.

* Supported by the National Key R&D Program of China (No. 2024YFA1013000), the National Natural Science Foundation of China (Grant No. 62122085 and 12231015), the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDB0690000 and the Youth Innovation Promotion Association of Chinese Academy of Sciences.

The SHA-3 function, to say, KECCAK, employs a sponge construction that accommodates messages of varying lengths for hash function inputs. The message undergoes padding and is then divided into uniform-sized message blocks. The 1600-bit initial state of KECCAK is XORed with the first message block, followed by 24 rounds of the KECCAK-f permutation applied to update the state and XORing of subsequent message blocks until absorption is complete. Finally, a final 24-round KECCAK-f is applied to the state, and selected state bits are extracted as the resulting digest. Since its introduction in 2008, KECCAK has emerged as a crucial hash function, undergoing extensive security analysis, including evaluations of preimage resistance and collision resistance [1,3,6,7,12,24,28,15,10,11,29,25,30].

In this paper, we focus on examining the security of the SHA-3 family against preimage attacks. A preimage attack is to find a message that has a specific hash digest. In the literature, the linear structure [12,16,15,26,17,14,18] is one of the major cryptanalytic tools for security evaluation of SHA-3 against preimage resistance. In 2016, Guo, Liu and Song [12] introduced the linear structure technique to linearize several rounds of KECCAK and derived preimage attacks on up to 4 rounds of KECCAK. Building on Guo *et al.*'s framework, Li and Sun [16,15] improved preimage attacks by the cross-linear structure and allocating approach. Further improvements in this line were proposed in [26,17,14,18].

In 2021, Dinur [5] devised a polynomial method-based algorithm for solving multivariate equation systems and obtained preimage and collision attacks on 4-round variants of KECCAK, including the existing best known preimage attacks on 4-round KECCAK-384 and KECCAK-512 with complexities of 2^{374} and 2^{502} bit operations respectively.

In 2011, an MITM attack on 2-round KECCAK was given by Naya-Plasencia *et al.* [22]. They computed the inverse of one round KECCAK from the target and obtained partial internal states. Then, after dividing the message block into many independent parts, the authors computed forward independently for each part until the known internal states and filter the messages. In 2023, Qin *et al.* [25] introduced a framework of MITM preimage attacks on sponge-based hashing, where two independent forward chunks are applied (without backward chunk). Starting from the r-bit outer part determined by the last message, the two neutral sets compute independently forward to the m-bit matching point, which is an m-bit deterministic relation on the two neutral sets by partially solving the inverse of the permutation from the n-bit target. With the help of linear structure technique, Qin *et al.* [25] presented an MITM preimage attack on 4-round KECCAK-512.

In addition, Bernstein [1] and Chang *et al.* [3] used algebraic techniques to speed up a brute-force (second) preimage search for up to 9 rounds of KECCAK. However, the best known preimage attacks on the SHA-3 functions with more than a tiny advantage over a brute-force search currently only reach 4 rounds.

Our Contribution. Different from the attacks mentioned above, we consider the first application of internal differentials to preimage attacks on SHA-3. Internal differential was initially developed by Peyrin [23] in the cryptanalysis of

Grøstl hash function. In 2013, Dinur, Dunkelman and Shamir [6] described a squeeze attack and conducted collision attacks on up to 5 rounds of SHA-3 by utilizing generalized internal differentials. In the work of Dinur *et al.*, a useful algorithm, called target internal difference algorithm (TIDA), was developed for constructing 1-round connector of internal differentials. Recently, Zhang, Hou and Liu [29,30] proposed the conditional internal differentials and improved the target internal difference algorithm by probabilistic linearization. With the help of these methods, Zhang *et al.* presented collision attacks for the SHA-3 functions reduced to up to 6 rounds. Inspired by the work of Dinur *et al.* [6] and Zhang *et al.* [29,30] on internal differential collision attacks, we propose a new framework for preimage attacks, which combines the squeeze attack with MITM attack and makes use of internal differentials. It brings preimage attacks on the SHA-3 variants up to 5 rounds. The main contributions are summarized as follows.

Squeeze meet-in-the-middle attack. Launching a meet-in-the-middle attack on sponge function essentially involves searching for a collision at the capacity part in both forward and backward directions. Squeeze attack is an effective method for collision attack. We propose the squeeze meet-in-the-middle attack by combining these two approaches, and utilize internal differentials to facilitate the squeezing process. The attack consists of three phases: forward phase, backward phase, and collision phase. Each of the first two phases consists of a TIDA stage and collecting messages stage.

Inverse internal differentials. We introduce inverse differential transition conditions on the Keccak Sbox and use it to constrain the initial message space. Combined with the linearization technique for the inverse of Sbox developed in this paper, the complexity of the backward phase in the attack is reduced.

Backward target internal difference algorithm. In the backward phase of the attack, we link the internal differential characteristics starting from the penultimate round to the initial message space by randomly selecting the last block and modifying the value of the second-to-last block. In Backward-TIDA, we also introduce a value-difference distribution table (VDDT) to speed up the construction of the connector.

The results of our attacks on SHA-3 are summarized in Table 1 with a comparison of the related previous work. We present the first 5-round preimage attacks that are much faster than a brute-force search[3] for four SHA-3 functions, SHAKE128, SHA3-224, SHA3-256 and SHAKE256, with complexities of $2^{100.5}$ Keccak computations, $2^{216.03}$ bit operations, $2^{254.33}$ bit operations and $2^{254.33}$ bit operations, respectively. We also obtain the best known 4-round preimage attacks for five SHA-3 functions, SHAKE128, SHA3-224, SHA3-256, SHAKE256 and SHA3-384, with complexities of $2^{81.5}$, $2^{135.5}$, $2^{151.5}$, $2^{151.5}$ and $2^{277.8}$ Keccak computations, respectively. Besides, our 4-round and 5-round preimage attacks on the two XOFs are valid for SHAKE128$(M, 256)$ and SHAKE256$(M, 512)$, which support the strongest collision resistance and are adopted in the pre-hash signa-

[3] An evaluation of the Keccak hash function is assumed to require thousands of bit operations (see also [3,5]), so the complexity of a brute-force search over 2^d messages is significantly larger than 2^d bit operations.

Target	Rounds	Complexity	Attack method	Reference
SHA3-224	4	2^{221}	Rotational cryptanalysis	[19]
	4	2^{217}†	Solving polynomial systems	[5]
	4	2^{213}	Linear structure	[13]
	4	2^{207}	Linear structure	[15]
	4	2^{192}	Linear structure	[14]
	4	$\mathbf{2^{135.5}}$	Internal differential	Section 6.5
	5	$\mathbf{2^{216.03}}$†	Internal differential	Section 6.2
SHA3-256	4	2^{252}	Rotational cryptanalysis	[19]
	4	2^{251}	Linear structure	[13]
	4	2^{246}†	Solving polynomial systems	[5]
	4	2^{239}	Linear structure	[15]
	4	2^{218}	Linear structure	[14]
	4	$\mathbf{2^{151.5}}$	Internal differential	Section 6.5
	5	$\mathbf{2^{254.33}}$†	Internal differential	Section 6.3
SHA3-384	3	2^{322}	Linear structure	[13]
	4	2^{378}	Rotational cryptanalysis	[19]
	4	2^{374}†	Solving polynomial systems	[5]
	4	$\mathbf{2^{277.8}}$	Internal differential	Section 6.4
SHA3-512	3	$2^{504.2}$	Internal differential	Section B
	3	2^{482}	Linear structure	[13]
	4	2^{506}	Rotational cryptanalysis	[19]
	4	$2^{504.58}$	MITM	[25]
	4	2^{502}†	Solving polynomial systems	[5]
SHAKE128	4	2^{106}#	Linear structure	[13]
	4	$\mathbf{2^{81.5}}$	Internal differential	Section 6.5
	5	$\mathbf{2^{100.5}}$	Internal differential	Section 6.1
SHAKE256	4	2^{251}#	Linear structure	[13]
	4	2^{239}#	Linear structure	[15]
	4	$\mathbf{2^{151.5}}$	Internal differential	Section 6.5
	5	$\mathbf{2^{254.33}}$†	Internal differential	Section 6.3

† The complexity is calculated by bit operations.

The attack target is SHAKE-X(M, d) with the digest length $d =$ X, and the complexity will increase by a factor of $2^{d-\mathtt{X}}$ for $d >$ X.

Table 1: Comparison of preimage attacks on round-reduced SHA-3

tures of the post-quantum digital signature standards ML-DSA [20] and SLH-DSA [21]. To the best of our knowledge, this is the first time that a preimage attack is shown for these SHAKE instances.

The attack techniques are demonstrated by implementing a practical preimage attack on a small variant of Keccak, *i.e.*, Keccak$[r = 704, c = 96, n_r = 4]$.

Organization. The rest of the paper is organized as follows. In Section 2, we describe the SHA-3 hash function. In Section 3, we propose the squeeze meet-in-the-middle attack. In Section 4, we list the notations used in this paper and

review the internal differentials. Section 5 presents the framework of our attacks, followed by detailed explanations over our techniques. The results of our attacks are given in Section 6. We conduct experiments for verifying our attacks in Section 7, and conclude the paper in Section 8.

2 Description of SHA-3

In this section, we give a brief description of the sponge construction and the `SHA-3` hash function, *i.e.*, the KECCAK hash function. Subsequently, the security strengths of `SHA-3` instances are given.

2.1 The Sponge Function

The sponge construction is a framework for constructing hash functions based on permutations. The sponge construction proceeds in two phases: absorbing phase and squeezing phase, as shown in Figure 1. The message is firstly padded by appending a bit string of 10*1, where 0* represents a shortest string of 0's so that the length of padded message is multiple of r, and cut into r-bit blocks. The b-bit internal state is initialized to be all zeros. In absorbing phase, each message block is XORed into the first r bits of the current state, and then applying a fixed permutation to the entire b-bit state. The sponge construction switches to the squeezing phase after all message blocks are processed. In this phase, the first r bits of the state are returned as output and the permutation is applied in each iteration. This process is repeated until all d bits digest are produced.

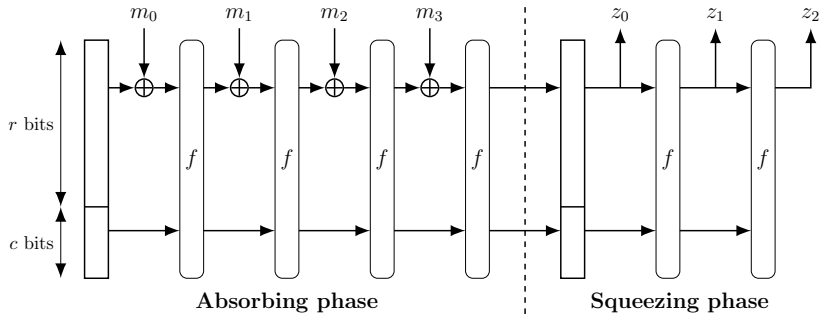


Fig. 1: The sponge construction

2.2 The Keccak Hash Function

The KECCAK permutation has 24 rounds, which operates on the 1600-bit state that can be viewed as a 3-dimensional array of bits. One bit of the state at

position (x, y, z) is noted as $A[x][y][z]$, where $0 \leq x, y < 5$ and $0 \leq z < 64$. The designers of KECCAK defined the following naming conventions: $A[\cdot][y][z]$ is a row, $A[x][\cdot][z]$ is a column, and $A[x][y][\cdot]$ is a lane; $A[x][\cdot][\cdot]$ is a sheet, $A[\cdot][y][\cdot]$ is a plane, and $A[\cdot][\cdot][z]$ is a slice.

There are five mappings in each round of the permutation:

$$\theta : A[x][y][z] \leftarrow A[x][y][z] + \sum_{y'=0}^{4} A[x-1][y'][z] + \sum_{y'=0}^{4} A[x+1][y'][z-1].$$

$$\rho : A[x][y][z] \leftarrow A[x][y][z + T(x,y)], \text{where } T(x,y) \text{ is a predefined constant.}$$

$$\pi : A[x][y][z] \leftarrow A[x'][y'][z], \text{where } \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} x' \\ y' \end{pmatrix}.$$

$$\chi : A[x][y][z] \leftarrow A[x][y][z] + (\neg(A[x+1][y][z])) \wedge A[x+2][y][z].$$

$$\iota : A \leftarrow A + RC[i_r], \text{where } RC[i_r] \text{ is the round constants.}$$

The addition and multiplication are in $GF(2)$. Since we analyse round-reduced variant with at most 5 rounds, we only give the first four round constants: 0000000000000001, 0000000000008082, 800000000000808a, 8000000080008000 (given in hexadecimal using the little-endian format).

2.3 Instances of SHA-3

The four instances of `SHA-3` family named `SHA3-d` are defined from KECCAK[c] by appending a two-bit suffix '01' to the message, where $b = 1600$, $c = 2d$ and $d \in \{224, 256, 384, 512\}$. After that, the padding of KECCAK is applied. `SHAKE128` and `SHAKE256` are two XOF instances with the capacity $c = 256$ and 512 respectively, and the original message M is appended with an additional 4-bit suffix '1111' before applying the padding rule. The suffixes "128" and "256" indicate the security strengths that these two functions can generally support. We summarize specifications and security strengths of the `SHA-3` functions in Table 2.

Function	**Rate Size**	**Capacity Size**	**Output Size**	**Security Strengths in Bits**		
				Collision	**Preimage**	**2nd Preimage**
`SHA3-224`	1152	448	224	112	224	224
`SHA3-256`	1088	512	256	128	256	256
`SHA3-384`	832	768	384	192	384	384
`SHA3-512`	576	1024	512	256	512	512
`SHAKE128`	1344	256	d	$\min(d/2, 128)$	$\geq \min(d, 128)$	$\min(d, 128)$
`SHAKE256`	1088	512	d	$\min(d/2, 256)$	$\geq \min(d, 256)$	$\min(d, 256)$

Table 2: Specifications and security strengths of `SHA-3` functions

3 Squeeze Meet-in-the-Middle Attack

In this section, we first recall Dinur *et al.*'s squeeze attack, then describe a generalized framework, called squeeze meet-in-the-middle attack.

3.1 Squeeze Attack

Dinur *et al.* [6] employed a method for collision attack on KECCAK by focusing on a specific subset of outputs to find hash function collisions. It describes how, by limiting outputs to a smaller subset and considering the probabilities of inputs leading to these outputs, one can potentially find collisions more efficiently than traditional methods. This strategy is called a *squeeze attack*.

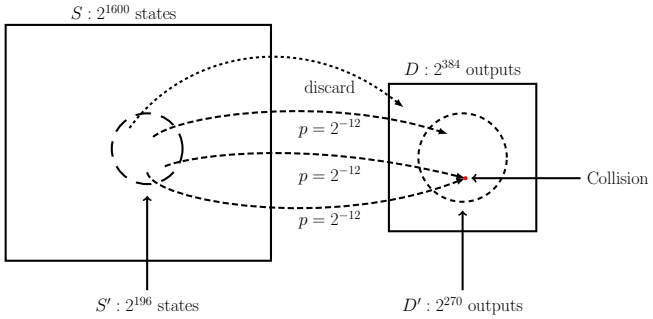


Fig. 2: A Squeeze Attack on 4-round `SHA3-384`

To illustrate this attack, assume that the hash function F maps a set S of possible inputs into a set D of possible outputs. By the birthday paradox, we have to try a subset $S' \subseteq S$ of size $|D|^{1/2}$. We consider a subset D' of D, where $|D'| = q|D|$, and the probability of picking an input in S' whose output is in D' is p. To find a collision in D', the number of inputs in S' we have to try is $(q^{1/2}/p)|D|^{1/2}$. When the mapping is random, $p = q$, making this attack worse than the birthday bound for all D' which are smaller than D. If we can exploit some non-random behavior of the hash function in order to find sets S' and D' for which $p^2 > q$, we can get an improved collision finding algorithm. This is called a squeeze attack, as we are forcing a larger than expected number of inputs to squeeze into a smaller subset of possible outputs where collisions are more likely. Figure 2 shows the collision attack on 4-round `SHA3-384` in [6].

3.2 Squeeze Meet-in-the-Middle Attack

In the sponge hash function, the meet-in-the-middle attack can be used to recover the original preimage of any digest H. For example, in the attack on `SHA-3`, given a digest H of length d-bit ($c/2 \leq d \leq 1600$), first fill it with all '0' bits

or random bits to a 1600-bit state $\overline{H}$. Then we use three blocks of message $M_0||M_1||M_2$ to perform the meet-in-the-middle attack. Set the value set of the capacity part to D. Randomly select multiple M_0 and M_2 and calculate $f(M_0)$ and $f^{-1}(f^{-1}(\overline{H}) \oplus M_2)$. The projections of $f(M_0)$ and $f^{-1}(f^{-1}(\overline{H}) \oplus M_2)$ in the capacity part are stored in sets D_L and D_S respectively, where D_L and D_R are subsets of D. We employ hash table technique to exhaustively search the elements in D_L and D_S until we find two values $f(M_0)$ and $f^{-1}(f^{-1}(\overline{H}) \oplus M_2)$ with the same value in capacity part. Finally, modify the value of M_1 to make $f(M_0) \oplus M_1 = f^{-1}(f^{-1}(\overline{H}) \oplus M_2)$, so $M_0||M_1||M_2$ is a preimage of H. According to Sasaki's analysis in [27], if the sizes of D_L and D_R satisfy $|D_L| \times |D_R| = |D| = 2^c$, the attacker can perform the above meet-in-the-middle attack with a success probability of 0.63, which is also the success probability of the preimage attack by exhaustive search. When $|D_L| = |D_R|$, the attack takes the minimum complexity of $2^{c/2+1}$. This attack does not undermine the security strength of $c/2$.

In this paper, we combine the squeeze attack with the meet-in-the-middle attack and propose a new preimage attack method. Consider a subset D' of D, where $|D'| = q|D|$, along with the input subset S'_L of f and input subset S'_R of f^{-1}. The outputs of S'_L and S'_R on the capacity part are filtered and stored in the subsets D'_L and D'_R of D', respectively, with probabilities $p_L = 2^{-K_1}$ and $p_R = 2^{-K_2}$. Set $|D'_L| = 2^{n_1}, |D'_R| = 2^{n_2}$, if $2^{n_1+n_2} = |D'| = q \cdot 2^c$, the adversary can also launch a meet-in-the-middle attack with a success probability of 0.63. The total complexity of the attack is $2^{K_1+n_1} + 2^{K_2+n_2}$, and the minimum value $q^{1/2} \cdot 2^{(K_1+K_2+c)/2+1}$ is taken when $K_1 + n_1 = K_2 + n_2$. If we can exploit some non-random behavior of the hash function in order to find set S'_L, S'_R and D'_L, D'_R for which $K_1 + K_2 + 2 < \log_2(1/q)$, we can get an improved preimage recovering algorithm. This is called a *squeeze meet-in-the-middle* attack.

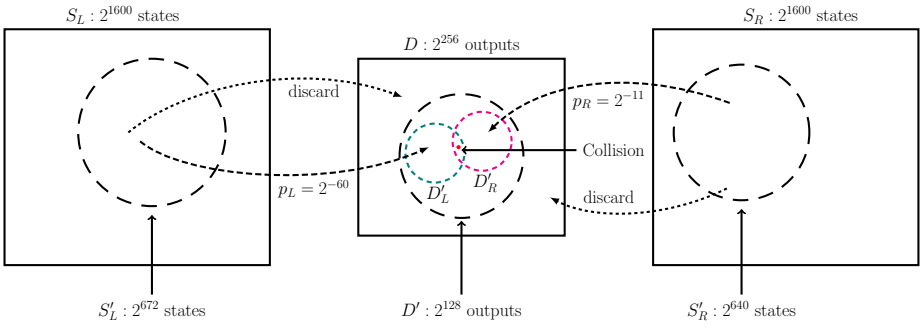


Fig. 3: A Squeeze Meet-in-the-Middle Attack on 5-round `SHAKE128`

In this paper, we utilize internal differentials to launch squeeze meet-in-the-middle attack. The size of D' is $2^{c/2}$ in our attack (using internal differentials of period 32), so the number of messages collected from the forward and backward directions needs to satisfy $2^{n_1+n_2} = 2^{c/2}$. Figure 3 shows our squeeze meet-in-the-middle attack model for 5-round `SHAKE128`. The detailed description of the attack framework will be shown in Section 5.

4 Notations and Review of Internal Differentials

In this section, we review the concepts of internal differentials, accompanied by a list of notations used in this paper.

