New Collision Attacks on Round-Reduced Keccak

Kexin Qiao\(^1,3,4,*\), Ling Song\(^1,2,3(\textit{AS})\), Meicheng Liu\(^1,*\), and Jian Guo\(^2\)

\(^1\) State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, China

\{qiaokexin,songling,liumeicheng\}@iie.ac.cn

\(^2\) Nanyang Technological University, Singapore
guojian@ntu.edu.sg

\(^3\) Data Assurance and Communication Research Center, Chinese Academy of Sciences, China

\(^4\) University of Chinese Academy of Sciences, China

Abstract. In this paper, we focus on collision attacks against \texttt{Keccak} hash function family and some of its variants. Following the framework developed by Dinur \textit{et al.} at FSE 2012 where 4-round collisions were found by combining 3-round differential trails and 1-round connectors, we extend the connectors one round further hence achieve collision attacks for up to 5 rounds. The extension is possible thanks to the large degree of freedom of the wide internal state. By linearization of all S-boxes of the first round, the problem of finding solutions of 2-round connectors are converted to that of solving a system of linear equations. However, due to the quick freedom reduction from the linearization, the system has solution only when the 3-round differential trails satisfy some additional conditions. We develop a dedicated differential trail search strategy and find such special differentials indeed exist. As a result, the first practical collision attack against 5-round \texttt{SHAKE128} and two 5-round instances of the \texttt{Keccak} collision challenges are found with real examples. We also give the first results against 5-round \texttt{Keccak}-224 and 6-round \texttt{Keccak} collision challenges. It is remarked that the work here is still far from threatening the security of the full 24-round \texttt{Keccak} family.

Keywords: \texttt{Keccak}, \texttt{SHA-3}, hash function, linearization, differential.

1 Introduction

The \texttt{Keccak} \cite{3,5} family of hash functions has attracted intensive cryptanalysis since its submission to the \texttt{SHA-3} competition in 2008 \cite{1,9,10,11,13,14,16,17,19}. In 2012, the National Institute of Standards and Technology of the U.S. selected \texttt{Keccak} as the winner of the \texttt{SHA-3} competition. The \texttt{SHA-3} family consists of four cryptographic hash functions of fixed digest sizes and two eXtendable-Output

* This work was done while the authors were visiting Nanyang Technological University.
Functions (XOFs) named SHAKE128 and SHAKE256, each of which is based on an instance of the Keccak algorithms [18]. Keccak[r, c, d] applies sponge construction with bitrate r and capacity c to generate d bit digests from arbitrary length messages where d = 224, 256, 384, 512 in the official SHA-3 versions and d = 160, 80 in the Keccak Crunchy Crypto Collision and Pre-image Contest [2]. Depending on the size of the internal state in r + c bits from the set {200, 400, 800, 1600}, each of the challenge versions contains 4 variants. SHAKE128 and SHAKE256 generate digests that can be extended to any desired length. The suffixes “128” and “256” indicate the security strengths against generic attacks that these two functions support.

In this paper, we focus on collision attacks against the Keccak family, i.e., to find two different messages such that their hash digests are the same. The best previous practical collision attacks on Keccak family are on Keccak-224 and Keccak-256 reduced to 4 rounds found by Dinur et al. [10] in 2012 and later furnished in the journal version [12]. After this, theoretical results improved to 5-round Keccak-256 [11]. However, the number of practically attacked rounds remains at 4. To promote cryptanalysis against Keccak, the Keccak design team proposed smaller variants in the Keccak challenge [2] with 160 digest size for collision attack and 80 digest size for preimage attack with each of the 4 sizes of internal states reduced to from 1 to 12 rounds. The ideal security levels of both are set to be \(2^{80}\) unit computations for collision and preimages, respectively. This is a level much lower than that of the main 4 instances of SHA-3, but still beyond the reach of current computation resource one may have. The current best solutions of collision challenges are instances reduced up to 4 rounds by Dinur et al. [10] and Mendel et al. [17]. Theoretical results were found by Dinur et al. [11] against Keccak-256 with complexities \(2^{115}\) using generalized internal differentials. To the best of our knowledge, this remains as the only results on collision attack against Keccak reduced to 5 or more rounds up to date.