4.1 Notations

The addition operation of the state is performed on $GF(2)$ or the linear space over $GF(2)$. We summarize the major notations to be used in this paper here.

n_r	Number of attacked rounds
c	Capacity of a sponge function
r	Rate of a sponge function
b	Width of a Keccak permutation in bits, $b = r + c$
d	Length of the digest in bits
p	Number of minimum fixed bits in the initial state due to padding
i	Period of a symmetric state
$\theta, \rho, \pi, \chi, \iota$	The five mappings that comprise a round.
L	Composition of θ, ρ, π and its inverse denoted by L^{-1}
$\mathtt{R}^j(\cdot)$	Keccak permutation reduced to the first j rounds
$\mathtt{S}(\cdot)$	5-bit Sbox operating on each row of Keccak state
$\delta_{in}, \delta_{out}$	5-bit input and output differences of an Sbox
$\overline{M}$	Padded message of M. Note that $\overline{M}$ is the fifth block in our attack
$M_0 \| \| M_1$	Concatenation of strings M_0 and M_1
α_{j-1}	Input internal difference of the j-th round function with period 32
β_{j-1}	Input internal difference of χ in the j-th round with period 32
$\beta \to \alpha^\star$	β is the internal difference input to χ and $\alpha^\star$ is the output difference
$\Delta(\cdot)$	Internal difference of one state
$\mathtt{E}[\cdot]$	Expectation of one random variable
k_j	Transition condition number of j-th round in forward phase
k'_j	Transition condition number of j-th round in backward phase

4.2 Internal Differentials

The internal differential collision attack is essentially a squeeze attack. Internal differential was initially developed by Peyrin [23] in the cryptanalysis of the Grøstl hash function. This method was later generalized by Dinur *et al.* [6] in collision attacks on Keccak. Similar to the case of standard differential analysis, the adversary's goal is to find several internal differential characteristics with high transition probability by tracking the differences between different parts of the internal state through the cryptographic function. The difference between standard differential and internal differential is that the input of the former is multiple message pairs, while the input of latter is multiple messages. Another difference is that in standard differential analysis, the adversary can check whether a collision occurred after each input of a message pair. In internal

differential analysis, the adversary has to input enough messages to enter the characteristic, and search for each collision subset at the end of the characteristic until a collision is found. It can be seen that internal differential analysis is a squeeze attack, the collision subset is a subset of the output set D, and the transition probability of the internal differential characteristic is the probability p that the input enter the output subset.

In KECCAK, a state is called a *symmetric state* if it has period i in the z-axis. This means that for all (x, y, z), there is some positive integer i less than 64 such that state A satisfies $A[x][y][z] = A[x][y][(z+i) \bmod 64]$. An interesting property of KECCAK is that after applying any of the θ, ρ, π, χ operations to a symmetric state, the new state is still a symmetric state and maintains its period.

In this paper, we set $i = 32$, and a symmetric state consists of two repetitions of slices 0-31. Each sequence of slices (0-31, 32-63) is called a *consecutive slice set* or *CSS* in short.

Note that the round constants are not periodic, so the ι operation will introduce a difference between the two CSS's, and this difference will propagate through other operations. To characterize the difference, the internal difference is defined as follows. The set $\{v+u|u \text{ is symmetric}\}$ obtained by adding all symmetric states to a single state v is called *internal difference*, recorded as $[i, v]$. If $v = \mathbf{0}$, the subset $[i, \mathbf{0}]$ is called *zero internal difference*, and other internal differences $[i, v]$ are cosets of $[i, \mathbf{0}]$. The state v is called the *representative state*. We choose v satisfying $v[x][y][z] = 0$ ($z \in \{32, 33, ..., 63\}$) as the *canonical representative state*. For a state v, we refer to its corresponding canonical representative state of the internal difference as its internal difference, denoted by $\Delta(v)$. Then, an internal differential for round function $\mathtt{R}$ is a pair of internal differences (α_1, α_2), and its probability is defined as $\Pr(\Delta(\mathtt{R}(v)) = \alpha_2 | \Delta(v) = \alpha_1)$.

Similarly to the standard differential characteristics, an internal differential characteristic is defined as the propagation of internal differences through round function. The internal differential transition of the j-th round in the characteristic is denoted by $\alpha_{j-1} \xrightarrow{L} \beta_{j-1} \xrightarrow{\chi} \alpha_j^\star \xrightarrow{\iota} \alpha_j$, where $j \geq 1$.

Definition 1. [29] *Given the non-zero input difference $\delta_{in} = (\delta_0, \ldots, \delta_4)^T$ of the 5-bit* KECCAK *Sbox, the output difference δ_{out} is determined by q ($2 \leq q \leq 4$) linear conditions with respect to the actual input $x = (x_0, \ldots, x_4)^T$. The q linear conditions $\{L_t(x)\}_{t=0}^{q-1}$ (without constant terms) are called differential transition conditions. Equivalently, $\delta_{out} = \mathtt{S}(x) \oplus \mathtt{S}(x \oplus \delta_{in}) = C \cdot x \oplus \eta$, where $C \in \mathbb{F}_2^{5\times 5}$ is a matrix ($rank(C) \in \{2, 3, 4\}$) and $\eta \in \mathbb{F}_2^5$ is a constant vector. It can be easily verified that C and η can be represented by δ_{in} as*

$$C = \begin{pmatrix} \delta_2 & \delta_1 & & & \\ & \delta_3 & \delta_2 & & \\ & & \delta_4 & \delta_3 & \\ \delta_4 & & & \delta_0 & \\ \delta_1 & \delta_0 & & & \end{pmatrix}, \eta = \mathtt{S}(\delta_{in}) = \begin{pmatrix} \delta_0 \oplus (\delta_1 \oplus 1)\delta_2 \\ \delta_1 \oplus (\delta_2 \oplus 1)\delta_3 \\ \delta_2 \oplus (\delta_3 \oplus 1)\delta_4 \\ \delta_3 \oplus (\delta_4 \oplus 1)\delta_0 \\ \delta_4 \oplus (\delta_0 \oplus 1)\delta_1 \end{pmatrix}.$$

Definition 2 (Transition Condition Number [29]). *Given an internal differential characteristic, for the input internal difference β_{j-1} of the j-th χ oper-*

ation, the rank of the set of all differential transition conditions obtained from β_{j-1} is called the transition condition number (denoted as k_j).

Property 1. [29] If the transition condition number of β_{j-1} is k, there are at most 2^k possible output internal differences and the lower bound of transition probability is 2^{-k}.

4.3 Target Internal Difference Algorithm

In [6], Dinur *et al.* proposed the *target internal difference algorithm* (TIDA) for constructing 1-round connector of internal differentials. The 1-round internal connector with the target internal difference α_1 can be constructed by finding a 2-block message $M_0||M_1$, such that

$$\Delta(\mathtt{R}(\mathtt{R}^{n_r}(M_0||0^c) \oplus (\overline{M_1}||0^c))) = \alpha_1. \tag{1}$$

Zhang *et al.* redesigned TIDA for conditional internal differentials in [29] and further improved the algorithm by probabilistic linearization method in [30]. Its essence is to transform the system (1) into two linear systems E_Δ and $E_{\Delta(L(\alpha_0))\to\alpha_1^\star}$, in a probabilistic way rather than in a deterministic way. The system E_Δ is called the *input difference system* (Def. 4), and its solution space contains correct input internal differences with a probability of $p_1^\star$. That is, solving $1/p_1^\star$ systems of E_Δ's gives an input internal difference α_0 propagated to α_1 on average. The system $E_{\Delta(L(\alpha_0))\to\alpha_1^\star}$ is *differential transition system* (Def. 5), used to determine whether the input internal difference α_0 can be propagated to α_1 for legal messages. To describe the algorithm, we recall the concept of difference density in [30].

Definition 3. [30] *Given a non-zero output difference δ_{out} of the* Keccak *Sbox, for the t-dimensional affine space U ($1 \le t \le 5$), the proportion of the input differences of δ_{out} in U is called the difference density of U with respect to δ_{out}, recorded as* $\mathrm{P}(U, \delta_{out})$.

We now recall the establishment process of linear system E_Δ. The first step is to select an 800-dimensional subspace (named W) to which the initial internal difference $\alpha_0 = (\Delta_\mathcal{R}, \Delta_\mathcal{C})$ propagates through the linear layer L, that is, $W = L(\Delta_\mathcal{R}, \Delta_\mathcal{C})$, where $\Delta_\mathcal{R}$ and $\Delta_\mathcal{C}$ respectively denote the internal differences in the rate part and capacity part. For each non-active Sbox involved in $\alpha_1^\star$, it is still constrained by five linear equations. For an active Sbox, given the output difference δ_{out}, we select an t-dimensional affine subspace U and add the corresponding $(5-t)$ linear equations to the system. The other details of establishment are shown in Procedure PIDS [30]. Assuming that E_Δ is consistent and $\Delta(L^{-1}(\beta_0))$ is a solution to E_Δ, the probability $p_1^\star$ of β_0 being the input difference of $\alpha_1^\star$ is determined by the product of the corresponding difference density of each selected affine subspace.

Definition 4 (Input Difference System). [30] *In internal differentials of* `SHA-3`*, given the characteristic starting from the second round, the linear system*

with respect to the input difference α_0 of the first round is the input difference system, regarded as E_Δ. The last $(c/2+p)$ bits of α_0 and the last $(c+p)$ bits of the input state of the second block are defined as padding and inner bits (or inner bits for short), which are known but can not be controlled. After applying Gaussian elimination to E_Δ, the equations related only to the inner bits are called the inner part (or inner system) of E_Δ, denoted as $E_\mathcal{C}$.

Definition 5 (Differential Transition System). [30] *Given an input difference β and its output difference α after χ, the linear system composed of all differential transition conditions and their values of the constant terms is called the differential transition system from β to α, regarded as $E_{\beta\to\alpha}$.*

5 Preimage Attacks Using Internal Differentials

In this section, we first describe the framework of preimage attacks on the `SHA-3` functions with reduced rounds. Afterwards we present the technical details of each phase in the attack and some optimizations for improving the attack.

5.1 The Attack Framework

Inspired by the work of Dinur *et al.* [6] and Zhang *et al.* [29,30] on collision attacks, we propose a novel attack framework for preimage attacks. In the attack, we utilize two consecutive internal differential characteristics in opposite directions. Taking the 5-round preimage attack as an example, the characteristic covering 4 rounds in the forward phase starts from the second round, and the characteristic covering 3.5 rounds in the backward phase starts from the nonlinear layer of the second-to-last round. Our preimage attack consists of three phases, namely the forward phase, backward phase, and collision phase, and we select 5-block messages as inputs, as depicted in Figure 4. Assume that the forward internal differential characteristic is $\alpha_1 \xrightarrow{R} \alpha_2 \xrightarrow{R} \alpha_3 \xrightarrow{R} \alpha_4 \xrightarrow{R} \overline{\alpha_5}$ and the backward internal differential characteristic is $\beta'_3 \xrightarrow{L^{-1}} \alpha'_3 \xrightarrow{R^{-1}} \alpha'_2 \xrightarrow{R^{-1}} \alpha'_1 \xrightarrow{R^{-1}} \overline{\alpha'_0}$, where $\overline{\alpha_5} = \overline{\alpha'_0}$ is truncated internal difference at the capacity part. Next, we give an overview of the three phases.

Phase I – Forward Phase. This phase includes two stages: Forward-TIDA and Forward-Collecting Messages.

- Forward-TIDA stage: For the target internal difference α_1, we establish the input difference system E_Δ and filter out the first blocks M_0 that make E_Δ consistent. After that, select α_0 from the solution space of each E_Δ that can be legally propagated to α_1.
- Forward-Collecting Messages stage: Perform linearizations on the first nonlinear layer χ and solve the differential transition systems $E_{\beta_1\to\alpha_2^\star}$ to get several subspaces of the second block M_1 passing the first two rounds. Then, compute the messages after 5 rounds functions from the subspaces and filter out the states where the truncated internal difference is $\overline{\alpha_5}$ to store in the set D'_L.

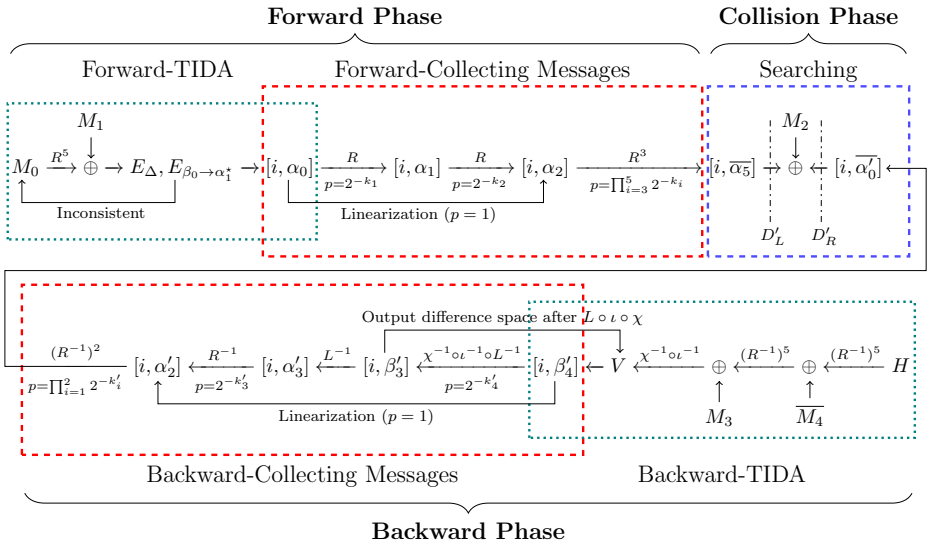


Fig. 4: The framework of 5-round preimage attack

Phase II – Backward Phase. This phase includes Backward-TIDA stage and Backward-Collecting Messages stage.

- Backward-TIDA stage: In this stage, we start with the digest H to be recovered and first append a suffix of '0' bits to H to pad it to 1600 bits, denoted by $\overline{H}$. The target internal difference is β_3'. After β_3' passes through χ, all possible output differences form an affine space, denoted as V. We randomly select M_3 and M_4, invert the state from $\overline{H}$ through the underlying permutation twice, and reversely calculate half of the round function, *i.e.*, $\chi^{-1} \circ \iota^{-1}$, until the internal difference β_4' of the 4.5-th round appears in V.
- Backward-Collecting Messages stage: Modify the state value with internal difference β_4', perform linearizations on the nonlinear layer χ^{-1} of the fourth round, and solve the differential transition systems $E_{\beta_2' \leftarrow \alpha_3'^{\star}}$ to obtain the states entering $[i, \alpha_2']$. Then continue to calculate backward and collect the states with truncated internal difference $\overline{\alpha_0'}$ and store them in the set D_R'.

Phase III – Collision Phase. This phase only includes the searching stage.

- Searching stage: Using hash table techniques, perform an exhaustive search in D_L' and D_R' until two states with the same capacity value belonging to different sets are found. Finally, modify the third block M_2 to complete the matching of the entire state.

Remark 1. The fifth block M_4 is introduced here to eliminate the impact of `SHA-3`'s padding rule on the degree of freedom. In this case, the fourth block M_3 is equivalent to the internal state where the first r bits are free variables and the last c bits are all zero.

Remark 2. Since the χ operation can be applied to each plane independently, we modify the values of the first few planes in the rate part in $[i, \beta_4']$, which is equivalent to modifying the fourth block M_3. In the Backward-Collecting Messages stage, we start from the set $[i, \beta_4']$ and calculate backward. After a match is achieved in the collision phase, we calculate the value of M_3 to complete the entire attack.

5.2 Forward Phase

Forward-TIDA stage. The Forward-TIDA stage can be abstracted as Algorithm 1, which modifies [30, Algorithm 1] and does not consider the padding bits. In this stage, for each active Sbox, we only keep the input differences with a transition condition number of 3 and restrict the possible input differences to a 1-dimensional affine subspace U. This is always possible because from the DDT, each non-zero output difference has at least two input differences with a transition condition number of 3, and any two non-zero vectors can always span a 1-dimensional affine space. For example, all possible input differences δ_{in} and transition condition numbers k of $\delta_{out} = 0\text{x}05$ are listed as follows. We choose 0x06 and 0x16 as possible input differences, and the subspace $\{0\text{x}06, 0\text{x}16\}$ can be characterized by four linear equations $\{(\delta_0, \ldots, \delta_4) \in \mathbb{F}_2^5 | \delta_0 = 0, \delta_1 = 1, \delta_2 = 1, \delta_3 = 0\}$.

δ_{in}	0x04	0x06	0x07	0x0f	0x11	0x16	0x17	0x19	0x1b	0x1d
k	2	3	4	4	3	3	4	4	4	4

Correspondingly, in the Procedure PIDS, for each active Sbox, we add 4 linear equations to the system E_Δ, and the probability $p_1^\star = 1$. Assuming that the total number of active Sboxes is m, the difference transition condition number of the input difference β_0 obtained by solving E_Δ is $3m$. From Observation 2 [24], it can be seen that solving the differential transition system $E_{\beta_0 \to \alpha_1^\star}$ can linearize all active Sboxes. For each non-active Sbox, we add three conditions to $E_{\beta_0 \to \alpha_1^\star}$ that can linearize it, and solve $E_{\beta_0 \to \alpha_1^\star}$ to linearzing the first nonlinear layer χ. Since the total number of Sboxes is 160, there are at most 480 independent linear equations in $E_{\beta_0 \to \alpha_1^\star}$, which means that completing the first round of linearization consumes at most 480 degrees of freedom. For `SHAKE128`, `SHA3-224`, `SHA3-256` and `SHAKE256`, more degrees of freedom are still reserved for passing subsequent rounds.

Forward-Collecting Messages stage. In this stage, we solve the differential transition system $E_{\beta_1 \to \alpha_2^\star}$ to obtain the solution space W_2. After the first round of linearization, $E_{\beta_1 \to \alpha_2^\star}$ is a linear system about the initial message. The details of the linearization are presented in Supplementary Material D in the full version of this paper.

Assuming the q_1 differential transition conditions are satisfied, the initial message in the solution space W_2 can pass through the first two rounds of the forward characteristic with a probability $2^{-k_1-k_2+q_1}$. We select enough initial

Algorithm 1: Forward-TIDA

Input: Target internal difference α_1 and target number of attacked rounds n_r
Output: the first block M_0, the value subspace W_1 of the second M_1, initial internal difference α_0

Set $E_\Delta = \emptyset, \alpha_1^\star = \alpha_1 \oplus RC[1]$ and $p_1^\star = 1$.
$(E_\Delta(\Delta_\mathcal{R}, \Delta_\mathcal{C}), p_1^\star) = \text{PIDS}(\Delta_T, p_1^\star)$.
Reduce $E_\Delta(\Delta_\mathcal{R}, \Delta_\mathcal{C}) = E_\mathcal{R}(\Delta_\mathcal{R}, \Delta_\mathcal{C}) \cup E_\mathcal{C}(\Delta_\mathcal{C})$.
Set W_1 to NULL.
do
  Set $\Delta_c^\star = (\delta'_{800-c/2}, ..., \delta'_{799})$.
  do
    Randomly select M_0 and compute $\Delta(\mathtt{R}^{n_r}(M_0))$.
    for *each integer* $j \in [800 - c/2, 800)$ **do**
      $\delta'_j = \Delta(\mathtt{R}^{n_r}(M_0))[j]$. `// the` j`-th bit of` $\Delta(\mathtt{R}^{n_r}(M_0))$
    end
  while $\Delta_c^\star$ *is not a solution of* $E_\mathcal{C}$;
  Solve $E_\mathcal{R}(\Delta_\mathcal{R}, \Delta_\mathcal{C}^\star)$ and obtain its solution space $U_\mathcal{C}$.
  do
    Randomly choose and delete a solution α_0 of $U_\mathcal{C}$.
    $\beta_0 = \Delta(L(\alpha_0))$.
    if β_0 *is the input difference of* $\alpha_1^\star$ **then**
      Obtain the differential transition system $E_{\beta_0 \to \alpha_1^\star}$. `// as defined in Def. 5`
      Solve the linear system $E_{\beta_0 \to \alpha_1^\star}(\mathtt{R}^{n_r}(M_0) \oplus (X||0^c))$ on X, and get its solution space W_1 if it has solutions.
    end
  while W_1 *is* NULL *and* $U_\mathcal{C} \neq \emptyset$;
while W_1 *is* NULL;
return (M_0, W_1, α_0)

messages from W_2 for calculation. Store the filtered messages in the set D'_L at the end of the forward internal differential characteristic.

Remark 3. In the attacks on 4-round `SHA3-384` and 3-round `SHA3-512`, we only add 1 linear equations for each active Sbox with difference density of 9/32 to the input difference system E_Δ and the corresponding difference density becomes 9/16. Each non-active Sbox still provides 5 equations. The probability $p_1^\star$ is therefore equal to the product of the difference densities of all active Sboxes in 5-dimensional space. After solving E_Δ and obtaining the input differential β_0, we solve the differential transition system $E_{\beta_0 \to \alpha_1^\star}$ and obtain the solution space W_1. In the Forward-Collecting Messages stage, we do not perform linearization, but simply collect messages at the end of the forward characteristic.

5.3 Backward Phase

Backward-TIDA stage. In this stage, our objective is to find the fifth block M_4 and multiple sets W_3 of M_3 that satisfy the target difference β'_{n_r-2}.

For a given β'_{n_r-2}, we consider the internal differential characteristic

$$\beta'_{n_r-2} \xrightarrow{\iota\circ\chi} \alpha'_{n_r-1} \xrightarrow{L} \beta'_{n_r-1} \xrightarrow{\iota\circ\chi} \alpha'_{n_r} = \Delta M,$$

where $M = \chi^{-1}\circ\iota^{-1}(M_3 \oplus \mathtt{R}^{-n_r}(\mathtt{R}^{-n_r}(\overline{H}) \oplus \overline{M_4}))$. Since for any internal difference input to χ, its output differences form an affine space, we know all the possible differences β'_{n_r-1}'s form an affine space, denoted by V. In the following, we discuss how to find M_3 and M_4 such that ΔM matches V.

A property of the χ operation can be exploited here that χ acts independently on each plane and is a reversible operation. In the n_r-round attacks, taking `SHA3-384` as example, the first two planes of the fourth message block M_3 are free variables, so we can modify M_3 to ensure that the projection of the internal difference ΔM matches V on the first two planes. At this point, we only need to randomly select the value of M_4 so that the last three planes of the difference ΔM align with an arbitrary element β'_{n_r-1} in V. However, for all `SHA-3` functions, there is always one plane that cannot be fully covered by the rate part (since each plane has 320 bits). To address this issue, we introduced the *value-difference distribution table* (`VDDT`).