**Our Contribution.** We develop an algebraic and differential hybrid method to launch collision attacks on Keccak family and practically find collisions of 5-round SHAKE128 and two 5-round instances of the Keccak collision challenges. Theoretical results, with complexities below the birthday bound, against 5-round Keccak-224 and 6-round Keccak collision challenges are also achieved.

These results follow a crucial observation that, the Keccak S-box can be re-expressed as linear transformations, when the input is restricted to some affine subspaces. It was already noted by Daemen et al. [3, 8] and Dinur et al. [10] that when the input and output differences are fixed, the solution set of the Keccak S-box contains affine subspaces of dimension up to 3. In this paper, we show the maximum subspaces allowing linearization of S-box is of dimension 2. Furthermore, all affine subspaces of dimension up to 2 allow S-box linearization, and for those of dimension 3, six 2-dimensional affine subspaces out of it could allow the linearization. With this property in mind, we enforce linearization of all S-boxes in the first round, under which the first round function of the
Keccak permutation is transformed into a linear one. Combining with an inversion method of the S-box layer of the second round, we convert the problem of finding two-round connectors into that of solving a system of linear equations. Solving the equation once will produce sufficiently many solutions so that at least one of them will follow the differential trails in the following 3 rounds or more.

A side effect of linearization of all S-boxes is quick reduction of freedom degrees, which in turn decides the existence of such two-round connectors. To solve this problem, we aim to find differential trails, which impose least possible conditions to the two-round connectors. We design a dedicated search strategy to find suitable differential trails of up to 4 rounds. Implementation confirmed the correctness of this idea, and found real examples of collisions for 5-round SHAKE128 and two instances of challenge versions.

We list our results together with the related previous work in Table 1. Note, the algorithm for building 2-round connectors is heuristic and there is no theoretical bound for the solving time. However, it applies to our attacked instances within practical time, so we indicate the real time cost instead of complexities here. Experiments are run on a server with 32 cores of AMD processors.

### Table 1: Collision attack results and comparison

<table>
<thead>
<tr>
<th>Target [r, c, d]</th>
<th>nr</th>
<th>Searching Complexity</th>
<th>Searching Time</th>
<th>Solving Time</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHAKE128</td>
<td>5</td>
<td>$2^{39}$</td>
<td>30m</td>
<td>25m</td>
<td>Sect. 6.2</td>
</tr>
<tr>
<td>Keccak[1440,160,160]</td>
<td>5</td>
<td>$2^{10}$</td>
<td>2.48h</td>
<td>9.6s</td>
<td>Sect. 6.1</td>
</tr>
<tr>
<td>Keccak[640,160,160]</td>
<td>6</td>
<td>$2^{624}$</td>
<td>N.A.</td>
<td>1h</td>
<td>Sect. 6.4</td>
</tr>
<tr>
<td>Keccak-224</td>
<td>4</td>
<td>$2^{24}$</td>
<td>2~3m</td>
<td>N.A.</td>
<td>[10]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2^{12}$</td>
<td>0.3s</td>
<td>2m15s</td>
<td>Sect. 6.6</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>$2^{10}$</td>
<td>N.A.</td>
<td>N.A.</td>
<td>Sect. 6.5</td>
</tr>
<tr>
<td>Keccak-256</td>
<td>4</td>
<td>$2^{24}$</td>
<td>15~30m</td>
<td>N.A.</td>
<td>[10]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2^{12}$</td>
<td>0.28s</td>
<td>7m</td>
<td>Sect. 6.6</td>
</tr>
</tbody>
</table>

1 N.A.: Not Available.
2 There is no theoretical estimate for the solving time of the heuristic algorithms used here.

Organization. The rest of the paper is organized as follows. In Section 2 and Section 3, notations and a brief description of Keccak family are given. In Section 4, we give a detailed description of the algebraic methods to achieve 2-round connectors. In Section 5, we give the dedicated search strategies for differential trails. Then the experimental results are presented in Section 6. We conclude the paper in Section 7.
2 Notations

We summarize the majority of notations to be used in this paper here.

- $c$: 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 of a hash function
- $n_r$: Number of rounds
- $\theta, \rho, \pi, \chi, \iota$: The five step mappings that comprise a round, a subscript $i$ denotes the operation at $i$-th round, e.g., $\theta_i$ denotes the $\theta$ layer at $i$-th round for $i = 0, 1, 2, \cdots$
- $L$: composition of $\theta, \rho, \pi$
- $L^{-1}$: Inverse of $L$
- $RC$: Round constant for a round of Keccak-f permutation
- $S(\cdot)$: 5-bit S-box operating on each row of Keccak state
- $R^i(\cdot)$: Keccak permutation reduced to the first $i$ rounds
- $\delta_{in}$: 5-bit input difference of an S-box,
- $\delta_{out}$: 5-bit output difference of an S-box
- DDT: Differential distribution table
- $\alpha_i$: Input difference of the $i$-th round function, $i = 0, 1, 2, \cdots$
- $\beta_i$: Input difference of $\chi$ in the $i$-th round, $i = 0, 1, 2, \cdots$
- $w_i$: Weight of the $i$-th round, $i = 0, 1, 2, \cdots$
- $m_1 || m_2$: Concatenation of strings $m_1$ and $m_2$
- $x$: Bit value vector before $\chi$ in the first round
- $y$: Bit value vector after the first round
- $z$: Bit value vector before $\chi$ in the second round

3 Description of Keccak

In this section, we give a brief description of Keccak family of hash functions. Keccak family applies sponge construction which processes messages 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 $r$. We denote the original message by $M$ and the padded message by $\overline{M} = M || 10^*1$. The $b$-bit internal state is initialized to be all 0’s. In absorbing phase, the padded message is split into blocks of $r$-bits and each message block is XORed into the first $r$ bits of current internal state, followed by the application of the fixed permutation to the entire $b$-bit state. This is repeated until all message blocks are processed. In the squeezing phase, the first $r$ bits of the state are returned as output, then the permutation is applied and another $r$ bits are outputted until all $d$ output bits are produced. When restricted to the case of $b = 1600$ and $c = 2d$, the four official instances of Keccak family are denoted by Keccak-224/256/384/512 respectively for $d = 224, 256, 384, 512$. SHAKE128 and SHAKE256 are defined from two instances.
of Keccak with the capacity $c$ being 256 and 512, respectively, and the additional appending of a four-bit suffix 1111 to the original message $M$ before applying the Keccak padding. Without further specifications, we presume the digest sizes are 256 and 512 for SHAKE128 and SHAKE256, respectively. We use $\overline{M}$ to denote the padded message for SHAKE as well. Instances of Keccak challenges will be denoted as $\text{Keccak}[r, c, n, d]$, where the parameters are explicitly indicated for the rate, capacity, number of rounds the permutation is reduced to, and bit size of the digest, respectively.

![Figure 1: Sponge Construction [4.](image)](image)

The Keccak permutation function in SHA-3 consists of 24 rounds of five layers operating on the 1600-bit state that can be represented as $5 \times 5$ 64-bit lanes. In general $2^l$ is used to denote the bit length of lanes. If $A$ denotes a 5-by-5-by-$2^l$ array of bits that represents the state, then its indices are the integer triples $(i, j, k)$ for which $0 \leq i < 5$, $0 \leq j < 5$, and $0 \leq k < 2^l$. The bit that corresponds to $(i, j, k)$ is denoted by $A[i, j, k]$. Names for single-dimensional sub-arrays and two-dimensional ones are defined by the Keccak designers: $A[i][j][k]$ is called a row, $A[i][j][·]$ is a column, and $A[i][·][k]$ is a lane; $A[i][j][·]$ is called a sheet, $A[i][·][j][·]$ is a plane, and $A[i][·][j][k]$ is a slice.

The five layers in each round of the permutation are given below:

\[
\begin{align*}
\theta : & A[i][j][k] \leftarrow A[i][j][k] + \sum_{j'=0}^{4} A[i - 1][j'][k] + \sum_{j'=0}^{4} A[i + 1][j'][k - 1] \\
\rho : & A[i][j][k] \leftarrow A[i][j][k + T(i, j)] \text{, where } T(i, j) \text{ is a predefined constant} \\
\pi : & A[i'][j'][k] \leftarrow A[i'][j'][k], \text{ where } \begin{pmatrix} i' \\ j' \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \cdot \begin{pmatrix} i \\ j \end{pmatrix} \\
\chi : & A[i][j][k] \leftarrow A[i][j][k] + (\neg A[i + 1][j][k]) \land A[i + 2][j][k], \\
\iota : & A \leftarrow A + RC[i_r] \text{, where } RC[i_r] \text{ is the round constants.}
\end{align*}
\]
It is interesting to note that the size of permutation can be reduced to one of \(\{25, 50, 100, 200, 400, 800\}\) by choosing \(2^l = 1, 2, 4, 8, 16, 32\), respectively for the size of the lanes. In such cases, the round functions are defined exactly in the same way except the rotation constants of the \(\rho\) operation are now in modulo the respective \(2^l\) instead of 64 in the original 1600-bit full permutation. These size-reduced permutations are not used in the SHA-3 instances, but in the Keccak challenges.