Definition 6 (Value-Difference Distribution Table). *Set Sbox* $\mathtt{S} : \mathbb{F}_2^5 \to \mathbb{F}_2^5$*. Given any input difference* δ_{in} *and* $2t$*-bit value* (y, y')*,* $\delta_{in} \in \mathbb{F}_2^5, y, y' \in \mathbb{F}_2^t$*, the entry* $\mathtt{VDDT}(\alpha, y, y')$ *in the value-difference distribution table records the number of elements in the set* $\{(x, x') \in \mathbb{F}_2^{2\times(5-t)} | \mathtt{S}^{-1}(x||y) + \mathtt{S}^{-1}(x'||y') = \delta_{in}\}$.

Taking `SHA3-384` as an example, its capacity part is the last 12 lanes of the state, then the last two lanes in the third plane are not controlled by the fourth block M_3. We first randomly take the fifth block M_4 to get the output difference β'_{n_r-1} that matches the last two planes. Then fix M_4, and check the `VDDT` with $2t$-bit value of each Sbox in the second plane to determine whether β'_{n_r-2} is the output difference, where $t = 2$. If not, continue to randomly select M_4 until a matching output difference is obtained. Then we construct the *inverse differential transition system* according to the *inverse differential transition condition* (Def. 7) and to obtain the set W_3 about states in $[i, \beta'_{n_r-1}]$. The details of this stage are shown in Algorithm 2.

Definition 7 (Inverse Differential Transition Condition and Inverse Differential Transition System). *Given the non-zero output difference* δ_{out} *and the input difference* δ_{in} *of the 5-bit* Keccak *Sbox. If each output value* y *that satisfies* q *independent linear conditions is a solution to* $\delta_{in} = \mathtt{S}^{-1}(x) + \mathtt{S}^{-1}(x + \delta_{out})$*, then the* q *linear conditions are called inverse differential transition conditions. Given an output difference* α *and its input difference* β *before* χ*, the linear system composed of inverse differential transition conditions is called the inverse differential transition system from* α *to* β*, regarded as* $E_{\beta\leftarrow\alpha}$*.*

Backward-Collecting Messages stage. In this stage, for the attacks on 4-round `SHA3-384` and 3-round `SHA3-512`, we directly calculation the messages

Algorithm 2: Backward-TIDA

Input: Target internal difference β'_{n_r-2} and a digest H
Output: The fifth block M_4, the set W_3 of the fourth block M_3, internal difference β'_{n_r-1}

Compute the affine space V generated by β'_{n_r-2} after $L \circ \iota \circ \chi$ operation.
Set $\overline{H} = H||0\ldots0, N_p = \lfloor c/320 \rfloor, N_l = c/64 - 5 \cdot \lfloor c/320 \rfloor$.
/* N_p represents the maximum number of planes that can be covered by the capacity part in the internal difference */
do
  Det = 1.// Determine whether β'_{n_r-1} matches ΔM in capacity
  if $N_p > 0$ **then**
    do
      Randomly select M_4 and compute $M = \chi^{-1} \circ \iota^{-1} \circ \mathtt{R}^{-n_r}(\mathtt{R}^{-n_r}(\overline{H}) \oplus \overline{M_4})$.
    while *there is no* $\beta'_{n_r-1} \in V$ *matching* ΔM *on the last* N_p *planes*;
    for *each Sbox with* β'_4 *in the* $(5 - N_p)$*-th plane* **do**
      Get the input difference δ_{in} and the value (y, y') of the last $2N_l$ bits corresponding to M.
      if $\mathtt{VDDT}(\delta_{in}, y, y') = 0$ **then**
        Det = 0, break.
      end
    end
  else
    Randomly select M_3, M_4 and compute $M = \chi^{-1} \circ \iota^{-1}(M_3 \oplus \mathtt{R}^{-n_r}(\mathtt{R}^{-n_r}(\overline{H}) \oplus \overline{M_4}))$.
    Randomly select $\beta'_{n_r-1} \in V$.
    if β'_{n_r-1} *does not match* ΔM *on the last plane* **then**
      Det = 0.
    end
  end
while Det = 0;
Solve all linear systems $E_{\beta'_{n_r-2} \leftarrow \alpha'^{\star}_{n_r-1}}(L^{-1}(X))$ on X, and get the union W_3 of their solution spaces.// as defined in Def. 7
return $(M_0, W_3, \beta'_{n_r-1})$

in W_3 backward and store the states in the set D'_R. For the other versions, the optimization objective is to solve the subspace W_4 of W_3, in which the states can pass through the backward characteristic with higher probability. Let the state in set $[i, \beta'_{n_r-1}]$ be $X \oplus \beta'_{n_r-1}$, where X is a symmetric state. We add additional linear equations to $E_{\beta'_{n_r-2} \leftarrow \alpha'^{\star}_{n_r-1}}(L^{-1}(X))$ to linearize χ^{-1}, then solve the inverse differential transition system $E_{\beta'_{n_r-3} \leftarrow \alpha'^{\star}_{n_r-2}}$ about X and obtain the solution set W_4. Then select states from W_4 and repeat the backward calculation and storage steps.

In fact, the process of constructing the inverse differential transition system about χ^{-1} is different from the forward phase. In order to linearize χ^{-1}, we introduce the concept of *inverse-linearizable affine subspace* (Def. 8).

Definition 8 (Inverse-Linearizable affine subspace). *Inverse-Linearizable affine subspaces are affine output subspaces on which the inverse of Sbox substitution is equivalent to a linear transformation. If V is a inverse-linearizable affine subspaces of an Sbox operation $\mathtt{S}(\cdot)$, $\forall y \in V$, $\mathtt{S}^{-1}(y) = A \cdot y + b$, where A is a matrix and b is a constant vector.*

Exhaustive search for the inverse-linearizable affine subspaces of the Keccak Sbox shows:

Observation 1 *Given $\delta_{in}, \delta_{out} \in \mathbb{F}_2^5$, denote the set $V = \{y = (y_0, ..., y_4) : \mathtt{S}^{-1}(y) + \mathtt{S}^{-1}(y + \delta_{out}) = \delta_{in}\}$ and $\mathtt{S}^{-1}(V) = \{\mathtt{S}^{-1}(y) : y \in V\}$, we have*

a. *There are totally 80 2-dimensional inverse-linearizable affine subspaces, as listed in Table 4 in Supplementary Material F in our full version paper. And there are 8 subspaces, which are a partition of $\mathbb{F}_2^5$. There does not exit any inverse-linearizable affine subspace with dimension 3 or more.*
b. *if $\mathtt{DDT}(\delta_{in}, \delta_{out}) = 2$ or 4, then V is an inverse-linearizable affine subspace.*
c. *if $\mathtt{DDT}(\delta_{in}, \delta_{out}) = 8$, then there are two 2-dimensional inverse-linearizable subspaces $W_i \subset V$, $i = 0, 1$, such that $W_0 \cup W_1 = V$. And there are two independent linear conditions L_1 and L_2, all elements in V satisfy L_1, and 6 elements satisfy L_2.*

δ_{out}	V	L_1	L_2
0x01	0x14,0x15,0x11,0x10,0x1a,0x1b,0x1d,0x1c	$y_4 = 1$	$y_1 = 0$
0x11	0x06,0x17,0x02,0x13,0x08,0x19,0x0e,0x1f	$y_4 + y_0 = 0$	$y_1 = 1$
0x09	0x00,0x09,0x05,0x0c,0x0a,0x03,0x0d,0x04	$y_4 = 0$	$y_1 = 0$
0x19	0x12,0x0b,0x16,0x0f,0x18,0x01,0x1e,0x07	$y_4 + y_0 = 1$	$y_1 = 1$

We summarize all cases where $\delta_{in} = 0x01$ and $\mathtt{DDT}(\delta_{in}, \delta_{out}) = 8$ in the above table. Taking $(\delta_{in}, \delta_{out}) = (0\text{x}01, 0\text{x}11)$ as an example, the solution set $V = W_0 \cup W_1$, where $W_0 = \{y = (y_0, ..., y_4) : y_0 + y_4 = 0, y_1 = 1, y_3 = 0\}$ and $W_1 = \{y = (y_0, ..., y_4) : y_0 + y_4 = 0, y_1 + y_2 = 0, y_3 = 1\}$. We call the linear condition that all values in V satisfy *the first linear condition*, such as $L_1 : y_4 + y_0 = 0$; and the linear condition that only 3/4 of the values in V satisfy is called the *the second linear condition*, such as $L_2 : y_1 = 1$. For the Sbox with $\mathtt{DDT} \leq 4$, the inverse differential transition conditions are all the first linear conditions. For the Sbox with $\mathtt{DDT} = 8$, there is one first linear condition and two second linear conditions. In the j-th round, assuming that the number of Sboxes with transition probabilities of $2^{-2}, 2^{-3}$, and 2^{-4} are $N_{k=2}, N_{k=3}$, and $N_{k=4}$ respectively, the differential transition probability is $2^{-(2N_{k=2}+3N_{k=3}+4N_{k=4})}$. After adding all the first linear conditions, there are $N_{k=2} + 3N_{k=3} + 4N_{k=4}$ equations in the inverse differential transition system $E_{\beta \leftarrow \alpha}$. The transition probability of the state obtained by solving $E_{\beta \leftarrow \alpha}$ is $2^{-N_{k=2}}$. If we add q second linear conditions belonging to different Sboxes to $E_{\beta \leftarrow \alpha}$ and solve the system, the transition probability of

the solution is $2^{-(N_{k=2}-q)} \cdot (3/4)^q = 2^{-N_{k=2}+q\cdot\log_2(3/2)}$. When there are sufficient degrees of freedom, for each Sbox with transition probability of 2^{-2}, we can add three inverse differential transition conditions to the $E_{\beta\leftarrow\alpha}$ to characterize the inverse-linearizable affine subspace (as W_0), so that the solved state deterministically pass χ^{-1}. The degrees of freedom consumed in solving system $E_{\beta\leftarrow\alpha}$ are $3N_{k=2} + 3N_{k=3} + 4N_{k=4} = N_{k=2} + k'_j$. In the attack, we adjust the number of equations according to different versions. From Observation 1, to linearize χ^{-1}, we also need to add at least 3 linear conditions to the output space of each Sbox.

Remark 4. Since the linearization technique is performed on χ^{-1}, the linear system contains an inverse difference transition system $E_{\beta'_{n_r-2}\leftarrow\alpha'^{\star}_{n_r-1}}$, so the complexity of solving the inverse difference transition systems in Algorithm 2 will be calculated in the total complexity of the linearization operation. For the attack without using linearization in the backward phase, the complexity of solving the inverse difference transition systems $E_{\beta'_{n_r-2}\leftarrow\alpha'^{\star}_{n_r-1}}$, *i.e.*, Step 23 of Algorithm 2, is not the dominant in the algorithm.

5.4 Collision Phase and Time Complexity

Collision Phase. In the Searching stage, we use hash table technique to exhaustively search the two sets D'_L and D'_R. Store the states in the set D'_R into hash table T_H, and traverse the elements in D'_L until an element v_1 is found that has the same value as the state v_2 in D'_R in the capacity part.

Set the size of D'_L is 2^{n_1} and the size of D'_R is 2^{n_2}. From Section 3.2, we know that the number of collected messages needs to satisfy $2^{n_1} \cdot 2^{n_2} = 2^c$ in order to find a collision in the capacity part with a success rate of 0.63. Assuming that the transition probability of the forward characteristic is 2^{-K_1} and the probability of the backward characteristic is 2^{-K_2}, the number of initial messages required in the forward phase is $2^{n_1+K_1}$, and the number of initial messages required in the backward phase is $2^{n_2+K_2}$. The complexity of the our attack is mainly dominated by the complexity of the forward phase and backward phase, and the time complexity of the collision phase can be ignored in comparison.

Complexity. The time complexity is determined by the complexity of the following parts. In the following analysis, j takes 1 or 2, where $j = 1$ means that this stage is in the forward phase, and $j = 2$ means that this phase is in the backward phase.

1. TIDA stage: Let the complexity of running a TIDA be 2^{I_j}, and the size of the initial message space is 2^{d_j}. Then the complexity $T_{A,j}$ of this stage is $2^{I_j} \cdot \max\{2^{n_j+K_j-d_j}, 1\}$.
2. Collecting Messages stage: Let the complexity of a linearization be 2^{L_j}, and linearization is performed 2^{s_j} times. The complexity $T_{B,j}$ of performing linearization is $2^{L_j+s_j}$. In addition, there are q_j differential transition conditions that are satisfied in this stage, so the complexity $T_{C,j}$ of calculating the n_r-round permutations of the selected messages is $2^{n_j+K_j-q_j}$. The total complexity of this part is $T_{B,j} + T_{C,j} = 2^{L_j+s_j} + 2^{n_j+K_j-q_j}$.

Thus, the final complexity T of preimage attack is summarized in Equation (2),

$$\sum_{j=1}^{2}(T_{A,j}+T_{B,j}+T_{C,j}) = \sum_{j=1}^{2}(2^{I_j}\cdot\max\{2^{n_j+K_j-d_j},1\}+2^{L_j+s_j}+2^{n_j+K_j-q_j}). \quad (2)$$

Remark 5. In the attack on 4-round `SHA3-384` and 3-round `SHA3-512`, we do not implement linearization in both the forward phase and backward phase. Therefore, the total complexity T is $T_{A,1} + T_{C,1} + T_{A,2} + T_{C,2}$.

6 Results and Complexity Analysis

In this section, we present the details of our preimage attacks on round-reduced `SHA-3`. Given a forward internal differential characteristic and a backward characteristic, the procedure of the attack is depicted as follows.

1. The digest H of length d bits is padded to $\overline{H}$ of length 1600 bits by adding an arbitrary $(1600-d)$-bit suffix. Calculate the number N_1 and N_2 of required initial messages for the forward phase and backward phase respectively based on the internal difference characteristic.
2. Run Algorithm 1 to obtain the first block M_0, internal difference α_0, and the value subspace W_1 of the second block M_1. If there are sufficient degrees of freedom in W_1, utilize the linearization technique for the first nonlinear layer χ and solve the differential transition system $E_{\beta_1\to\alpha_2^\star}$ to obtain the subspace W_2 of W_1.
3. Traverse the messages in the affine space W_1 or W_2, and propagate the truncated difference after n_r rounds. Store the states that pass the forward internal difference characteristics into the set D'_L. If the size of D'_L is less than N_1, go back to Step 2.
4. Run Algorithm 2 to obtain the fifth block M_4, internal difference β'_{n_r-1}, and the value set W_3 of $[i,\beta'_{n_r-1}]$. If there are sufficient degrees of freedom in W_3, utilize the linearization technique for the (n_r-1)th nonlinear layer χ^{-1} and solve the differential transition system $E_{\beta'_{n_r-2}\leftarrow\alpha'^\star_{n_r-1}}$ to obtain the subspace W_4 of W_3.
5. Traverse the messages in the set W_3 or the affine space W_4, and propagate the truncated difference backward through n_r rounds. Store the states that pass the backward internal difference characteristics into the set D'_R. If the size of D'_R is less than N_2, go back to Step 4.
6. Exhaustively search the sets D'_L and D'_R until we find states v_1 and v_2 that are consistent in the capacity part.
7. Calculate the third block $M_2 = v_1 \oplus v_2$ and the fourth block $M_3 = R^{n_r}(R^{n_r}(R^{n_r}(M_0)\oplus M_1)\oplus M_2)\oplus(R^{-1})^{n_r}((R^{-1})^{n_r}(\overline{H})\oplus\overline{M_4})$.
8. Return a preimage $M_0||M_1||M_2||M_3||\overline{M_4}$ of the digest H.

Remark 6. In this paper, we use the Strassen algorithm to solve linear systems with a complexity of μ^ω bit operations, where $\omega = log_2(7)$ and μ is the maximum between the number of variables and the number of equations in the linear system. In the following, $(\mu^\omega)^\dagger$ means μ^ω bit operations.

Remark 7. Our attack eliminates the effect of padding bits by introducing the fifth block M_4. For any digest of length within r bits, the cost of a preimage attack is equivalent to that of attacking a digest of length $c/2$. Consequently, the attack procedures for `SHA3-256` and `SHAKE256` are identical, and we only describe the attack of `SHA3-256`. But it is important to stress here that our preimage attacks on round-reduced variants of `SHAKE` are valid for `SHAKE128`(M, d) and `SHAKE256`(M, d) with $d \leq 1344$ and $d \leq 1088$ respectively, including `SHAKE128`$(M, 256)$ and `SHAKE256`$(M, 512)$.

6.1 A Preimage Attack on 5-round SHAKE128

For 5-round `SHAKE128`, we use the internal differential characteristic given in Characteristic 1 in Supplementary Material G in the full version of this paper. The transition condition numbers are $(k_2, k_3, k_4, \overline{k_5}, k'_1, k'_2, k'_3, k'_4) = (155, 34, 15, 11, 7, 4, 98, 399)$. The scales of the sets D'_L and D'_R in the collision phase satisfy $2^{n_1+n_2} = 2^{128}$. In this subsection and subsequent chapters, we denote the number of Sboxes with transition probabilities of $2^{-2}, 2^{-3}$ and 2^{-4} in the j-th round of the backward phase as $N_{j,k=2}, N_{j,k=3}$ and $N_{j,k=4}$. We continue to use the notation $T_{A,j}, T_{B,j}, T_{C,j}$ $(j = 1, 2)$ in the complexity analysis in Section 5.4.

Forward Phase. For $\alpha_1^\star$, there are 124 active Sboxes and 36 non-active Sboxes. During the Forward-TIDA stage, for the input difference system E_Δ, each active Sbox provides 4 linear equations, and each non-active Sbox provides 5 linear equations. Thus the system E_Δ contains 676 equations, of which the inner part contains 67 equations. The output probability $p_1^\star$ of Procedure PIDS is 1. Therefore, the complexity of TIDA is 2^{67}. Since the transition condition number k_1 of the first round is limited to $124 \times 3 = 372$, the degree of freedom of the message space W_1 output by Algorithm 1 is at least $r/2 - 372 = 300$. The number of Forward-TIDA executions is $\max(2^{n_1+k_2+k_3+k_4+\overline{k_5}-300}, 1) = \max(2^{n_1-85}, 1)$. The complexity of this stage is $T_{A,1} = 2^{67} \cdot \max(2^{n_1-85}, 1) = \max(2^{n_1-18}, 2^{67})$.

In the collecting messages stage, there are 372 fixed differential transition conditions among the 480 linear conditions that need to be added in the linearization process. Each linearization is equivalent to solving a linear system with 300 variables and $36 \times 3 + k_2 = 263$ equations, with a complexity of $300^\omega = (2^{23.12})^\dagger$. The iteration number of linearization is $\max(2^{n_1+k_2+k_3+k_4+\overline{k_5}-(r/2-480)}, 1) = \max(2^{n_1-23}, 1)$. The total complexity $T_{B,1}$ of linearization is $(\max(2^{n_1+0.12}, 2^{23.12}))^\dagger$, and the complexity of absorbing and filtering the second blocks is $T_{C,1} = 2^{n_1+k_3+k_4+\overline{k_5}} = 2^{n_1+60}$.

Backward Phase. For the subspace V composed of β'_4, the dimension is $k_4 = 399$, and its projection dimension on the last plane is 160. Therefore, in each run of Backward-TIDA, only one M_4 needs to be randomly selected, and the complexity is 1. The degree of freedom of the message set W_3 output by Algorithm 2 is at least $20 \times 32 - 399 = 241$. The number of TIDA runs is $\max(2^{n_2+98-241}, 1) = 1$, $T_{A,2} = 1 \cdot 1 = 1$.

In the collecting messages stage, there are 13 non-active Sboxes for β'_3, $(N_{3,k=2}, N_{4,k=2}, N_{4,k=4}) = (38, 48, 6)$. Each linearization is equivalent to solving a linear system with 241 variables and $13 \times 3 + N_{4,k=2} + k'_3 + N_{3,k=2} = 223$ equations, with complexity of $241^\omega = (2^{22.24})^\dagger$. Based on the analysis in Section 5.3, we add inverse differential transition conditions to each active Sbox in the third round, resulting in an additional consumption of $N_{3,k=2}$ degrees of freedom, so the number of linearizations is $\max(2^{n_2+k'_1+k'_2+k'_3+N_{3,k=2}-(20\times32-480)}, 1) = \max(2^{n_2-13}, 1)$. $T_{B,2} = (\max(2^{n_2+9.24}, 2^{22.24}))^\dagger$, $T_{C,2} = 2^{n_2+k'_1+k'_2} = 2^{n_2+11}$.

Complexity. The total complexity $T = T_{A,1} + T_{B,1} + T_{C,1} + T_{A,2} + T_{B,2} + T_{C,2} = \max(2^{n_1-18}, 2^{67}) + (\max(2^{n_1+0.12}, 2^{23.12}))^\dagger + 2^{n_1+60} + 1 + (\max(2^{n_2+9.24}, 2^{22.24}))^\dagger + 2^{n_2+11}$. We take $(n_1, n_2) = (39.5, 88.5)$, in which case the complexity T takes the minimum value $2^{100.5}$.