The first three layers are linear mappings and we denote their composition by \(L \triangleq \theta \circ \rho \circ \pi\). The only non-linear layer of the permutation is \(\chi\), which can be seen as a S-box layer that applies 5-bit substitution to 320 rows of the state. We use \(S(x)\) to denote the substitution of a 5-bit input value \(x\). The difference distribution table of the S-box is denoted by \(\text{DDT}\), where \(\text{DDT}(\delta_{in}, \delta_{out})\) represents the size of the set \(\{x : S(x) + S(x + \delta_{in}) = \delta_{out}\}\). We denote the Keccak permutation reduced to the first \(i\) rounds as \(R_i\) (note the round functions are identical up to a difference of constant addition in \(\iota\) and we will omit \(\iota\) as it has little impact on our differential collision attack), i.e., \(R_i(M)\) is the state after \(i\) rounds processing of the padded message \(M\).

4 Overview of Our Collision Attack

In this section, we give an overview of our collision attacks, followed by the details of the algebraic methods to achieve two-round connectors. Without further specification, we assume in this paper the length of the messages used are of one block after padding. To fulfill the Keccak padding rule, one needs to fix the last bit of the padded message to be “1”, hence the first \(r - 1\) bits of the state are under the full control of the attacker through the message bits, and the last \(c\) bits of the state are fixed to zeros as in the IV specified by Keccak. When applied to SHAKE, there are \(r - 6\) free bits under control, by setting the last 6 bits of the padded message to be all 1’s so it is compatible with the specific SHAKE padding rule.

Following the framework by Dinur et al. [10], as well as many other collision attacks utilizing differential trails, our collision attacks consist of two parts, i.e., a high probability differential trail and a connector linking the differential trail with the initial value, as depicted in Figure 2. Let \(\Delta S_I\) and \(\Delta S_O\) denote the input and output differences of the differential trail, respectively. Dinur et al. explored a method, which they call “target difference algorithm”, to find message pairs \((M, M')\) such that the output difference after one round permutation is \(\Delta S_I\), formally \(R^1(M||0^r) + R^1(M'||0^r) = \Delta S_I\). In what follows, we show an algebraic method to extend this connector to two rounds, i.e., a new target difference algorithm to find \((M, M')\) such that \(R^2(M||0^r) + R^2(M'||0^r) = \Delta S_I\). The differential trail is then fulfilled probabilistically with many such message pairs, collision can be produced if the first \(d\) bits of \(\Delta S_O\) are 0. As we are aiming at low complexity attacks, finding solutions of connectors should be practical so that this part will not dominate the overall complexities of collision finding. Details of the differential trail search will be discussed in Section 5.
Overall, the constraints of the two-round connectors are that the last $c + 1$ (or $c + 6$) bits of the initial state are fixed, and the output difference after two rounds is given and fixed (this is determined by the differential trail to be used), we are to utilize the degree of freedom from the first $r - 1$ (or $r - 6$) bits of the initial state to find solutions efficiently. We will start with some observations on the Keccak S-box, then move to the details of solution finding algorithm.

4.1 S-box linearization and affine subspaces

The key observation is that internal state is much larger than the digest size, providing large number of freedom degrees to attackers. One can choose some subsets of the available spaces with special properties to achieve fast enumerations. In case of Keccak, we are to choose the subsets which are linear with respective to the S-box, i.e., the expression of S-box can be re-written as linear transformation when the input is restricted to such subsets. It is obvious to note the S-box is non-linear when the entire $2^5$ input space is considered. However, affine subspaces of size up to 4, as to be shown below, could be found so that the S-box can be linearized. Note that the S-box is the only nonlinear part of the Keccak round function. Hence, the entire round function becomes linear when restricted to such subspaces. Formally, we define

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

For example, when input is restricted to the subset $\{00000, 00001, 00100, 00101\}$ ($\{00, 01, 04, 05\}$ in hex), the corresponding output set of the Keccak S-box is $\{00000, 01001, 00101, 01100\}$ ($\{00, 09, 05, 0C\}$ in hex), and the expression of the
S-box can be re-written as linear transformation:

\[
y = \begin{pmatrix}
1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
\end{pmatrix} \cdot x
\]  

(1)

where \( x \) and \( y \) are bit vector representation of input and output values of the Keccak S-box with the last bit on top. By rotation symmetry, four more linearizable affine subspaces can be deduced from one.

Exhaustive search for the linearizable affine subspaces of the Keccak S-box shows:

**Observation 1** Out of the entire 5-dimensional input space,

a. there are totally 80 2-dimensional linearizable affine subspaces, as listed in Table 5 in Appendix A.

b. there does not exist any linearizable affine subspace with dimension 3 or more.

For completeness, any 1-dimensional subspace is automatically linearizable affine subspace.

Since the affine subspaces are to be used together with differential trails, we are interested in those linearizable affine subspaces with fixed input and output differences, which is more relevant with the differential distribution table (DDT) of S-boxes. Referring to the DDT of Keccak S-box postponed to Appendix B, we observe:

**Observation 2** Given a 5-bit input difference \( \delta_{\text{in}} \) and a 5-bit output difference \( \delta_{\text{out}} \) such that DDT(\( \delta_{\text{in}}, \delta_{\text{out}} \)) \( \neq 0 \), denote the value solution set \( V = \{ x : S(x) + S(x + \delta_{\text{in}}) = \delta_{\text{out}} \} \) and \( S(V) = \{ S(x) : x \in V \} \), we have

a. if DDT(\( \delta_{\text{in}}, \delta_{\text{out}} \)) = 2 or 4, then \( V \) is a linearizable affine subspace.

b. if DDT(\( \delta_{\text{in}}, \delta_{\text{out}} \)) = 8, then there are six 2-dimensional subsets \( W_i \subset V, i = 0, 1, \ldots, 5 \) such that \( W_i(i = 0, 1, \ldots, 5) \) are linearizable affine subspaces.

It is interesting to note the 2-dimensional linearizable affine subspaces obtained from analysis of DDT cover all the 80 cases in observation 1. It is already noted in [15] there is one-to-one correspondence between linearizable affine subspaces and entries with value 2 or 4 in DDT. As for the DDT entries of value 8, we will leave the 6 choices of 2-dimensional linearizable affine subspaces for later usage. As an example, the 3-dimensional affine subspace corresponding to DDT(01, 01), i.e., with both input and output differences being 01, is \( \{10, 11, 14, 15, 18, 19, 1C, 1D\} \) and the six 2-dimensional linearizable affine subspaces from it are

\[
\begin{align*}
\{10, 11, 14, 15\}, \\
\{10, 11, 18, 19\}, \\
\{10, 11, 1C, 1D\}, \\
\{14, 15, 18, 19\}, \\
\{14, 15, 1C, 1D\}, \\
\{18, 19, 1C, 1D\}.
\end{align*}
\]  

(2)
When projected to the whole \texttt{Keccak} state, direct product of affine subspaces of each individual S-box form affine subspaces of the entire state with larger dimensions. In other words, when all the S-boxes in the round function are linearized, the entire round function becomes linear. This will be the way we are to handle the S-box layer of the first round of the 2-round connector.