6.2 A Preimage Attack on 5-round SHA3-224

For 5-round `SHA3-224`, we use the internal differential characteristic given in Characteristic 2 in Supplementary Material G in our full version paper. The transition condition numbers are $(k_2, k_3, \overline{k_{4,5}}, k'_1, k'_2, k'_3, k'_4) = (158, 30, 17.64, 13, 8, 88, 384)$, where $\overline{k_{4,5}}$ is the transition probability from β_3 to the truncated internal difference $\overline{\alpha_5}$. The scales of the sets D'_L and D'_R in the collision phase satisfy $2^{n_1+n_2} = 2^{224}$.

Forward Phase. For $\alpha_1^\star$, there are 122 active Sboxes and 38 non-active Sboxes. During the Forward-TIDA stage, each active Sbox provides 4 linear equations, and each non-active Sbox provides 5 linear equations. The input difference system E_Δ contains 678 equations, of which the inner part contains 102 equations. The output probability $p_1^\star$ of Procedure PIDS is 1. Therefore, the complexity of TIDA is 2^{102}. Since the transition condition number k_1 of the first round is limited to $122 \times 3 = 366$, the degree of freedom of the message space W_1 output by Algorithm 1 is at least $800 - 224 - 366 = 210$. The complexity $T_{A,1}$ of this stage is $2^{102} \cdot \max(2^{n_1+k_2+k_3+\overline{k_{4,5}}-210}, 1) = \max(2^{n_1+97.64}, 2^{102})$.

In the collecting messages stage, the remaining degrees of freedom after each linearization are $r/2 - 480 = 96$, which can satisfy the 96 differential transition conditions in the second round. Each linearization is equivalent to solving a linear system with 210 variables and $38 \times 3 + 96 = 210$ equations, with complexity of $210^\omega = (2^{21.68})^\dagger$. The number of linearizations is $\max(2^{n_1+k_2+k_3+\overline{k_{4,5}}-(r/2-480)}, 1) = 2^{n_1+109.64}$. $T_{B,1} = (2^{n_1+131.32})^\dagger$, $T_{C,1} = 2^{n_1+(k_2-96)+k_3+\overline{k_{4,5}}} = 2^{n_1+109.64}$.

Backward Phase. For `SHA3-224`, the last 7 lanes of the state are the capacity part, in the fourth plane, the last two lanes are constant. In the Backward-TIDA stage, we consider the two Sboxes that generate the difference on the fourth plane. For the subspace V composed of β'_4, the dimension is $k_4 = 384$. Its projection dimensions on the second-to-last plane and the last two planes are 160 and 290 respectively. V can be equivalently expressed as a vector $\beta'_4(t)$, where t has 384 degrees of freedom. The second-to-last plane and the last two planes of $\beta'_4(t)$ have 160 and 290 degrees of freedom respectively. If the fourth plane of $\beta'_4(t)$ is fixed, $\beta'_4(t)$ is changed to $\beta'_4(t')$, 160 degrees of freedom of the last two planes

are consumed. The fifth plane of $\beta_4'(t')$ has 130 degrees of freedom. For any 4-bit value (y, y'), there is always an input difference δ_{in} such that $\texttt{VDDT}(\delta_{in}, y, y') = 4$, and the input value of the Sbox forms a 2-dimensional affine subspace. To determine β_4', we first randomly select M_4 and calculate $M = \mathtt{R}^{-5}(\mathtt{R}^{-5}(\overline{H}) + \overline{M_4})$. By choosing the input difference of each Sbox of $\Delta(M)$ in the fourth plane, fix the truncated difference of $\beta_4'(t)$ in the fourth plane to obtain $\beta_4'(t')$, where each Sbox satisfies the above conditions on VDDT. Let the affine subspace spanned by the fifth plane of $\beta_4'(t')$ be V', with a dimension of 130. Then check whether the fifth plane of $\Delta(\chi^{-1} \circ \iota^{-1}(M))$ is in V', with a probability of 2^{-30}. Therefore, in each run of Backward-TIDA, an average of 2^{30} the fifth blocks M_4 need to be randomly selected, and the complexity is 2^{30}. The degree of freedom of the message space W_3 output by Algorithm 2 is at least $17 \times 32 - 384 = 160$. In the third round, 15 additional the second linear conditions L_2 are added, the complexity $T_{A,2}$ of TIDA is at most $2^{30} \cdot \max(2^{n_2+k_1'+k_2'+k_3'+15-160}, 1) = \max(2^{n_2-6}, 2^{30})$.

In the collecting messages stage, there are 19 non-active Sboxes for β_3', $(N_{4,k=2}, N_{4,k=3}, N_{4,k=4}) = (41, 98, 2)$. Each linearization is equivalent to solving a linear system with $17 \times 32 - k_4' + N_{4,k=2} = 201$ variables and $19 \times 3 + N_{4,k=2} + N_{3,k=2} + 15 = 201$ equations, with complexity of $201^\omega = (2^{21.50})^\dagger$. In the inverse differential transition system $E_{\beta_2' \leftarrow \alpha_3'^\star}$, $(N_{3,k=2}, N_{3,k=3}, N_{3,k=4}) = (41, 3, 0)$, in addition to $41 + 2 \times 3 = 47$ first linear conditions, we also add 15 second linear conditions L_2 defined in Section 5.3 to $E_{\beta_2' \leftarrow \alpha_3'^\star}$. Then, we solve $E_{\beta_2' \leftarrow \alpha_3'^\star}$ to obtain the subspace W_4. The probability that a randomly selected state from W_4 can pass the third round χ^{-1} is $2^{-(k_3'-47-15)} \cdot (3/4)^{15} = 2^{-k_3'+55.77}$. The number of linearizations is $\max(2^{n_2+k_1'+k_2'+k_3'+15 \cdot \log_2(4/3)-(17\times 32-480-N_{4,k=4})}, 1) = 2^{n_2+53.23}$. $T_{B,2} = (2^{n_2+74.73})^\dagger$, $T_{C,2} = 2^{n_2+k_1'+k_2'+k_3'-55.77} = 2^{n_2+53.23}$.

Complexity. The total complexity $T = T_{A,1} + T_{B,1} + T_{C,1} + T_{A,2} + T_{B,2} + T_{C,2} = \max(2^{n_1+97.64}, 2^{102}) + (2^{n_1+131.32})^\dagger + 2^{n_1+47.64} + \max(2^{n_2-6}, 2^{30}) + (2^{n_2+74.73})^\dagger + 2^{n_2+53.23}$. We take $(n_1, n_2) = (83.71, 140.29)$, in which case the complexity T takes the minimum value $(2^{216.03})^\dagger$.

6.3 Preimage Attacks on 5-round SHA3-256/SHAKE256

For 5-round SHA3-256, we use the same characteristic used to attack 5-round SHA3-224. The scales of the sets D_L' and D_R' satisfy $2^{n_1+n_2} = 2^{256}$.

Forward Phase. During the Forward-TIDA stage, the input difference system E_Δ contains 678 equations, of which the inner part contains 134 equations. The output probability $p_1^\star$ of Procedure PIDS is 1. Therefore, the complexity of TIDA is 2^{134}. The degree of freedom of the message space W_1 output by Algorithm 1 is at least $800 - 256 - 366 = 178$. The complexity $T_{A,1}$ of Forward-TIDA stage is $2^{134} \cdot \max(2^{n_1+k_2+k_3+\overline{k_{4,5}}-178}, 1) = 2^{n_1+161.64}$.

In the collecting messages stage, the remaining degrees of freedom after each linearization are $r/2 - 480 = 64$, which can satisfy the 96 differential transition conditions in the second round. Each linearization is equivalent to solving a linear system with 178 variables and $38 \times 3 + 64 = 178$ equations, with complexity of $178^\omega = (2^{21.01})^\dagger$. The number of linearizations

is $\max(2^{n_1+k_2+k_3+\overline{k_{4,5}}-(r/2-480)}, 1) = 2^{n_1+141.64}$. $T_{B,1} = (2^{n_1+162.65})^\dagger, T_{C,1} = 2^{n_1+(k_2-64)+k_3+\overline{k_{4,5}}} = 2^{n_1+141.64}$.

Backward Phase. For `SHA3-256`, the last 8 lanes of the state are the capacity part, and in the fourth plane, the last three lanes are constant. In the Backward-TIDA stage, we first determine the value of the target internal difference on the fourth plane. For half of 6-bit values (y, y'), there is an input difference δ_{in} such that $\texttt{VDDT}(\delta_{in}, y, y') \geq 3$, and there are three input values of Sbox covered by a 2-dimensional affine space W_0. We first randomly select M_4 and check whether each Sbox in the fourth plane of $\Delta(\texttt{R}^{-5}(\texttt{R}^{-5}(\overline{H}) + \overline{M_4}))$ has an input difference that satisfies $\texttt{VDDT} \geq 3$, with a probability of 2^{-32}. Then follow the step in Section 6.2 to determine the fifth plane of β'_4, with a probability of 2^{-30}. Therefore, in each run of Backward-TIDA, an average of 2^{30+32} the fifth blocks M_4 needs to be randomly selected, and the complexity is 2^{62}. After obtaining the internal difference β'_4, take the subspace W_3 in $[i, \beta'_4]$ as the message space. The degrees of freedom of W_3 come from the first three planes and the two-dimensional affine subspace W_0 corresponding to each Sbox on and the fourth plane, which is $15 \times 32 + 2 \times 32 = 544$. In the attack, we constrain the subspace W_3 and select states from it, and then check whether the selected states can be obtained by modifying M_3. Since only 3 values in each W_0 can be obtained by modifying the value of M_3, the number of valid messages in W_3 is $2^{544} \cdot (3/4)^{32} = 2^{530.72}$. In the third round, 15 additional the second linear conditions L_2 are added, the complexity $T_{A,2}$ of TIDA is at most $2^{62} \cdot \max(2^{n_2+k'_1+k'_2+k'_3+15-(530.72-k'_4)}, 1) = 2^{n_2+39.28}$.

In the collecting messages stage, each linearization is equivalent to solving a linear system with 201 variables and 201 equations, with complexity of $201^\omega = (2^{21.50})^\dagger$. The process of constructing the inverse differential transition system $E_{\beta'_2 \leftarrow \alpha'^\star_3}$ is the same as that in Section 6.2. The number of linearizations is $\max(2^{n_2+k'_1+k'_2+k'_3+15\cdot\log_2(4/3)+32\cdot\log_2(4/3)-(17\times32-480-N_{4,k=4})}, 1) = 2^{n_2+66.51}$. $T_{B,2} = (2^{n_2+88.01})^\dagger, T_{C,2} = 2^{n_2+k'_1+k'_2+k'_3-55.77+32\cdot\log_2(4/3)} = 2^{n_2+66.51}$.

Complexity. The total complexity The total complexity $T = T_{A,1} + T_{B,1} + T_{C,1} + T_{A,2} + T_{B,2} + T_{C,2} = 2^{n_1+161.64} + (2^{n_1+162.65})^\dagger + 2^{n_1+141.64} + 2^{n_2+39.28} + (2^{n_2+88.01})^\dagger + 2^{n_2+66.51}$. We take $(n_1, n_2) = (90.68, 165.32)$, in which case the complexity T takes the minimum value $(2^{254.33})^\dagger$.

6.4 A Preimage Attack on 4-round SHA3-384

For 4-round `SHA3-384`, we use the internal differential characteristic given in Characteristic 3 in Supplementary Material G in our full version paper. The transition condition numbers are $(k_2, k_3, \overline{k_4}, k'_1, k'_2, k'_3) = (25, 18, 5, 33, 48, 210)$. The scales of the sets D'_L and D'_R in the collision phase satisfy $2^{n_1+n_2} = 2^{384}$.

Forward Phase. For $\alpha^\star_1$, there are 77 active Sboxes and 83 non-active Sboxes. During the Forward-TIDA stage, each non-active Sbox provides 5 linear equations, and 27 active Sboxes do not provide equations. The other 50 active Sboxes provide 1 linear equation, and the corresponding difference density changes from 9/32 to 9/16. The input difference system E_Δ contains 465

equations, of which the inner part contains 98 equations. The output probability $p_1^\star$ of Procedure PIDS is $2^{84.88}$. Therefore, the complexity of TIDA is $2^{84.88+98} = 2^{182.88}$. By the method of [30], since the expected transition condition number $\mathtt{E}(k_1)$ of the first round is 263, the average degree of freedom of the message space W_1 output by Algorithm 1 is at least $r/2 - 263 = 153$. $T_{A,1} = 2^{182.88} \cdot \max(2^{n_1+k_2+k_3+\overline{k_4}-153}, 1) = \max(2^{n_1+78.88}, 2^{182.88}), T_{C,1} = 2^{n_1+k_2+k_3+\overline{k_4}} = 2^{n_1+49}$.

Backward Phase. For `SHA3-384`, the last 12 lanes of the state are the capacity part, in the second plane, the last two lanes are constant. In the Back-TIDA stage, we consider the last two planes. For the subspace V composed of β_3', the dimension is $k_3 = 210$, and its projection dimension on the last two planes is 194. We follow the steps in Algorithm 2 to first find $\beta_2' \in V$ that matches the last two planes of $\Delta(\chi^{-1} \circ \iota^{-1} \circ \mathtt{R}^{-4}(\mathtt{R}^{-4}(\overline{H}) + \overline{M_4}))$, and then check whether β_2' satisfies $\mathtt{VDDT} > 0$ for each Sbox in the second plane. For half of 4-bit values (y, y'), there are 24 input differences δ_{in} that satisfy $\mathtt{VDDT}(\delta_{in}, y, y') > 0$; for the other half of 4-bit values, there are 14 input differences δ_{in} that satisfy $\mathtt{VDDT}(\delta_{in}, y, y') > 0$, and the minimum non-value is 2. Therefore, the complexity of running Backward-TIDA once is $(24/32)^{16} \cdot (14/32)^{16} \cdot 2^{10\times 32-194} = 2^{151.72}$. The degree of freedom of the message set W_3 output by Algorithm 2 is at least $10 \times 32 + 32 - 210 = 142$. The number of TIDA runs is $\max(2^{n_2+k_1'+k_2'-142}, 1) = \max(2^{n_2-61}, 1)$. $T_{A,2} = 2^{151.72} \cdot \max(2^{n_2-61}, 1) = \max(2^{n_2+90.72}, 2^{151.72}), T_{C,2} = 2^{n_2+k_1'+k_2'} = 2^{n_2+81}$.

Complexity. The total complexity $T = T_{A,1} + T_{C,1} + T_{A,2} + T_{C,2} = \max(2^{n_1+78.88}, 2^{182.88}) + 2^{n_1+49} + \max(2^{n_2+90.72}, 2^{151.72}) + 2^{n_2+81}$. We take $(n_1, n_2) = (197.92, 186.08)$, in which case T takes the minimum value $2^{277.8}$.

6.5 Preimage Attacks on 4-round SHAKE128/SHA3-224/SHA3-256/SHAKE256

For 4-round `SHAKE128`, `SHA3-224` and `SHA3-256`, we use internal differential characteristic given in Characteristic 4 in Supplementary Material G in the full version of this paper. The transition condition numbers are $(k_2, k_3, \overline{k_4}, k_1', k_2', k_3') = (24, 22, 0, 11, 12, 239)$. Next we describe the details of the attack on `SHA3-256`/`SHAKE256`, and discuss later the attacks on the other two versions.

Forward Phase. For $\alpha_1^\star$, there are 115 active Sboxes and 45 non-active Sboxes. During the Forward-TIDA stage, each active Sbox provides 4 linear equations, and each non-active Sbox provides 5 linear equations. The input difference system E_Δ contains 685 equations, of which the inner part contains 141 equations. The output probability $p_1^\star$ of Procedure PIDS is 1. Therefore, the complexity of TIDA is 2^{141}. Since the transition condition number k_1 of the first round is limited to $115 \times 3 = 345$, the degree of freedom of the message space W_1 output by Algorithm 1 is at least $800 - 256 - 345 = 199$. The complexity $T_{A,1}$ of Forward-TIDA stage is $2^{141} \cdot \max(2^{n_1+k_2+k_3+\overline{k_4}-199}, 1) = \max(2^{n_1-12}, 2^{141})$.

In the collecting messages stage, the remaining degrees of freedom after each linearization are $r/2-480 = 96$, which can satisfy all differential transition conditions in the second round. Each linearization is equivalent to solving a linear system with 199 variables and $45\times3+k_2 = 159$ equations, with complexity of $199^\omega = (2^{21.46})^\dagger$. The number of linearizations is $\max(2^{n_1+k_2+k_3+\overline{k_4}-(r/2-480)}, 1) = \max(2^{n_1-18}, 1)$. $T_{B,1} = (\max(2^{n_1+3.46}, 2^{21.46}))^\dagger, T_{C,1} = 2^{n_1+k_3+\overline{k_4}} = 2^{n_1+22}$.

Backward Phase. For the subspace V composed of β_3', the dimension is $k_3 = 284$, and its projection dimension on the last two planes is 201. In Backward-TIDA, we give up the degree of freedom of the fourth plane, randomly select M_4 and find $\beta_3' \in V$ that matches $\chi^{-1} \circ \mathtt{R}^{-4}(\mathtt{R}^{-4}(\overline{H}) \oplus M_4)$ in the last two planes. Therefore, in each run of Backward-TIDA, about $2^{320-201} = 2^{109}$ the fifth blocks M_4 need to be randomly selected, and the complexity is 2^{109}. The degree of freedom of the message set W_3 output by Algorithm 2 is at least $15 \times 32 - 284 = 196$. No linearization is performed in this phase. The number of Backward-TIDA runs is $\max(2^{n_2+k_1'+k_2'-196}, 1) = \max(2^{n_2-173}, 1)$. $T_{A,2} = 2^{109} \cdot \max(2^{n_2-173}, 1) = \max(2^{n_2-64}, 2^{109}), T_{C,2} = 2^{n_2+k_1'+k_2'} = 2^{n_2+23}$.

Complexity. The total complexity $T = T_{A,1} + T_{B,1} + T_{C,1} + T_{A,2} + T_{C,2} = \max(2^{n_1-12}, 2^{141}) + (\max(2^{n_1+3.46}, 2^{21.46}))^\dagger + 2^{n_1+22} + \max(2^{n_2-64}, 2^{109}) + 2^{n_2+23}$. We take $(n_1, n_2) = (128.5, 127.5)$, T takes the minimum value $2^{151.5}$.

For 4-round `SHAKE128` and `SHA3-224`, the attack has the same procedure as `SHA3-256`, with complexities of $2^{81.5}$ and $2^{135.5}$ respectively. The differences are that their inner parts in E_Δ contain less equations (73 for `SHAKE128` and 109 for `SHA3-224`) and the scales of the sets D_L' and D_R' in the collision phase satisfy $2^{n_1+n_2} = 2^{64}$ for `SHAKE128` and 2^{112} for `SHA3-224`, due to different capacities. Besides, the backward linearization is used in the attack on `SHAKE128` and cuts down the complexity by a factor of 2^{-6}. The details of the attacks can also be found in Supplementary Material A.

6.6 Summary of Preimage Attacks

We summarize our preimage attacks in Table 3. The preimage attack on 3-round `SHA3-512` is shown in the Supplementary Material B in our full version paper, the canonical representative states of the last round in the forward characteristic and the first round in backward characteristic are in CP-kernel. For 4-round and 5-round `SHA-3`, we use MILP to search for internal differential characteristic with more rounds. The canonical representative states of the last two rounds in the forward characteristic and the first two rounds in the backward characteristic are in the CP-kernel. Since ι brings about extra internal differences, we do not find the characteristics where the canonical representative states of the three consecutive rounds are all in CP-kernel. In order to make the forward and backward characteristic match in the capacity part, we first search for characteristics in one direction, and then continue to search in the other direction starting from the truncated internal difference of the characteristics at the matching point.

Target	n_r	k_2	k_3	k_4	k_5	k'_1	k'_2	k'_3	k'_4	Char.	Compl. ($\log_2$)
SHA3-512	3	7	3	-	-	4	64	-	-	5	504.2
SHAKE128	4	24	22	0	-	11	12	239	-	4	81.5
SHA3-224	4	24	22	0	-	11	12	239	-	4	135.5
SHA3-256/SHAKE256	4	24	22	0	-	11	12	239	-	4	151.5
SHA3-384	4	25	18	5	-	33	48	210	-	3	277.8
SHAKE128	5	155	34	15	11	7	4	98	399	1	100.5
SHA3-224	5	158	30	17.64	13	8	88	384	2	216.03	
SHA3-256/SHAKE256	5	158	30	17.64	13	8	88	384	2	254.33	

Table 3: The parameters of characteristics and complexities

7 Experiments

In order to illustrate the internal differential preimage attack, and to validate the new attack boundary, we implement an actual preimage attack on a reduced version of Keccak, called Keccak$[c = 704, r = 96, n_r = 4]$. The internal state size of Keccak$[704, 96, 4]$ is 800 bits, with the first 704 bits being the rate part and the remaining 96 bits the capacity part. The underlying permutation is reduced to 4 rounds, and the length of the digest is set to 704 bits.

In our preimage attack on this Keccak instance, the basic framework aligns with the analysis of 5-round SHAKE128 as shown in Section 6.1, by using 5-block messages for the preimage. Initially, we find a 3-round forward characteristic from round 2 to round 4 and a 3-round backward characteristic from round 1 to round 3 through MILP, see also Characteristic 6 in Supplementary Material G in the full version of this paper. On a desktop with an Intel Core i9-13900KF processor, it takes 1000 seconds to recover the preimage of a digest with all '1' with 16 threads, consuming 32MB of memory. The theoretical complexity needs to satisfy the product of the number of forward and backward messages to be $2^{66.5}$. In the experiment, 2^{35} and 2^{32} messages are calculated in the forward phase and backward phase respectively, and the product is 2^{67}, which matches the theoretical value. We provide the details of the attack as well as a concrete preimage in Supplementary Material C in our full version paper.