4.2 A connector covering two rounds

The core idea of our two-round connector is to convert the problem to solving a system of linear equations. Two rounds of \texttt{Keccak} permutation can be expressed as $\chi_1 \circ L_1 \circ \chi_0 \circ L_0$ (omitting the $\iota$). With the $\chi_0$ layer linearized by the techniques discussed above, i.e., given the input and output differences of $\chi$, the first three operations $L_1 \circ \chi_0 \circ L_0$ become linear. We will give details of the method how input and output differences of $\chi_0$ are selected later. Now, we show how the $\chi_1$ can be inverted by adding more constraints of linear equations. In our attack setting, the output difference of $\chi_1$ is given as $\Delta S_I$—input difference of the 3-round differential trail. It is not necessary that all S-boxes of the $\chi_1$ layer are active, i.e., with a non-zero difference. Here only active S-boxes are concerned, and each of them is inverted by randomly choosing an input difference with non-zero number of solutions, we call it \textit{compatible} input difference. Formally, given the output difference $\delta_{out}$, compatible input differences are \{\delta_{in} : DDT(\delta_{in}, \delta_{out}) \neq 0\}. As noted previously \cite{3,8,10}, for any pair of $(\delta_{in}, \delta_{out})$, the solution set $V = \{x : S(x) + S(x + \delta_{in}) = \delta_{out}\}$ forms an affine subspace. In other words, $V$ can be deduced from the set $\{0, 1\}^5$ by setting up $i$ constraints that turn to be binary linear equations, when the size of solution set $V$ is $2^{5-i}$. For example, corresponding to $DDT(03, 02)$ is the 2-dimensional affine subspace $\{14, 17, 1C, 1F\}$ which can be formulated by the following three linear equations:

$$
\begin{pmatrix}
0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1
\end{pmatrix}
\cdot
\begin{pmatrix}
x
\end{pmatrix}
=
\begin{pmatrix}
1 \\
0 \\
1
\end{pmatrix}.
$$

(3)

It is important to note, under the $i$ linear constraints or set $V$, there is a bijective relation between $\delta_{in}$ and $\delta_{out}$, i.e., given one the other can be deduced deterministically. Hence, each active S-box in $\chi_1$ layer is inverted by a choice of compatible input difference together with the corresponding $i$ linear constraints on the input values. Once input difference and linear constraints for all active S-boxes of $\chi_1$ are enforced and fulfilled, solutions of 2-round connector are found. Note a compatible input difference of $\chi_1$ is a choice of $\beta_1$, and $\alpha_1$ can be uniquely determined by the relation $\alpha_1 = L^{-1}(\beta_1)$. In the remaining part of this subsection, more details on implementation of this idea are given.

As depicted in Figure 3, the variables of our equation system are the bit values before the first $\chi$ layer denoted by vector $x$. $y$ and $z$ are bit vectors of intermediate values for further interpretation where $y$ represents the output after the first $\chi$ layer and $z$ the bits before the second $\chi$ layer. The main task is to derive all constraints on differences and affine subspaces to that on the variables.
Now, suppose $\beta_1$ and $\beta_0$ (details will be given in Section 4.4) are fixed, and $\Delta S_I$ (aka. $\alpha_2$) is given, we show how the system of equations could be set up. With the input difference $\beta_1$ and output difference $\Delta S_I$ of $\chi_1$, all the linear constraints on the input affine subspaces of the active S-boxes can be derived and stored as

$$G \cdot z = m,$$

where $G$ is a block-diagonal matrix in which each diagonal block together with corresponding constants in $m$ formulates the constraints of one active S-box. Similar procedure is done for input affine subspaces of the first round, except that the input is restricted to linearizable affine subspaces for all S-boxes regardless whether or not the S-box is active so that $\chi_0$ layer can be replaced by a linear transformation $\chi_L$. We denote the constraints by

$$A \cdot x = t. \quad (4)$$

Then $x$ and $y$ can be linked by

$$\chi_L \cdot x + \chi_C = y,$$

where $\chi_C$ denotes the constant offsets for the affine subspaces. Furthermore, the two equation systems can be linked by

$$L \cdot (y + RC[0]) = z,$$

where $RC[0]$ denotes the round constant of the first round. Note, only active S-boxes of the second round are concerned, i.e., only part of bits of $z$ are known, hence the same applies to $y$, and we use $y'$ to denote the known bits of $y$ for later. Overall, the constraints on $z$ can be derived to that on $x$ as

$$G \cdot L \cdot (\chi_L \cdot x + \chi_C + RC[0]) = m. \quad (5)$$

Note an additional constraint $x$ needs to fulfil is that the last $c + 1$ (or $c + 6$) bits of initial state are pre-fixed, which can be derived as

$$L^{-1}(x) = CA, \quad (6)$$

where $CA$ denotes the preset values for bits of the inner state and padding bits. We use $E_M$ to denote the equation systems (4), (5) and (6), solutions fulfilling $E_M$ will be solutions of 2-round connectors.