8 Conclusions

In this paper, we present preimage attacks on up to 5 rounds against all SHA-3 variants by introducing internal differential analysis. We combine the squeeze attack and MITM attack to develop squeeze meet-in-the-middle attack, and implement it with the internal differential. The new attack framework is divided into the forward phase, the backward phase and the collision phase. The TIDA algorithm is redesigned to construct the internal differential connector in the forward phase and the backward phase. In addition, we also develop a new linearization technique and apply it to the collecting message stage, which further reduces the time complexity. For 4-round SHAKE128, SHA3-224, SHA3-256, SHAKE256 and SHA3-384, our preimage attack outperforms the best known attacks. And the

first 5-round preimage attacks that are much faster than a brute-force search are presented for SHAKE128, SHA3-224, SHA3-256 and SHAKE256.

In addition, we have also tried to apply our attack to other sponge hash functions, such as Xoodyak [4] and Ascon [8]. We propose a preimage attack on 3-round Xoodyak with a time complexity of 2^{118}. The details of the attack are presented in Supplementary Material I. For ASCON, the situation becomes more complicated. To launch a squeezed meet-in-the-middle preimage attack, we face two challenges. The first one is to find a full state that corresponds to the digest. In our attack framework as depicted in Fig. 4, the Backward-TIDA starts from a full state H, but for ASCON a 128/256-bit digest is obtained by squeezing 64-bit state twice or four times due to its rate of 64. The second challenge is to effectively construct internal differential connectors or develop new properties like internal differentials. The rate of ASCON is small such that the degree of freedom is very limited after constructing internal differential connectors. We leave these challenges as open problems.

We stress that our attack does not threaten the security of the full SHA-3.

References

1. Bernstein, D.J.: Second preimages for 6 (7?(8??)) rounds of keccak. NIST mailing list (2010)
2. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Keccak sponge function family main document. Submission to NIST (Round 2) **3**(30), 320–337 (2009)
3. Chang, D., Kumar, A., Morawiecki, P., Sanadhya, S.K.: 1st and 2nd Preimage Attacks on 7, 8 and 9 Rounds of Keccak-224,256,384,512. In: SHA-3 workshop (2014)
4. Daemen, J., Hoffert, S., Peeters, M., Assche, G.V., Keer, R.V.: Xoodyak, a lightweight cryptographic scheme. IACR Trans. Symmetric Cryptol. **2020**(S1), 60–87 (2020). https://doi.org/10.13154/TOSC.V2020.IS1.60-87
5. Dinur, I.: Cryptanalytic applications of the polynomial method for solving multivariate equation systems over GF(2). In: Advances in cryptology – EUROCRYPT 2021. 40th annual international conference on the theory and applications of cryptographic techniques, Zagreb, Croatia, October 17–21, 2021. Proceedings. Part I, pp. 374–403. Cham: Springer (2021). https://doi.org/10.1007/978-3-030-77870-5_14
6. Dinur, I., Dunkelman, O., Shamir, A.: Collision attacks on up to 5 rounds of SHA-3 using generalized internal differentials. In: Moriai, S. (ed.) Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers. Lecture Notes in Computer Science, vol. 8424, pp. 219–240. Springer (2013), https://doi.org/10.1007/978-3-662-43933-3_12
7. Dinur, I., Dunkelman, O., Shamir, A.: Improved practical attacks on round-reduced keccak. J. Cryptol. **27**(2), 183–209 (2014). https://doi.org/10.1007/s00145-012-9142-5
8. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2: Lightweight authenticated encryption and hashing. J. Cryptol. **34**(3), 33 (2021). https://doi.org/10.1007/S00145-021-09398-9
9. Dworkin, M.J.: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions (2015). https://doi.org/10.6028/nist.fips.202

10. Guo, J., Liao, G., Liu, G., Liu, M., Qiao, K., Song, L.: Practical Collision Attacks against Round-Reduced SHA-3. J. Cryptol. **33**(1), 228–270 (2020). https://doi.org/10.1007/s00145-019-09313-3
11. Guo, J., Liu, G., Song, L., Tu, Y.: Exploring SAT for cryptanalysis: (quantum) collision attacks against 6-round SHA-3. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13793, pp. 645–674. Springer (2022), https://doi.org/10.1007/978-3-031-22969-5_22
12. Guo, J., Liu, M., Song, L.: Linear structures: Applications to cryptanalysis of round-reduced keccak. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10031, pp. 249–274 (2016), https://doi.org/10.1007/978-3-662-53887-6_9
13. Guo, J., Liu, M., Song, L.: Linear structures: applications to cryptanalysis of round-reduced Keccak. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 249–274. Springer (2016)
14. He, L., Lin, X., Yu, H.: Improved preimage attacks on 4-round keccak-224/256. IACR Trans. Symmetric Cryptol. **2021**(1), 217–238 (2021). https://doi.org/10.46586/TOSC.V2021.I1.217-238
15. Li, T., Sun, Y.: Preimage attacks on round-reduced Keccak-224/256 via an allocating approach. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 556–584. Springer (2019)
16. Li, T., Sun, Y., Liao, M., Wang, D.: Preimage attacks on the round-reduced keccak with cross-linear structures. IACR Trans. Symmetric Cryptol. **2017**(4), 39–57 (2017). https://doi.org/10.13154/TOSC.V2017.I4.39-57
17. Lin, X., He, L., Yu, H.: Improved preimage attacks on 3-round keccak-224/256. IACR Trans. Symmetric Cryptol. **2021**(3), 84–101 (2021). https://doi.org/10.46586/TOSC.V2021.I3.84-101
18. Liu, F., Isobe, T., Meier, W., Yang, Z.: Algebraic attacks on round-reduced keccak. In: Baek, J., Ruj, S. (eds.) Information Security and Privacy - 26th Australasian Conference, ACISP 2021, Virtual Event, December 1-3, 2021, Proceedings. Lecture Notes in Computer Science, vol. 13083, pp. 91–110. Springer (2021). https://doi.org/10.1007/978-3-030-90567-5_5, https://doi.org/10.1007/978-3-030-90567-5_5
19. Morawiecki, P., Pieprzyk, J., Srebrny, M.: Rotational Cryptanalysis of Round-Reduced Keccak. In: Moriai, S. (ed.) Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers. Lecture Notes in Computer Science, vol. 8424, pp. 241–262. Springer (2013). https://doi.org/10.1007/978-3-662-43933-3_{1}{3}, https://doi.org/10.1007/978-3-662-43933-3_13
20. National Institute of Standards and Technology: Module-Lattice-Based Digital Signature Standard (Aug 2024). https://doi.org/10.6028/NIST.FIPS.204
21. National Institute of Standards and Technology: Stateless Hash-Based Digital Signature Standard (Aug 2024). https://doi.org/10.6028/NIST.FIPS.205
22. Naya-Plasencia, M., Röck, A., Meier, W.: Practical analysis of reduced-round keccak. In: Bernstein, D.J., Chatterjee, S. (eds.) Progress in Cryptology - INDOCRYPT 2011 - 12th International Conference on Cryptology in India, Chennai, India, December 11-14, 2011. Proceedings. Lecture Notes in Computer

Science, vol. 7107, pp. 236–254. Springer (2011). https://doi.org/10.1007/978-3-642-25578-6_{1}{8}, https://doi.org/10.1007/978-3-642-25578-6_18
23. Peyrin, T.: Improved differential attacks for ECHO and grøstl. In: Rabin, T. (ed.) Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6223, pp. 370–392. Springer (2010), https://doi.org/10.1007/978-3-642-14623-7_20
24. Qiao, K., Song, L., Liu, M., Guo, J.: New collision attacks on round-reduced keccak. In: Coron, J., Nielsen, J.B. (eds.) 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. Lecture Notes in Computer Science, vol. 10212, pp. 216–243 (2017), https://doi.org/10.1007/978-3-319-56617-7_8
25. Qin, L., Hua, J., Dong, X., Yan, H., Wang, X.: Meet-in-the-middle preimage attacks on sponge-based hashing. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14007, pp. 158–188. Springer (2023). https://doi.org/10.1007/978-3-031-30634-1_6, https://doi.org/10.1007/978-3-031-30634-1_6
26. Rajasree, M.S.: Cryptanalysis of round-reduced KECCAK using non-linear structures. In: Hao, F., Ruj, S., Gupta, S.S. (eds.) Progress in Cryptology - INDOCRYPT 2019 - 20th International Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11898, pp. 175–192. Springer (2019). https://doi.org/10.1007/978-3-030-35423-7_9, https://doi.org/10.1007/978-3-030-35423-7_9
27. Sasaki, Y.: Memoryless unbalanced meet-in-the-middle attacks: impossible results and applications. In: Applied cryptography and network security. 12th international conference, ACNS 2014, Lausanne, Switzerland, June 10–13, 2014. Proceedings, pp. 253–270. Berlin: Springer (2014). https://doi.org/10.1007/978-3-319-07536-5_16
28. Song, L., Liao, G., Guo, J.: Non-full Sbox Linearization: Applications to Collision Attacks on Round-Reduced Keccak. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10402, pp. 428–451. Springer (2017), https://doi.org/10.1007/978-3-319-63715-0_15
29. Zhang, Z., Hou, C., Liu, M.: Collision attacks on round-reduced SHA-3 using conditional internal differentials. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14007, pp. 220–251. Springer (2023), https://doi.org/10.1007/978-3-031-30634-1_8
30. Zhang, Z., Hou, C., Liu, M.: Probabilistic linearization: Internal differential collisions in up to 6 rounds of SHA-3. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14923, pp. 241–272. Springer (2024). https://doi.org/10.1007/978-3-031-68385-5_8, https://doi.org/10.1007/978-3-031-68385-5_8

Supplementary Material

A Preimage Attacks on 4-round SHAKE128 and SHA3-224

For 4-round `SHAKE128` and `SHA3-224`, we use the same characteristic used to attack 4-round `SHA3-256`/`SHAKE256`.

SHAKE128. The scales of the sets D'_L and D'_R satisfy $2^{n_1+n_2} = 2^{128}$.

Forward Phase. During the Forward-TIDA stage, the input difference system E_Δ contains 685 equations, of which the inner part contains 73 equations. The output probability $p_1^\star$ of Procedure PIDS is 1. Therefore, the complexity of TIDA is 2^{73}. The degree of freedom of the message space W_1 output by Algorithm 1 is at least $r/2 - 345 = 327$. The complexity $T_{A,1}$ of Forward-TIDA stage is $2^{73} \cdot \max\left(2^{n_1+k_2+k_3+\overline{k_4}-327}, 1\right) = 2^{73}$.

In the collecting messages stage, each linearization is equivalent to solving a linear system with 327 variables and $45 \times 3 + k_2 = 159$ equations, with complexity of $327^\omega = (2^{23.47})^\dagger$. The number of linearizations is $\max\left(2^{n_1+k_2+k_3+\overline{k_4}-(r/2-480)}, 1\right) = 1$. $T_{B,1} = (2^{23.47})^\dagger, T_{C,1} = 2^{n_1+k_3+\overline{k_4}} = 2^{n_1+22}$.

Backward Phase. For `SHAKE128`, the last 4 lanes of the state are the capacity part. For the subspace V composed of β'_3, the dimension is $k_3 = 239$, and its projection dimension on the last plane is 151. Therefore, in each run of Backward-TIDA, an average of 2^9 the fifth blocks M_4 needs to be randomly selected, and the complexity is 2^9. After obtaining the target difference β'_3, take the subspace W_3 in $[i, \beta'_3]$ as the message space. The degrees of freedom of W_3 output by Algorithm 2 is at least $20 \times 32 - 239 = 401$. The complexity $T_{A,2}$ of this stage is $2^9 \cdot \max\left(2^{n_2+k'_1+k'_2+k'_3-401}, 1\right) = 2^9$.

In the collecting messages stage, there are 55 non-active Sboxes, $(N_{2,k=2}, N_{3,k=2}, N_{3,k=3}, N_{3,k=4}) = (6, 76, 29, 0)$. Each linearization is equivalent to solving a linear system with 401 variables and $55 \times 3 + N_{3,k=2} + k'_2 + N_{2,k=2} = 259$ equations, with complexity of $401^\omega = (2^{24.30})^\dagger$. Based on the analysis in Section 5.3, we add an inverse differential transition condition to each active Sbox in the second round, resulting in an additional consumption of $N_{2,k=2} = 6$ degrees of freedom, so the number of linearizations is $\max\left(2^{n_2+k'_1+k'_2+6-(20\times32-480)}, 1\right) = 1$. $T_{B,2} = (2^{24.30})^\dagger, T_{C,2} = 2^{n_2+k'_1} = 2^{n_2+11}$.

Complexity. The total complexity The total complexity $T = T_{A,1} + T_{B,1} + T_{C,1} + T_{A,2} + T_{B,2} + T_{C,2} = 2^{73} + (2^{23.47})^\dagger + 2^{n_1+22} + 2^9 + (2^{24.30})^\dagger + 2^{n_2+11}$. We take $(n_1, n_2) = (58.5, 69.5)$, in which case T takes the minimum value $2^{81.5}$.

SHA3-224. The scales of the sets D'_L and D'_R satisfy $2^{n_1+n_2} = 2^{224}$.

Forward Phase. During the Forward-TIDA stage, the input difference system E_Δ contains 685 equations, of which the inner part contains 109 equations. The output probability $p_1^\star$ of Procedure PIDS is 1. Therefore, the complexity of TIDA is 2^{109}. The degree of freedom of the message space W_1 output by Algorithm 1 is at least $r/2-345 = 231$. The complexity $T_{A,1}$ of Forward-TIDA stage is $2^{109} \cdot \max(2^{n_1+k_2+k_3+\overline{k_4}-231}, 1) = \max(2^{n_1-76}, 2^{109})$.

In the collecting messages stage, each linearization is equivalent to solving a linear system with 231 variables and $45 \times 3 + k_2 = 159$ equations, with complexity of $231^\omega = (2^{22.06})^\dagger$. The number of linearizations is $\max(2^{n_1+k_2+k_3+\overline{k_4}-(r/2-480)}, 1) = \max(2^{n_1-50}, 1)$. $T_{B,1} = (\max(2^{n_1-27.94}, 2^{22.06}))^\dagger, T_{C,1} = 2^{n_1+k_3+\overline{k_4}} = 2^{n_1+22}$.

Backward Phase. For `SHA3-224`, the last 7 lanes of the state are the capacity part. For the subspace V composed of β'_3, the dimension is $k_3 = 239$, and its projection dimension on the last two planes is 201. We implement Backward-TIDA using the method in Section 6.6. In each run of Backward-TIDA, an average of 2^{109} the fifth blocks M_4 needs to be randomly selected, and the complexity is 2^{109}. The degrees of freedom of W_3 output by Algorithm 2 is at least $15 \times 32 - 284 = 196$. No linearization is performed in this phase. The number of Backward-TIDA runs is $\max(2^{n_2+k'_1+k'_2-196}, 1) = \max(2^{n_2-173}, 1)$. $T_{A,2} = 2^{109} \cdot \max(2^{n_2-173}, 1) = \max(2^{n_2-64}, 2^{109}), T_{C,2} = 2^{n_2+k'_1+k'_2} = 2^{n_2+23}$.

Complexity. The total complexity $T = T_{A,1} + T_{B,1} + T_{C,1} + T_{A,2} + T_{C,2} = \max(2^{n_1-76}, 2^{109}) + (\max(2^{n_1-27.94}, 2^{22.06}))^\dagger + 2^{n_1+22} + \max(2^{n_2-64}, 2^{109}) + 2^{n_2+23}$. We take $(n_1, n_2) = (112.5, 111.5)$, T takes the minimum value $2^{135.5}$.

B A Preimage Attack on 3-round SHA3-512

For 3-round `SHA3-512`, we use the internal differential characteristic given in Characteristic 5 in Supplementary Material G. The transition condition numbers are $(k_2, \overline{k_3}, k'_1, k'_2) = (7, 3, 4, 64)$. The scales of the sets D'_L and D'_R in the collision phase satisfy $2^{n_1+n_2} = 2^{512}$.

Forward Phase. For $\alpha_1^\star$, there are 65 active Sboxes and 95 non-active Sboxes. During the Forward-TIDA stage, each non-active Sbox provides 5 linear equations, 39 active Sboxes do not provide linear equations. The other 26 active Sboxes provide 1 linear equation, and the corresponding difference density changes from 9/32 to 9/16. The remaining active Sboxes do not add equations. The input difference system E_Δ contains 501 equations, of which the inner part contains 230 equations. The output probability $p_1^\star$ of Procedure PIDS is $2^{83.24}$. Therefore, the complexity of TIDA is $2^{230+83.24} = 2^{313.24}$. Since the expected transition condition number $\mathtt{E}(k_1)$ of the first round is 225, the average degree of freedom of the message space W_1 output by Algorithm 1 is at least $r/2 - 225 = 63$. $T_{A,1} = 2^{313.24} \cdot \max(2^{n_1+k_2+\overline{k_3}-63}, 1) = \max(2^{n_1+260.24}, 2^{313.24}), T_{C,1} = 2^{n_1+k_2+\overline{k_3}} = 2^{n_1+11}$.

Backward Phase. For `SHA3-512`, the last 16 lanes of the state are the capacity part, in the second plane, the last lane is constant. In the Backward-TIDA

stage, we consider the two Sboxes that generate the differential on the second plane. For the subspace V composed of β_2', the dimension is $k_2 = 64$, and its projection dimension on the last three planes is 64. We follow the steps in Algorithm 2 to first find $\beta_2' \in V$ that matches the last three planes of $\Delta(M)$, and then check whether β_2' satisfies $\texttt{VDDT} > 0$ for each Sbox in the second plane. For each 2-bit value (y, y'), 28 of the 32 input difference δ_{in} satisfy $\texttt{VDDT}(\delta_{in}, y, y') > 0$, and the minimum non-value is 8. Therefore, the complexity of running Backward-TIDA once is $(28/32)^{32} \cdot 2^{15\times32-64} = 2^{422.16}$. The degree of freedom of the message space W_3 output by Algorithm 2 is at least $8 \times 32 - 64 = 192$. The number of TIDA runs is $\max(2^{n_2+k_1'-192}, 1) = \max(2^{n_2-188}, 1)$. $T_{A,2} = 2^{422.16} \cdot \max(2^{n_2-188}, 1) = \max(2^{n_2+234.16}, 2^{422.16}), T_{C,2} = 2^{n_2+k_1'} = 2^{n_2+4}$.

Complexity. The total complexity $T = T_{A,1} + T_{C,1} + T_{A,2} + T_{C,2} = \max(2^{n_1+260.24}, 2^{313.24}) + 2^{n_1+11} + \max(2^{n_2+234.16}, 2^{422.16}) + 2^{n_2+4}$. We take $(n_1, n_2) = (242.96, 269.04)$, in which case T takes the minimum value $2^{504.2}$.

C A Preimage Attack on Keccak$[r = 704, c = 96, n_r = 4]$

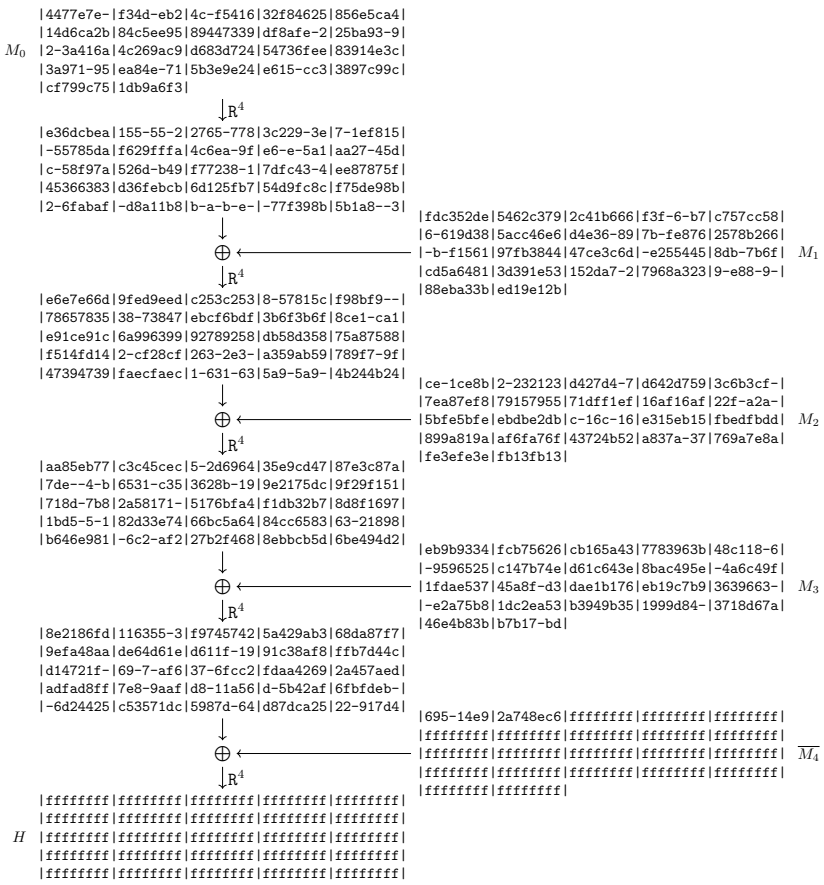
For Keccak$[r = 704, c = 96, n_r = 4]$, we use the internal differential characteristic given in Characteristic 6 in Supplementary Material G. The parameters are $(k_2, \overline{k_3}, \overline{k_4}, k_1', k_2', k_3') = (27, 12.5, 0, 6, 8, 128)$, and there are 65 active Sboxes and 15 non-active Sboxes for $\alpha_1^\star$. The scales of the sets D_L' and D_R' in the collision phase satisfy $2^{n_1+n_2} = 2^{48}$.

In the forward phase, we set 4 linear equations for each active Sbox and 5 linear equations for each non-active Sbox, and obtain $65 \times 4 + 15 \times 5 = 335$ equations in input difference system. By applying the Procedure PIDS, the probability $p_1^\star = 1$. There are only a few equations in the inner part of the system E_Δ, so the complexity of Forward-TIDA can be ignored. Since the initial degrees of freedom are 352, the first round of linearization does not need to be performed multiple times. After one linearization, the remaining degrees of freedom are $352 - 240 = 112$. Then add the linear conditions in the differential transition system $E_{\beta_1 \to \alpha_2^\star}$, which consumes 27 degrees of freedom, and we can get a message subspace W_2 with a dimension of 85. We randomly select 2^{35} messages in W_2, and $2^{22.5}$ of them enter the set D_L'.

In the backward phase, we directly fix the target difference β_3'. Since the last three lanes of the state are capacity parts and have sufficient degrees of freedom, we gave up the degrees of freedom of the last plane. In Backward-TIDA stage, M_4 is randomly selected until each Sbox in the last plane can be matched by modifying the value of M_3. In the experiment, we randomly select about 2^{19} the fifth blocks M_4 to find a fourth block M_3 that allowed the message to propagate to β_3' on the last plane. At this time, M_3 and the states in $[i, \beta_3']$ have a degree of freedom of $20 \times 16 = 320$. Then we impose 240 linearization conditions and 12 inverse differential transition conditions on the state in $[i, \beta_3']$ to propagate from $\alpha_3'^\star$ to β_3', and obtain a subspace W_4 with $320 - 240 - 12 = 68$ degrees of freedom. We randomly select 2^{32} messages from W_4, and 2^{26} of them enter the set D_R'.

In the collision phase, we store the states in set D'_L into hash table H_T, and perform an exhaustive search on set D'_R, and finally find a collision in the capacity part. Then modify the values of M_2 and M_3 to complete the entire preimage attack. The theoretical complexity needs to satisfy the product of the number of forward and backward messages to be $2^{(48+\overline{k_3}+\overline{k_4}+k'_1)/2} = 2^{66.5}$.

Below we give a preimage of the digest with all '1' bits for KECCAK$[r = 704, c = 96, n_r = 4]$.



Preimage: A preimage $(M_0||M_1||M_2||M_3||\overline{M_4})$ for the digest H with all '1' bits.

D Sbox Linearization

In standard differential attack, Qiao *et al.* [24] chose some available subsets with special properties to achieve fast enumerations. A similar approach can be applied to internal differentials in order to find initial messages conforming 3-round internal differential characteristic with a greater probability. It is obvious to note the Sbox is non-linear when the entire 2^5 input space is considered. In case of `SHAKE128` and `SHAKE256`, we are to choose affine subspaces of size up to 4 which are linear with respective to the Sbox. Note that χ is the only nonlinear part of the KECCAK round function. Hence, the first round function becomes linear when the initial messages are restricted to such a subspace. Formally, we consider the following definition.

Definition 9 (Linearizable affine subspace [24]). *Linearizable affine subspaces are affine input subspaces on which Sbox substitution is equivalent to a linear transformation. If V is a linearizable affine subspaces of an Sbox operation $\mathtt{S}(\cdot)$, $\forall x \in V$, $\mathtt{S}(x) = A \cdot x + b$, where A is a matrix and b is a constant vector.*

For example, when input $x = (x_0, 0, x_2, 0, 1)$, the expression of the Sbox can be re-written as linear transformation:

$$\mathtt{S}(x) = \begin{pmatrix} 1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0 \end{pmatrix} \cdot x \oplus \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix},$$

where x are bit vector of input values of the KECCAK Sbox. Exhaustive search for the linearizable affine subspaces of the KECCAK Sbox shows:

Observation 2 ([24]) *Given $\delta_{in}, \delta_{out} \in \mathbb{F}_2^5$, denote the value solution set $V = \{x : \mathtt{S}(x) + \mathtt{S}(x + \delta_{in}) = \delta_{out}\}$ and $\mathtt{S}(V) = \{\mathtt{S}(x) : x \in V\}$, we have*

a. there does not exit any linearizable affine subspace with dimension 3 or more.
b. if $\mathtt{DDT}(\delta_{in}, \delta_{out}) = 2$ or 4, the V is a linearizable affine subspace.
c. if $\mathtt{DDT}(\delta_{in}, \delta_{out}) = 8$, then there are six 2-dimensional subspaces $W_i \subset V$, $i = 0, 1, ..., 5$ such that $W_i (i = 0, 1, ..., 5)$ are linearizable affine subspaces.

Based on the above observations, given the input internal difference and the output internal difference of the first round, we can first add the restriction of differential transition conditions in the initial message space. Then add 3 linear conditions to each inactive Sbox and add one condition to each active Sbox with 2 differential transition conditions until the whole KECCAK state is the direct product of all linearizable affine subspaces. Finally, the conditional internal difference technique is applied to the nonlinear layers of the second and third rounds to find messages conforming the internal differential characteristics of the first three rounds. The full linearization consists of the following steps:

1. For each active Sbox in the first round, add corresponding differential transition conditions to the initial message space to construct the differential transition system $E_{\beta\to\alpha}$. If the number of differential transition conditions is 2, an additional linear condition is added to limit the input to a linearizable affine subspace. And modifying the constants of each condition to maintain $E_{\beta\to\alpha}$ being consistent.
2. For each non-active Sbox, add 3 linear conditions to system $E_{\beta\to\alpha}$ that restrict its linearizable affine subspace. And modifying the constants of each condition to maintain $E_{\beta\to\alpha}$ being consistent.
3. Solve system $E_{\beta\to\alpha}$ to obtain its solution space V, and consider $\mathtt{R}(V)$ as the new initial message space. Utilize conditional internal differential techniques to select states passing the first three round functions from subspace $\mathtt{R}(V)$.

E Procedure PIDS

Procedure PIDS($\alpha_1^\star$, MDST)

Input: Internal difference $\alpha_1^\star$ and MDST.
Output: Probabilistic input difference system $E_\Delta(\Delta_\mathcal{R}, \Delta_\mathcal{C})$, probability $p_1^\star$.

Set $E_\Delta = \emptyset$, $p_1^\star = 1$ and $W = L(\Delta_\mathcal{R}, \Delta_\mathcal{C})$.
/* W is a variable vector $(w_0, ..., w_{799})$, $w_i = w_i(\Delta_\mathcal{R}, \Delta_\mathcal{C})$ is a linear function about $\Delta_\mathcal{R} = (\delta_0, ..., \delta_{799-c/2}), \Delta_\mathcal{C} = (\delta_{800-c/2}, ..., \delta_{799})$ */
for $j = 0 \to 160$ **do**
    Get the output difference δ_{out} of the j-th Sbox from $\alpha_1^\star$.
    if $\delta_{out} = 0$ **then**
        $E_\Delta = E_\Delta \cup \{w_{5j}(\Delta_\mathcal{R}, \Delta_\mathcal{C}) = 0, \ldots, w_{5j+4}(\Delta_\mathcal{R}, \Delta_\mathcal{C}) = 0\}$.
    else
        Select one t-dimensional affine subspace $U \subset \mathbb{F}_2^5$.
        Compute the corresponding difference density $\mathrm{P}(U, \delta_{out})$ and $5-t$ linear equations $\{l_0^{(k)} \cdot w_{5j} + \cdots + l_4^{(k)} \cdot w_{5j+5} = q^{(k)}\}_{k=1}^{5-t}$.
        $E_\Delta = E_\Delta \cup \{l_0^{(k)} \cdot w_{5j}(\Delta_\mathcal{R}, \Delta_\mathcal{C}) + \cdots + l_4^{(k)} \cdot w_{5j+4}(\Delta_\mathcal{R}, \Delta_\mathcal{C}) = q^{(k)}\}_{k=1}^{5-t}$.
        $p_1^\star = p_1^\star \cdot \mathrm{P}(U, \delta_{out})$.
    end
end
return $(E_\Delta, p_1^\star)$

F Inverse-Linearizable Affine Subspaces

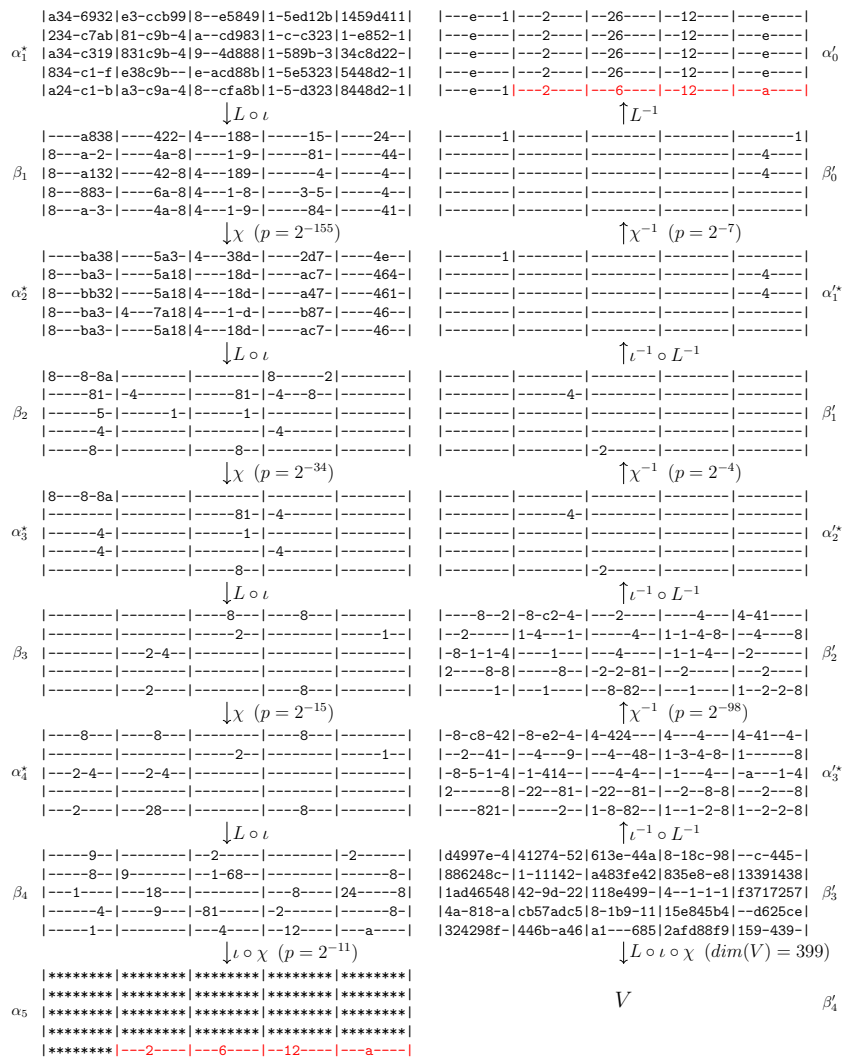
The 80 2-dimensional linearizable affine subspaces are listed in Table 4.

$\{00, 09, 05, 0c\}$	$\{12, 0b, 16, 0f\}$	$\{00, 09, 0a, 03\}$	$\{05, 0c, 0a, 03\}$	$\{00, 12, 0a, 18\}$
$\{09, 12, 03, 18\}$	$\{00, 0b, 0a, 01\}$	$\{09, 0b, 03, 01\}$	$\{12, 0b, 18, 01\}$	$\{16, 0f, 18, 01\}$
$\{00, 09, 0d, 04\}$	$\{05, 0c, 0d, 04\}$	$\{0a, 03, 0d, 04\}$	$\{05, 16, 0d, 1e\}$	$\{0c, 16, 04, 1e\}$
$\{05, 0f, 0d, 07\}$	$\{0c, 0f, 04, 07\}$	$\{12, 0b, 1e, 07\}$	$\{16, 0f, 1e, 07\}$	$\{18, 01, 1e, 07\}$
$\{00, 12, 14, 06\}$	$\{0a, 18, 14, 06\}$	$\{09, 0b, 15, 17\}$	$\{03, 01, 15, 17\}$	$\{00, 05, 14, 11\}$
$\{09, 0c, 14, 11\}$	$\{12, 05, 06, 11\}$	$\{00, 05, 15, 10\}$	$\{09, 0c, 15, 10\}$	$\{0b, 0c, 17, 10\}$
$\{14, 15, 11, 10\}$	$\{00, 16, 14, 02\}$	$\{12, 16, 06, 02\}$	$\{0b, 0f, 06, 02\}$	$\{05, 16, 11, 02\}$
$\{0d, 1e, 11, 02\}$	$\{09, 0f, 15, 13\}$	$\{12, 16, 17, 13\}$	$\{0b, 0f, 17, 13\}$	$\{0c, 0f, 10, 13\}$
$\{04, 07, 10, 13\}$	$\{06, 17, 02, 13\}$	$\{14, 15, 1a, 1b\}$	$\{11, 10, 1a, 1b\}$	$\{00, 12, 1a, 08\}$
$\{0a, 18, 1a, 08\}$	$\{14, 06, 1a, 08\}$	$\{15, 06, 1b, 08\}$	$\{14, 17, 1a, 19\}$	$\{09, 0b, 1b, 19\}$
$\{03, 01, 1b, 19\}$	$\{15, 17, 1b, 19\}$	$\{06, 17, 08, 19\}$	$\{02, 13, 08, 19\}$	$\{0a, 0d, 1a, 1d\}$
$\{03, 04, 1a, 1d\}$	$\{18, 0d, 08, 1d\}$	$\{0a, 0d, 1b, 1c\}$	$\{03, 04, 1b, 1c\}$	$\{01, 04, 19, 1c\}$
$\{14, 15, 1d, 1c\}$	$\{11, 10, 1d, 1c\}$	$\{1a, 1b, 1d, 1c\}$	$\{0a, 1e, 1a, 0e\}$	$\{18, 1e, 08, 0e\}$
$\{01, 07, 08, 0e\}$	$\{05, 16, 1d, 0e\}$	$\{0d, 1e, 1d, 0e\}$	$\{11, 02, 1d, 0e\}$	$\{10, 02, 1c, 0e\}$
$\{03, 07, 1b, 1f\}$	$\{18, 1e, 19, 1f\}$	$\{01, 07, 19, 1f\}$	$\{11, 13, 1d, 1f\}$	$\{0c, 0f, 1c, 1f\}$
$\{04, 07, 1c, 1f\}$	$\{10, 13, 1c, 1f\}$	$\{06, 17, 0e, 1f\}$	$\{02, 13, 0e, 1f\}$	$\{08, 19, 0e, 1f\}$

Table 4: The 2-dimensional inverse-linearizable affine subspaces of KECCAK Sbox

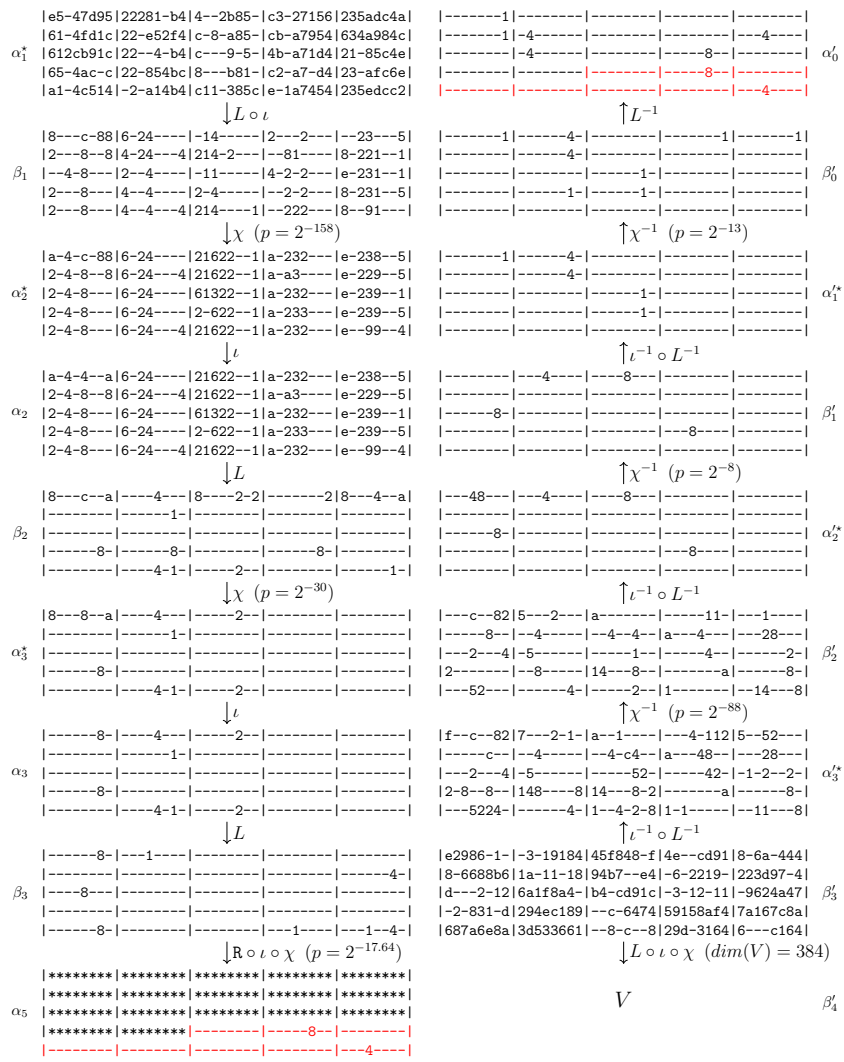
G Internal Differential Characteristics for the Attacks

The internal difference $[i, v]$ is represented by its canonical representative state defined in Section 4.2. Each state is given as a matrix of 5×5 lanes of 64 bits, order from left to right, where each lane is given in hexadecimal using the little-endian format. The symbol '-' is used in order to denote a zero 4-bit value.



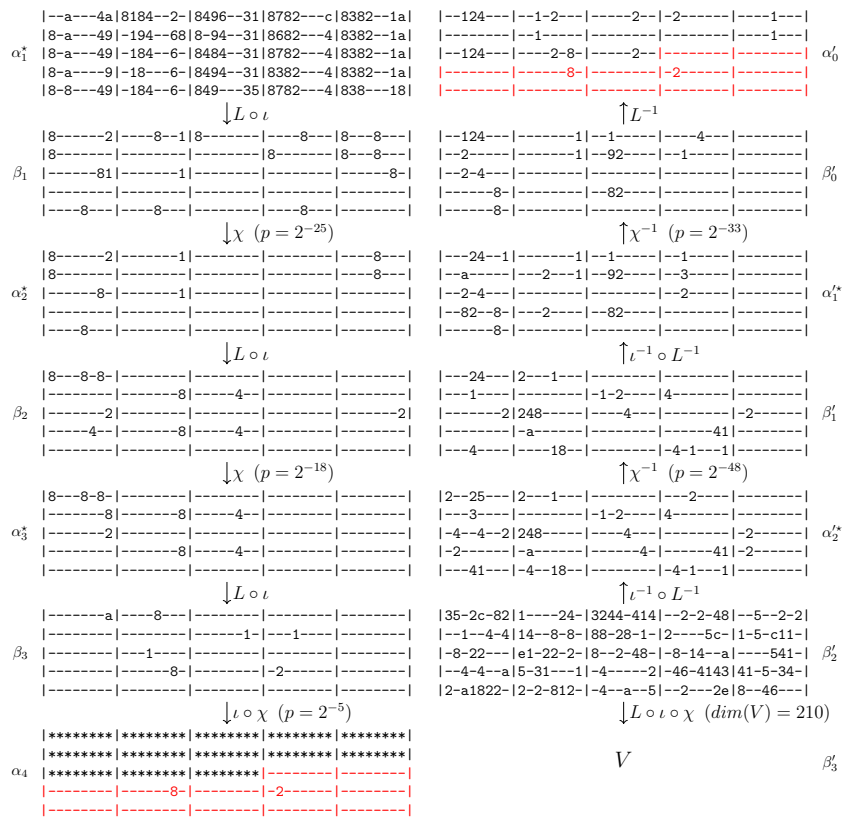
The forward characteristic shown on the left and backward characteristic shown on the right have a period of $i = 32$ for the 5-round attack on SHAKE128, as described in Section 6.1.

Characteristic 1: The internal differential characteristic with parameters $(k_2, k_3, k_4, \overline{k_5}, k_1', k_2', k_3', k_4') = (155, 34, 15, 11, 7, 4, 98, 399)$.