**Algorithm for building two-round connectors.** We use the basic linearization procedure to generate the equations for confining $x$ to a smaller subspace suitable for linearization of the first $\chi$ layer and use the main linearization procedure to generate the final equations to bypass the second $\chi$ layer. One of the inputs of the basic procedure is the equation system $E_M$ on $x$ values, other inputs include the input and output differences of the first S-box layer $\beta_0, \alpha_1$ and $y'$.

The Basic Linearization Procedure.

**Inputs:** $E_M$, $\beta_0, \alpha_1$, $y'$.

**Outputs:** updated $E_M$, $\chi_L, \chi_C$. 

10
1. Initialize a matrix $\chi_L$ and a vector $\chi_C$.
2. Iterate on each bit of $y'$, calculate the index of the bit in S-box level, say the $j$-th bit of the $i$-th S-box in the first round. Then for the $i$-th S-box in the first round:
   (a) If the $i$-th S-box has not been processed in this procedure before, then:
      (i) If it is non-active, randomly choose a linearizable 2-dimensional subspace, check whether the 3 equations specifying this 2-dimensional affine subspace is consistent with the current $E_M$.
      
      If so, add them to $E_M$ and update $\chi_L$ and $\chi_C$ with the $j$-th line of the matrix which specifies the affine linear transformation. Continue to next bit of $y'$ in step 2.
      Otherwise, try another linearizable 2-dimensional subspace. If all linearizable 2-dimensional subspaces have been tried and no consistent equations exist, output “No Solution in basic procedure”.

   (ii) Otherwise it is active: find its input and output differences from $\beta_0$ and $\alpha_1$, i.e., $\delta_{in}, \delta_{out}$.
   Case 1. When $\text{DDT}(\delta_{in}, \delta_{out}) = 8$, randomly choose one of the six linearizable 2-dimensional subspaces and the corresponding equation to specialize this 2-dimensional subspace (the other two of the three equations to formulate the 2-dimensional subspace have already been indicated in $E_M$ after choosing $\beta_0$ procedure).
   If current $E_M$ is consistent with this linear equation, add it to $E_M$ and update $\chi_L$ and $\chi_C$ with the $j$-th line of the matrix which specifies the linear map from the 2-dimensional subspace to the output 2-dimensional subspace of S-box. Continue to next bit of $y'$ in step 2.
   Otherwise, try another randomly chosen 2-dimensional linearizable subspace. If all six 2-dimensional linearizable subspaces have been chosen and no consistent equation exist, output “No Solution in basic procedure”.

   Case 2. When $\text{DDT}(\delta_{in}, \delta_{out}) = 2$ or $4$, update $\chi_L$ and $\chi_C$ with the $j$-th line of the matrix which specifies the affine linear transformation of the input 1 or 2-dimensional subspace to the output 1 or 2-dimensional subspace of S-box. Continue to next bit of $y'$ in step 2.

   (b) Otherwise, if the $i$-th S-box has already been processed in this procedure: update $\chi_L$ and $\chi_C$ with the $j$-th line of the matrix which specifies the affine linear transformation of the predefined linearizable subspace to the output subspace of S-box.
3. Output the current equations system $E_M$ as well as $\chi_L$ and $\chi_C$ such that $\chi_L \cdot x + \chi_C = y'$.

The inputs to the Main procedure are $\beta_0, \alpha_1, \beta_1, \alpha_2(\Delta S_I)$ and $E_M$ we get after choosing $\beta_0$.

The Main Linearization Procedure.

**Input:** $E_M, \beta_0, \alpha_1, \beta_1, \alpha_2$. 

11
Output: Updated $E_M$.

1. Using $\beta_1$ and $\alpha_2$, initialize a coefficient matrix $G$ and a constant vector $m$ that specify the linear equations to constrain the input bits of the second S-box layer for deriving the equation $G \cdot z = m$.
2. Derive the $L$ into the matrix format for $L \cdot (y + RC[0]) = z$.
3. Initialize a counter to 0.
4. Execute the basic linear procedure with indexes of known bits $y'$ in $y$ and $E_M, \beta_0$ and $\alpha_1$. If the procedure succeeds, it will return the matrix specifying the linearization of the first S-box layer such that $\chi_L \cdot x + \chi_C = y'$, then continue to Step 6. Otherwise, go to step 5.
5. Increment the counter. If the counter’s value is equal to a preset threshold $T_1$, output “Failed”. Otherwise, go to step 4.
6. Test whether the equation system (5) is consistent with $E_M$. If so, add the new system to $E_M$ and output final $E_M$. Otherwise, go to step 5.