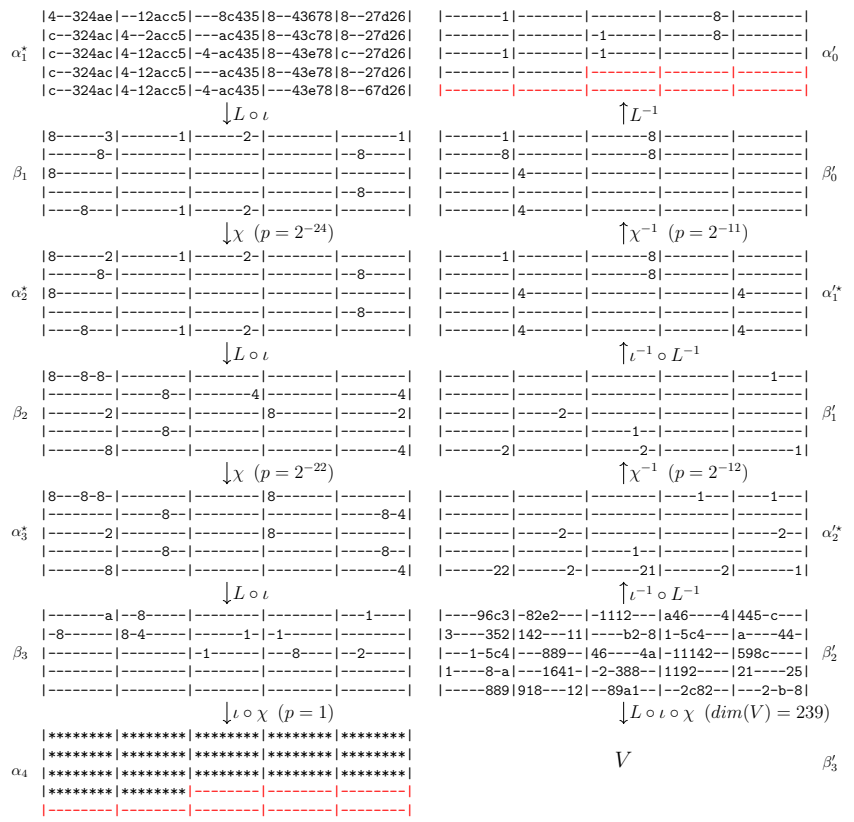
The forward characteristic shown on the left and backward characteristic shown on the right have a period of $i = 32$ for the 5-round attack on `SHA3-224`, `SHA3-256` and `SHAKE256`, as described in Section 6.2 and Section 6.3.

Characteristic 2: The internal differential characteristic with parameters $(k_2, k_3, \overline{k_{4,5}}, k_1', k_2'.k_3', k_4') = (158, 30, 17.64, 13, 8, 88, 384)$.



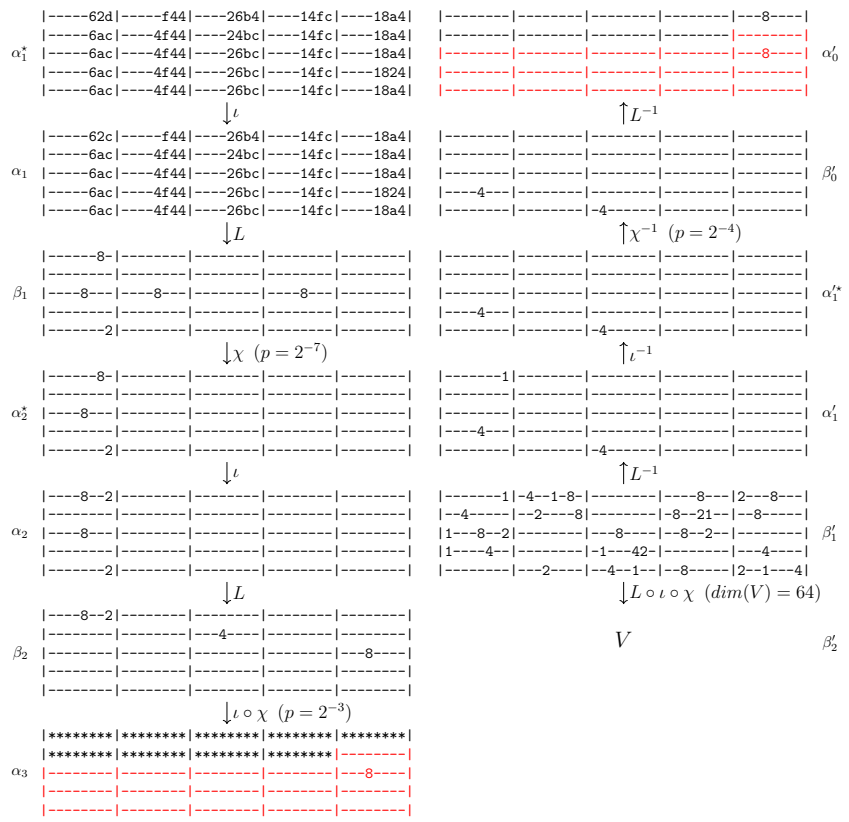
The forward characteristic shown on the left and backward characteristic shown on the right have a period of $i = 32$ for the 4-round attack on `SHA3-384`, as described in Section 6.4.

Characteristic 3: The internal differential characteristic with parameters $(k_2, k_3, \overline{k_4}, k_1', k_2', k_3') = (25, 18, 5, 33, 48, 210)$.



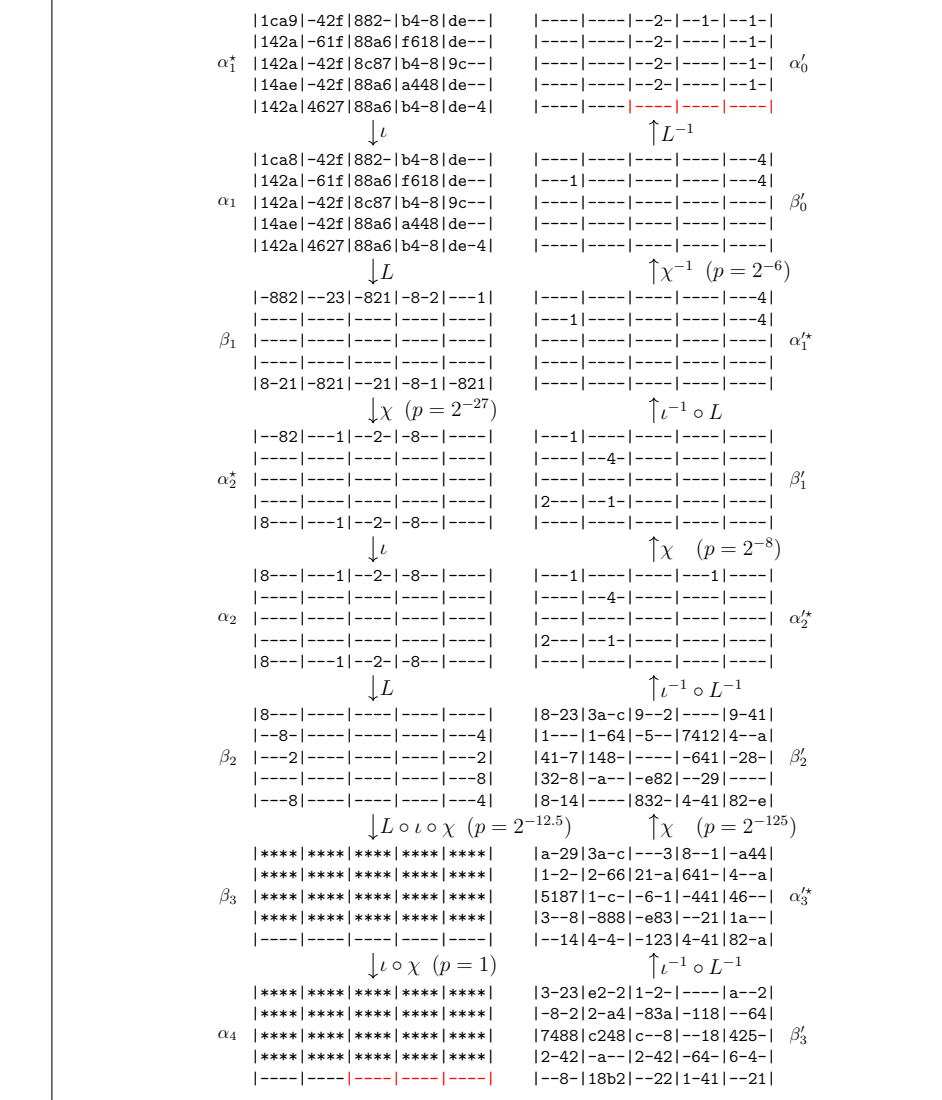
The forward characteristic shown on the left and backward characteristic shown on the right have a period of $i = 32$ for the 4-round attack on `SHAKE128`, `SHA3-224`, `SHA3-256` and `SHAKE256`, as described in Section 6.4.

Characteristic 4: The internal differential characteristic with parameters $(k_2, k_3, \overline{k_4}, k_1', k_2', k_3') = (24, 22, 0, 11, 12, 239)$.



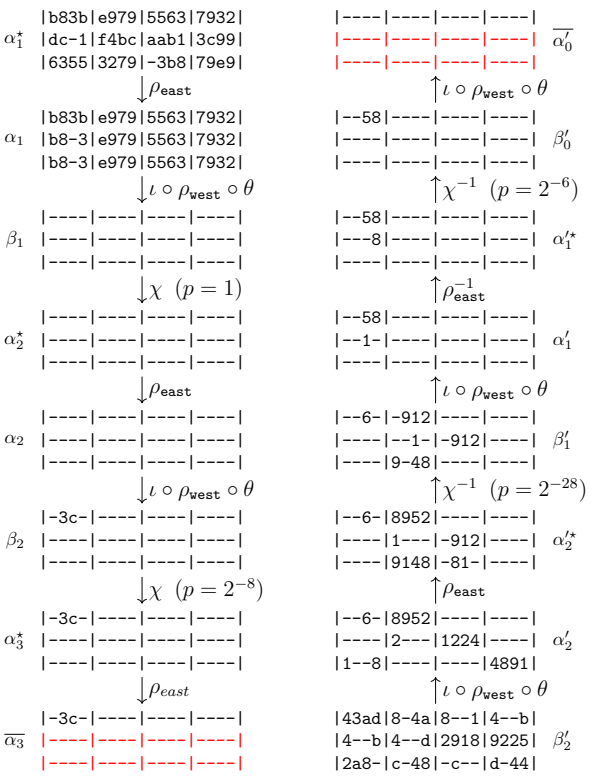
The forward characteristic shown on the left and backward characteristic shown on the right have a period of $i = 32$ for the 3-round attack on `SHA3-512`, as described in Section 6.4.

Characteristic 5: The internal differential characteristic with parameters $(k_2, \overline{k_3}, k_1', k_2') = (7, 3, 4, 64)$.



The forward characteristic shown on the left and backward characteristic shown on the right have a period of $i = 16$ for the attack on KECCAK$[r = 704, c = 96, n_r = 4]$, as described in Section 7.

Characteristic 6: The internal differential characteristic with parameters $(k_2, \overline{k_3}, \overline{k_4}, k_1', k_2', k_3') = (27, 12.5, 0, 6, 8, 125)$.



The characteristic has a period of $i = 16$ for the 3-round preimage attack on Xoodyak-XOF with a 128-bit digest, as described in Section I.3.

Characteristic: The internal differential characteristic with parameters $(k_1, k_2, k_3, k_1', k_2', k_3') = (116, 0, 8, 6, 28, 72)$.

H Value-Difference Distribution Table of KECCAK Sbox

Here we list the value-difference distribution table corresponding to 2-bit value, 4-bit value and 6-bit value (y, y'). $\texttt{VDDT}(\delta_{in}, y, y') = \#\{(x, x') \in \mathbb{F}_2^{2\times(5-t)} | \mathtt{S}^{-1}(x||y) + \mathtt{S}^{-1}(x'||y') = \delta_{in}\}$.

(y_4, y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$
$(0,0)$	00	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	01	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	02	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	03	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
	04	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	05	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	06	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	07	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
	08	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	09	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	0a	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	0b	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
	0c	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	0d	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	0e	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	0f	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
	11	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	12	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	13	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
	15	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	16	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	17	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
	19	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	1a	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	1b	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
	1d	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	1e	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	1f	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
$(1,1)$	00	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	01	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	02	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	03	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$
	04	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	05	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	06	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	07	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$
	08	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	09	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	0a	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	0b	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$
	0c	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	0d	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	0e	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	0f	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$
	11	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	12	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	13	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$
	15	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	16	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	17	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$
	19	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	1a	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	1b	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$
	1d	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	1e	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	1f	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$

Table 5: 2-bit Value-Difference Distribution Table of KECCAK Sbox

(y_4, y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$
$(0,1)$	01	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	02	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	03	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
	05	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	06	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	07	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
	09	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	0a	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	0b	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
	0d	8	$\{0b, 16, 12, 0f, 1a, 03, 1e, 07\}$
	0e	8	$\{00, 16, 04, 12, 1a, 08, 0c, 1e\}$
	0f	8	$\{16, 09, 0d, 12, 1a, 01, 05, 1e\}$
	10	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	11	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	12	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	13	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
	14	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	15	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	16	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	17	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
	18	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	19	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	1a	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	1b	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
	1c	16	$\{00, 0b, 16, 09, 0d, 04, 12, 0f, 1a, 01, 08, 03, 05, 0c, 1e, 07\}$
	1d	8	$\{00, 09, 0d, 04, 01, 08, 05, 0c\}$
	1e	8	$\{0b, 09, 0d, 0f, 01, 03, 05, 07\}$
	1f	8	$\{00, 0b, 04, 0f, 08, 03, 0c, 07\}$
$(1,0)$	01	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	02	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	03	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$
	05	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	06	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	07	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$
	09	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	0a	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	0b	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$
	0d	8	$\{02, 17, 06, 13, 0a, 1b, 0e, 1f\}$
	0e	8	$\{14, 02, 10, 06, 0a, 18, 1c, 0e\}$
	0f	8	$\{15, 02, 11, 06, 0a, 19, 1d, 0e\}$
	10	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	11	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	12	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	13	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$
	14	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	15	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	16	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	17	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$
	18	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	19	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	1a	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	1b	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$
	1c	16	$\{15, 14, 02, 17, 10, 11, 06, 13, 0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	1d	8	$\{15, 14, 10, 11, 18, 19, 1d, 1c\}$
	1e	8	$\{15, 17, 11, 13, 1b, 19, 1d, 1f\}$
	1f	8	$\{14, 17, 10, 13, 1b, 18, 1c, 1f\}$

Table 6: 2-bit Value-Difference Distribution Table of Keccak Sbox

(y_3, y_4, y_3', y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$	(y_3, y_4, y_3', y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$
$(0,0,0,0)$	00	8	$\{00, 0b, 16, 09, 0d, 04, 12, 0f\}$	$(1,1,1,1)$	00	8	$\{0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	02	4	$\{0b, 09, 0d, 0f\}$		01	4	$\{18, 19, 1d, 1c\}$
	04	8	$\{00, 0b, 16, 09, 0d, 04, 12, 0f\}$		02	4	$\{1b, 19, 1d, 1f\}$
	06	4	$\{0b, 09, 0d, 0f\}$		03	4	$\{1b, 18, 1c, 1f\}$
	09	4	$\{00, 09, 0d, 04\}$		04	8	$\{0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	0b	4	$\{00, 0b, 04, 0f\}$		05	4	$\{18, 19, 1d, 1c\}$
	0d	4	$\{00, 09, 0d, 04\}$		06	4	$\{1b, 19, 1d, 1f\}$
	0f	4	$\{00, 0b, 04, 0f\}$		07	4	$\{1b, 18, 1c, 1f\}$
	12	4	$\{00, 16, 04, 12\}$		11	4	$\{0a, 1b, 0e, 1f\}$
	16	4	$\{00, 16, 04, 12\}$		12	4	$\{0a, 18, 1c, 0e\}$
	19	4	$\{0b, 16, 12, 0f\}$		13	4	$\{0a, 19, 1d, 0e\}$
	1b	4	$\{16, 09, 0d, 12\}$		15	4	$\{0a, 1b, 0e, 1f\}$
	1d	4	$\{0b, 16, 12, 0f\}$		16	4	$\{0a, 18, 1c, 0e\}$
	1f	4	$\{16, 09, 0d, 12\}$		17	4	$\{0a, 19, 1d, 0e\}$
$(1,0,0,0)$	01	4	$\{01, 08, 05, 0c\}$	$(0,1,1,1)$	08	8	$\{15, 14, 02, 17, 10, 11, 06, 13\}$
	03	4	$\{08, 03, 0c, 07\}$		09	4	$\{15, 14, 10, 11\}$
	05	4	$\{01, 08, 05, 0c\}$		0a	4	$\{15, 17, 11, 13\}$
	07	4	$\{08, 03, 0c, 07\}$		0b	4	$\{14, 17, 10, 13\}$
	08	8	$\{1a, 01, 08, 03, 05, 0c, 1e, 07\}$		0c	8	$\{15, 14, 02, 17, 10, 11, 06, 13\}$
	0a	4	$\{01, 03, 05, 07\}$		0d	4	$\{15, 14, 10, 11\}$
	0c	8	$\{1a, 01, 08, 03, 05, 0c, 1e, 07\}$		0e	4	$\{15, 17, 11, 13\}$
	0e	4	$\{01, 03, 05, 07\}$		0f	4	$\{14, 17, 10, 13\}$
	11	4	$\{1a, 03, 1e, 07\}$		19	4	$\{02, 17, 06, 13\}$
	13	4	$\{1a, 01, 05, 1e\}$		1a	4	$\{14, 02, 10, 06\}$
	15	4	$\{1a, 03, 1e, 07\}$		1b	4	$\{15, 02, 11, 06\}$
	17	4	$\{1a, 01, 05, 1e\}$		1d	4	$\{02, 17, 06, 13\}$
	1a	4	$\{1a, 08, 0c, 1e\}$		1e	4	$\{14, 02, 10, 06\}$
	1e	4	$\{1a, 08, 0c, 1e\}$		1f	4	$\{15, 02, 11, 06\}$
$(0,0,1,0)$	01	4	$\{00, 09, 0d, 04\}$	$(1,1,0,1)$	08	8	$\{0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	03	4	$\{00, 0b, 04, 0f\}$		09	4	$\{18, 19, 1d, 1c\}$
	05	4	$\{00, 09, 0d, 04\}$		0a	4	$\{1b, 19, 1d, 1f\}$
	07	4	$\{00, 0b, 04, 0f\}$		0b	4	$\{1b, 18, 1c, 1f\}$
	08	8	$\{00, 0b, 16, 09, 0d, 04, 12, 0f\}$		0c	8	$\{0a, 1b, 18, 19, 1d, 1c, 0e, 1f\}$
	0a	4	$\{0b, 09, 0d, 0f\}$		0d	4	$\{18, 19, 1d, 1c\}$
	0c	8	$\{00, 0b, 16, 09, 0d, 04, 12, 0f\}$		0e	4	$\{1b, 19, 1d, 1f\}$
	0e	4	$\{0b, 09, 0d, 0f\}$		0f	4	$\{1b, 18, 1c, 1f\}$
	11	4	$\{0b, 16, 12, 0f\}$		19	4	$\{0a, 1b, 0e, 1f\}$
	13	4	$\{16, 09, 0d, 12\}$		1a	4	$\{0a, 18, 1c, 0e\}$
	15	4	$\{0b, 16, 12, 0f\}$		1b	4	$\{0a, 19, 1d, 0e\}$
	17	4	$\{16, 09, 0d, 12\}$		1d	4	$\{0a, 1b, 0e, 1f\}$
	1a	4	$\{00, 16, 04, 12\}$		1e	4	$\{0a, 18, 1c, 0e\}$
	1e	4	$\{00, 16, 04, 12\}$		1f	4	$\{0a, 19, 1d, 0e\}$
$(1,0,1,0)$	00	8	$\{1a, 01, 08, 03, 05, 0c, 1e, 07\}$	$(0,1,0,1)$	00	8	$\{15, 14, 02, 17, 10, 11, 06, 13\}$
	02	4	$\{01, 03, 05, 07\}$		01	4	$\{15, 14, 10, 11\}$
	04	8	$\{1a, 01, 08, 03, 05, 0c, 1e, 07\}$		02	4	$\{15, 17, 11, 13\}$
	06	4	$\{01, 03, 05, 07\}$		03	4	$\{14, 17, 10, 13\}$
	09	4	$\{01, 08, 05, 0c\}$		04	8	$\{15, 14, 02, 17, 10, 11, 06, 13\}$
	0b	4	$\{08, 03, 0c, 07\}$		05	4	$\{15, 14, 10, 11\}$
	0d	4	$\{01, 08, 05, 0c\}$		06	4	$\{15, 17, 11, 13\}$
	0f	4	$\{08, 03, 0c, 07\}$		07	4	$\{14, 17, 10, 13\}$
	12	4	$\{1a, 08, 0c, 1e\}$		11	4	$\{02, 17, 06, 13\}$
	16	4	$\{1a, 08, 0c, 1e\}$		12	4	$\{14, 02, 10, 06\}$
	19	4	$\{1a, 03, 1e, 07\}$		13	4	$\{15, 02, 11, 06\}$
	1b	4	$\{1a, 01, 05, 1e\}$		15	4	$\{02, 17, 06, 13\}$
	1d	4	$\{1a, 03, 1e, 07\}$		16	4	$\{14, 02, 10, 06\}$
	1f	4	$\{1a, 01, 05, 1e\}$		17	4	$\{15, 02, 11, 06\}$

Table 7: 4-bit Value-Difference Distribution Table of KECCAK Sbox

(y_3, y_4, y_3', y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$	(y_3, y_4, y_3', y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$
$(0,1,0,0)$	01	2	$\{17, 13\}$	$(1,0,1,1)$	01	2	$\{1a, 1e\}$
	02	4	$\{14, 02, 10, 06\}$		02	4	$\{1a, 08, 0c, 1e\}$
	03	2	$\{15, 11\}$		03	2	$\{1a, 1e\}$
	05	2	$\{17, 13\}$		05	2	$\{1a, 1e\}$
	06	4	$\{14, 02, 10, 06\}$		06	4	$\{1a, 08, 0c, 1e\}$
	07	2	$\{15, 11\}$		07	2	$\{1a, 1e\}$
	09	2	$\{02, 06\}$		09	2	$\{03, 07\}$
	0b	2	$\{02, 06\}$		0b	2	$\{01, 05\}$
	0d	2	$\{02, 06\}$		0d	2	$\{03, 07\}$
	0f	2	$\{02, 06\}$		0f	2	$\{01, 05\}$
	10	4	$\{14, 02, 10, 06\}$		10	4	$\{1a, 08, 0c, 1e\}$
	11	2	$\{15, 11\}$		11	2	$\{08, 0c\}$
	13	2	$\{17, 13\}$		13	2	$\{08, 0c\}$
	14	4	$\{14, 02, 10, 06\}$		14	4	$\{1a, 08, 0c, 1e\}$
	15	2	$\{15, 11\}$		15	2	$\{08, 0c\}$
	17	2	$\{17, 13\}$		17	2	$\{08, 0c\}$
	18	4	$\{15, 17, 11, 13\}$		18	4	$\{01, 03, 05, 07\}$
	19	2	$\{14, 10\}$		19	2	$\{01, 05\}$
	1a	4	$\{15, 17, 11, 13\}$		1a	4	$\{01, 03, 05, 07\}$
	1b	2	$\{14, 10\}$		1b	2	$\{03, 07\}$
	1c	4	$\{15, 17, 11, 13\}$		1c	4	$\{01, 03, 05, 07\}$
	1d	2	$\{14, 10\}$		1d	2	$\{01, 05\}$
	1e	4	$\{15, 17, 11, 13\}$		1e	4	$\{01, 03, 05, 07\}$
	1f	2	$\{14, 10\}$		1f	2	$\{03, 07\}$
$(1,1,0,0)$	01	2	$\{0a, 0e\}$	$(0,0,1,1)$	01	2	$\{0b, 0f\}$
	03	2	$\{0a, 0e\}$		03	2	$\{09, 0d\}$
	05	2	$\{0a, 0e\}$		05	2	$\{0b, 0f\}$
	07	2	$\{0a, 0e\}$		07	2	$\{09, 0d\}$
	09	2	$\{1b, 1f\}$		09	2	$\{16, 12\}$
	0a	4	$\{0a, 18, 1c, 0e\}$		0a	4	$\{00, 16, 04, 12\}$
	0b	2	$\{19, 1d\}$		0b	2	$\{16, 12\}$
	0d	2	$\{1b, 1f\}$		0d	2	$\{16, 12\}$
	0e	4	$\{0a, 18, 1c, 0e\}$		0e	4	$\{00, 16, 04, 12\}$
	0f	2	$\{19, 1d\}$		0f	2	$\{16, 12\}$
	10	4	$\{1b, 19, 1d, 1f\}$		10	4	$\{0b, 09, 0d, 0f\}$
	11	2	$\{18, 1c\}$		11	2	$\{09, 0d\}$
	12	4	$\{1b, 19, 1d, 1f\}$		12	4	$\{0b, 09, 0d, 0f\}$
	13	2	$\{18, 1c\}$		13	2	$\{0b, 0f\}$
	14	4	$\{1b, 19, 1d, 1f\}$		14	4	$\{0b, 09, 0d, 0f\}$
	15	2	$\{18, 1c\}$		15	2	$\{09, 0d\}$
	16	4	$\{1b, 19, 1d, 1f\}$		16	4	$\{0b, 09, 0d, 0f\}$
	17	2	$\{18, 1c\}$		17	2	$\{0b, 0f\}$
	18	4	$\{0a, 18, 1c, 0e\}$		18	4	$\{00, 16, 04, 12\}$
	19	2	$\{19, 1d\}$		19	2	$\{00, 04\}$
	1b	2	$\{1b, 1f\}$		1b	2	$\{00, 04\}$
	1c	4	$\{0a, 18, 1c, 0e\}$		1c	4	$\{00, 16, 04, 12\}$
	1d	2	$\{19, 1d\}$		1d	2	$\{00, 04\}$
	1f	2	$\{1b, 1f\}$		1f	2	$\{00, 04\}$

Table 8: 4-bit Value-Difference Distribution Table of KECCAK Sbox

(y_3, y_4, y_3', y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$	(y_3, y_4, y_3', y_4')	δ_{in}	VDDT	$S^{-1}(x\|\|y)$
$(0,1,1,0)$	01	2	$\{02, 06\}$	$(1,0,0,1)$	01	2	$\{03, 07\}$
	03	2	$\{02, 06\}$		03	2	$\{01, 05\}$
	05	2	$\{02, 06\}$		05	2	$\{03, 07\}$
	07	2	$\{02, 06\}$		07	2	$\{01, 05\}$
	09	2	$\{17, 13\}$		09	2	$\{1a, 1e\}$
	0a	4	$\{14, 02, 10, 06\}$		0a	4	$\{1a, 08, 0c, 1e\}$
	0b	2	$\{15, 11\}$		0b	2	$\{1a, 1e\}$
	0d	2	$\{17, 13\}$		0d	2	$\{1a, 1e\}$
	0e	4	$\{14, 02, 10, 06\}$		0e	4	$\{1a, 08, 0c, 1e\}$
	0f	2	$\{15, 11\}$		0f	2	$\{1a, 1e\}$
	10	4	$\{15, 17, 11, 13\}$		10	4	$\{01, 03, 05, 07\}$
	11	2	$\{14, 10\}$		11	2	$\{01, 05\}$
	12	4	$\{15, 17, 11, 13\}$		12	4	$\{01, 03, 05, 07\}$
	13	2	$\{14, 10\}$		13	2	$\{03, 07\}$
	14	4	$\{15, 17, 11, 13\}$		14	4	$\{01, 03, 05, 07\}$
	15	2	$\{14, 10\}$		15	2	$\{01, 05\}$
	16	4	$\{15, 17, 11, 13\}$		16	4	$\{01, 03, 05, 07\}$
	17	2	$\{14, 10\}$		17	2	$\{03, 07\}$
	18	4	$\{14, 02, 10, 06\}$		18	4	$\{1a, 08, 0c, 1e\}$
	19	2	$\{15, 11\}$		19	2	$\{08, 0c\}$
	1b	2	$\{17, 13\}$		1b	2	$\{08, 0c\}$
	1c	4	$\{14, 02, 10, 06\}$		1c	4	$\{1a, 08, 0c, 1e\}$
	1d	2	$\{15, 11\}$		1d	2	$\{08, 0c\}$
	1f	2	$\{17, 13\}$		1f	2	$\{08, 0c\}$
$(1,1,1,0)$	01	2	$\{1b, 1f\}$	$(0,0,0,1)$	01	2	$\{16, 12\}$
	02	4	$\{0a, 18, 1c, 0e\}$		02	4	$\{00, 16, 04, 12\}$
	03	2	$\{19, 1d\}$		03	2	$\{16, 12\}$
	05	2	$\{1b, 1f\}$		05	2	$\{16, 12\}$
	06	4	$\{0a, 18, 1c, 0e\}$		06	4	$\{00, 16, 04, 12\}$
	07	2	$\{19, 1d\}$		07	2	$\{16, 12\}$
	09	2	$\{0a, 0e\}$		09	2	$\{0b, 0f\}$
	0b	2	$\{0a, 0e\}$		0b	2	$\{09, 0d\}$
	0d	2	$\{0a, 0e\}$		0d	2	$\{0b, 0f\}$
	0f	2	$\{0a, 0e\}$		0f	2	$\{09, 0d\}$
	10	4	$\{0a, 18, 1c, 0e\}$		10	4	$\{00, 16, 04, 12\}$
	11	2	$\{19, 1d\}$		11	2	$\{00, 04\}$
	13	2	$\{1b, 1f\}$		13	2	$\{00, 04\}$
	14	4	$\{0a, 18, 1c, 0e\}$		14	4	$\{00, 16, 04, 12\}$
	15	2	$\{19, 1d\}$		15	2	$\{00, 04\}$
	17	2	$\{1b, 1f\}$		17	2	$\{00, 04\}$
	18	4	$\{1b, 19, 1d, 1f\}$		18	4	$\{0b, 09, 0d, 0f\}$
	19	2	$\{18, 1c\}$		19	2	$\{09, 0d\}$
	1a	4	$\{1b, 19, 1d, 1f\}$		1a	4	$\{0b, 09, 0d, 0f\}$
	1b	2	$\{18, 1c\}$		1b	2	$\{0b, 0f\}$
	1c	4	$\{1b, 19, 1d, 1f\}$		1c	4	$\{0b, 09, 0d, 0f\}$
	1d	2	$\{18, 1c\}$		1d	2	$\{09, 0d\}$
	1e	4	$\{1b, 19, 1d, 1f\}$		1e	4	$\{0b, 09, 0d, 0f\}$
	1f	2	$\{18, 1c\}$		1f	2	$\{0b, 0f\}$

Table 9: 4-bit Value-Difference Distribution Table of KECCAK Sbox

$(y_2, y_3, y_4, y_2', y_3', y_4')$	δ_{in}	VDDT	$S^{-1}(x\|\|y)$
$(0,0,0,0,0,0)$	00	4	$\{00, 0b, 16, 09\}$
$(0,1,0,0,0,0)$	08	3	$\{01, 08, 03\}$
$(0,0,0,0,1,0)$	08	3	$\{00, 0b, 09\}$
$(0,1,0,0,1,0)$	00	4	$\{1a, 01, 08, 03\}$
$(0,0,1,0,0,1)$	00	4	$\{15, 14, 02, 17\}$
$(0,1,1,0,0,1)$	0c	3	$\{1b, 18, 19\}$
$(0,0,1,0,1,1)$	0c	3	$\{15, 14, 17\}$
$(0,1,1,0,1,1)$	00	4	$\{0a, 1b, 18, 19\}$
$(1,0,0,0,0,0)$	04	4	$\{0d, 04, 12, 0f\}$
$(1,1,0,0,0,0)$	0c	3	$\{05, 0c, 07\}$
$(1,0,0,0,1,0)$	0c	3	$\{0d, 04, 0f\}$
$(1,1,0,0,1,0)$	04	4	$\{05, 0c, 1e, 07\}$
$(1,0,1,0,0,1)$	04	4	$\{10, 11, 06, 13\}$
$(1,1,1,0,0,1)$	08	3	$\{1d, 1c, 1f\}$
$(1,0,1,0,1,1)$	08	3	$\{10, 11, 13\}$
$(1,1,1,0,1,1)$	04	4	$\{1d, 1c, 0e, 1f\}$
$(0,0,0,1,0,0)$	04	4	$\{00, 0b, 16, 09\}$
$(0,1,0,1,0,0)$	0c	3	$\{01, 08, 03\}$
$(0,0,0,1,1,0)$	0c	3	$\{00, 0b, 09\}$
$(0,1,0,1,1,0)$	04	4	$\{1a, 01, 08, 03\}$
$(0,0,1,1,0,1)$	04	4	$\{15, 14, 02, 17\}$
$(0,1,1,1,0,1)$	08	3	$\{1b, 18, 19\}$
$(0,0,1,1,1,1)$	08	3	$\{15, 14, 17\}$
$(0,1,1,1,1,1)$	04	4	$\{0a, 1b, 18, 19\}$
$(1,0,0,1,0,0)$	00	4	$\{0d, 04, 12, 0f\}$
$(1,1,0,1,0,0)$	08	3	$\{05, 0c, 07\}$
$(1,0,0,1,1,0)$	08	3	$\{0d, 04, 0f\}$
$(1,1,0,1,1,0)$	00	4	$\{05, 0c, 1e, 07\}$
$(1,0,1,1,0,1)$	00	4	$\{10, 11, 06, 13\}$
$(1,1,1,1,0,1)$	0c	3	$\{1d, 1c, 1f\}$
$(1,0,1,1,1,1)$	0c	3	$\{10, 11, 13\}$
$(1,1,1,1,1,1)$	00	4	$\{1d, 1c, 0e, 1f\}$

Table 10: 6-bit Value-Difference Distribution Table of KECCAK Sbox (VDDT ≥ 3)

I Preimage Attack on 3-round Xoodyak-XOF

In the following, we give a brief description of the `Xoodyak` hash function and present a preimage attack on 3-round `Xoodyak-XOF`.

I.1 The Xoodyak Hash Function

`Xoodyak` is a permutation-based AEAD and hashing scheme, we focus on `Xoodyak-XOF`. The `Xoodyak-XOF` ($b = 384, c = 256, r = 128$) offers an arbitrary output length l and the preimage resistance is $\min(2^{128}, 2^l)$. We aim to conduct a preimage attack on `Xoodyak-XOF` with a 128-bit digest.

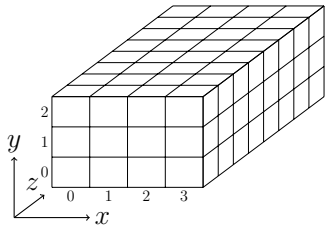


Fig. 1: Toy version of the `Xoodoo` state

`Xoodyak` is built from a fixed 384-bit permutation (called `Xoodoo`), whose state (shown in Figure 1) bit denoted by $A^{(r)}[x][y][z]$ is located at the x-th column, y-th row and z-th lane in the round r, where $0 \leq x \leq 3$, $0 \leq y \leq 2$, $0 \leq z \leq 31$. For `Xoodoo`, all the coordinates are considered modulo 4 for x, modulo 3 for y and modulo 32 for z. The permutation consists of the iteration of a round function $R = \rho_{\text{east}} \circ \chi \circ \iota \circ \rho_{\text{west}} \circ \theta$. Denote the internal states of the round r as

$$\theta : A[x][y][z] \leftarrow A[x][y][z] + \sum_{y'=0}^{2} (A[x-1][y'][z-5] + A[x-11][y'][z-14]).$$

$$\rho_{\text{west}} : A[x][0][z] \leftarrow A[x][0][z], A[x][1][z] \leftarrow A[x-1][1][z], A[x][2][z] \leftarrow A[x][2][z-11].$$

$$\iota : A \leftarrow A + RC[i_r], \text{where } RC[i_r] \text{ is the round constants.}$$

$$\chi : A[x][y][z] \leftarrow A[x][y][z] + (\neg(A[x][y+1][z])) \wedge A[x][y+2][z].$$

$$\rho_{\text{east}} : A[x][0][z] \leftarrow A[x][0][z], A[x][1][z] \leftarrow A[x][1][z-1], A[x][2][z] \leftarrow A[x-2][2][z-8].$$

The addition and multiplication are in $GF(2)$. Since we analyze the round-reduced variant with 3 rounds, we only give the first three round constants: 0x00000058, 0x00000038, 0x000003c0 (given in hexadecimal using the little-endian format).

I.2 The Attack Framework

In the attack on `Xoodyak`, our attack framework is fundamentally similar to framework in the preimage attack against `SHA-3`. We select 3-block messages as inputs, and the target internal difference algorithm in the backward phase is the same as that in the forward phase, which is performed by establishing and solving the input difference system, as shown in Figure 2. The internal differential transition of the j-th round in the characteristic is denoted by $\alpha_{j-1} \xrightarrow{\iota \circ \rho_{\text{west}} \circ \theta} \beta_{j-1} \xrightarrow{\chi} \alpha_j^{\star} \xrightarrow{\rho_{\text{east}}} \alpha_j$, where $j \geq 1$. The period i of the internal differential is 16. Next, we outline these three phases.

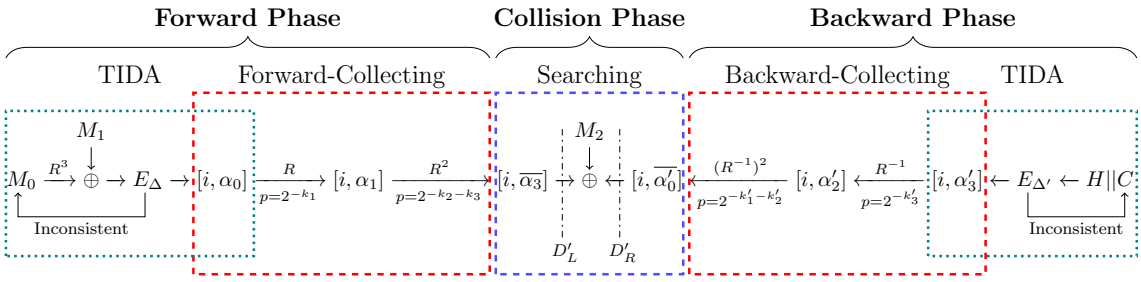


Fig. 2: The framework of 3-round preimage attack

Phase I – Forward Phase. This phase includes two stages: Forward-TIDA and Forward-Collecting Messages.

- Forward-TIDA stage: For the target internal difference α_1, we establish the input difference system E_Δ and filter out the first blocks M_0 that make E_Δ consistent. After that, select α_0 from the solution space of each E_Δ that can be legally propagated to α_1.
- Forward-Collecting Messages stage: Solve the differential transition systems $E_{\beta_0 \to \alpha_1^{\star}}$ to get the second block M_1 passing the first round. Then, compute the messages after 3 rounds functions from the subspaces and filter out the states where the truncated internal difference is $\overline{\alpha_3}$ to store in the set D'_L.

Phase II – Backward Phase. This phase includes Backward-TIDA stage and Backward-Collecting Messages stage.

- Backward-TIDA stage: In this stage, we start with the 128 bits digest H to be recovered and first append a random 256 bits suffix C to H, padding it to 384 bits, denoted by $\overline{H} = H||C$. For the target internal difference α'_2, we establish the input difference system $E_{\Delta'}$, and filter out the random suffix C that make $E_{\Delta'}$. After that, select α'_3 from the solution space of each $E_{\Delta'}$ that can be legally propagated to α'_2.

- Backward-Collecting Messages stage: Solve the differential transition systems $E_{\beta'_2 \leftarrow \alpha'^{\star}_3}$ to get the state $H||C$ passing the last round. Then continue to calculate backward and collect the states with truncated internal difference $\overline{\alpha'_0}$ and store them in the set D'_R.

Phase III – Collision Phase. This phase only includes the searching stage.

- Searching stage: Using hash table techniques, perform an exhaustive search in D'_L and D'_R until two states with the same capacity value belonging to different sets are found. Finally, modify the third block M_2 to complete the matching of the entire state.

I.3 Results

For 3-round `Xoodyak-XOF`, we utilize the internal differential characteristic given in Section G. The transition condition numbers are $(k_1, k_2, k_3, k'_1, k'_2, k'_3) = (116, 0, 8, 6, 28, 72)$. The scales of the sets D'_L and D'_R in the collision phase satisfy $2^{n_1+n_2} = 2^{128}$.

Forward Phase. According to the DDT of `Xoodoo`'s 3-bit Sbox, any non-zero output difference has 4 possible input differences, and these input differences form a 2-dimensional affine space. Therefore, we use the solution space corresponding to a linear equation to characterize the active Sbox in the TIDA stage. For $\alpha^{\star}_1$, there are 58 active Sboxes and 6 non-active Sboxes. During the Forward-TIDA stage, for the input difference system E_{Δ}, each active Sbox provides 1 linear equation, and each non-active Sbox provides 3 linear equations. Thus the system E_{Δ} contains 76 equations, of which the inner part contains 29 equations. Therefore, the complexity of TIDA is 2^{29}. The solution space of each consistent system E_{Δ} contains 2^{17} input differences. Since the differential transition condition number corresponding to each non-zero input difference of `Xoodoo`'s Sbox is 2, the differential transition condition number of the first round is $k_1 = 58 \cdot 2 = 116$. When the input difference is determined, the degree of freedom of the second block message M_1 is $r/2 = 64$, which can only meet 64 differential transition conditions. To meet the remaining 52 conditions, an average of 2^{52} input differences are required. Therefore, to obtain a message $(M_0||M_1)$ that passes the first round of internal differential characteristics, an average of $2^{52-17} = 2^{35}$ TIDA runs are required, with a complexity of 2^{64}.

In the collecting messages stage, to store 2^{n_1} states in the set D'_L, $2^{n_1+k_2+k_3}$ messages passing the first round are required. The total complexity of the forward phase is given by $T_1 = 2^{64+n_1+k_2+k_3}$.

Backward Phase. One property of `Xoodoo`'s Sbox is that its inverse function is equal to itself, so the way to construct the input difference system $E_{\Delta'}$ in the Backward-TIDA stage is the same as in the Forward-TIDA stage, and we obtain the state $H||C$ that can pass the last round of internal differential characteristics by changing the value of the suffix C of the digest H. Since the degree of freedom of the suffix C is 256, it can be demonstrated that the complexity of the TIDA stage in the backward phase is not the dominant factor, and there are enough

degrees of freedom to meet the 72 differential transition conditions in the last round. The total complexity of the backward phase is $T_2 = 2^{n_2+k_1'+k_2'}$.

Complexity. The total complexity $T = T_1 + T_2 = 2^{64+n_1+k_2+k_3} + 2^{n_2+k_1'+k_2'}$. We take $(n_1, n_2) = (45, 83)$, in which case the complexity T takes the minimum value 2^{118}.