Note that the algorithms do not succeed all the time. To overcome this problem, from the input difference $\Delta S_I$ of a 3-round differential trail, we repeat random picks of compatible input differences $\beta_1$ until the main procedure succeeds. As the number of active S-boxes in $\alpha_2$ is large enough (range from tens to hundreds in our experiments), there are enough different cases for $\beta_1$ resulting in high final success probability. An interesting point is that the inversion from $\alpha_2$ to $\beta_1$ does not need to maintain high probability because this transition is covered by our two-round connector. Besides, the unconstrained number of active S-boxes of an input difference allows more freedom in searching of the most suitable three round differential trails. We will describe the the searching strategies in Section 5. Finally, exhaustive search of solution for the following 3-round differential trails can be performed from the solution space of $E_M$.

4.3 Analysis of degree of freedom

The degree of freedom of solution space of final $E_M$ is a key factor on success of our method. A solution space with degree of freedom larger than the weight of the 3-round differential trail is possible to suggest a message pair with collision digest. After the linearization of the first round, the degree of freedom is $\sum_{i=0}^{b-1} DF(1)^i$ in which $DF(1)^i$ is the degree of freedom of 5-bit input space of the $i$-th S-box in the first round. The value is assigned for $DF(1)^i$ according to rules in Table 2.

The constraints on the initial state reduce $(c + p)$ degree of freedom where $c$ is the capacity and $p$ is due to the padding rule. We have $p = 1$ for Keccak and $p = 6$ for SHAKE. Another decrease on degree of freedom is due to the constraints on the input values of the S-box layer in the second round. The definition of $DF(2)^i$, the degree of freedom of 5-bit input values to S-boxes in the second round, is

$$DF(2)^i = \begin{cases} 
1, & DDT(\delta_{in}, \delta_{out}) = 2, \\
2, & DDT(\delta_{in}, \delta_{out}) = 4, \\
3, & DDT(\delta_{in}, \delta_{out}) = 8, \\
5, & DDT(\delta_{in}, \delta_{out}) = 0,
\end{cases}$$

(7)
Table 2: Rules for value assignment for $\text{DF}^{(1)}_i$.

<table>
<thead>
<tr>
<th>$\text{DF}^{(1)}_i$</th>
<th>Non-active</th>
<th>$\text{DDT}((\delta_{in}, \delta_{out})=2$</th>
<th>$\text{DDT}((\delta_{in}, \delta_{out})=4$</th>
<th>$\text{DDT}((\delta_{in}, \delta_{out})=8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Involved in $y'$</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Not involved in $y'$</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

* The value of $\text{DF}^{(1)}_i$ is based on whether the $i$-th S-box is involved in $y'$ and the value $\text{DDT}(\delta_{in}, \delta_{out})$ where $\delta_{in}$ and $\delta_{out}$ are the input and output differences of the $i$-th S-box in the first round.

where $\delta_{in}$ and $\delta_{out}$ are the input and output differences of the $i$-th S-box in the second round. For the $i$-th S-box in the second round, we add $(5 - \text{DF}^{(2)}_i)$ equations to $E_M$ and suppose to deduce the degree of freedom by this amount.

The degree of freedom of the final $E_M$ is estimated as

$$\text{DF} = \sum_{i=0}^{\frac{b}{5}-1} \text{DF}^{(1)}_i - (c + p) - \sum_{i=0}^{\frac{b}{5}-1} (5 - \text{DF}^{(2)}_i).$$

Large $\text{DF}$ benefits our search for collisions in rounds beyond the second round.

4.4 How to choose $\beta_0$

So far we have not given details on how $\beta_0$ can be selected. We follow Dinur et al.’s work [10] in a more general way to uniquely determine $\beta_0$, the difference before $\chi$ layer in the first round. The algorithm is called “target difference algorithm” and consists of difference phase and value phase.

Given $\Delta S_I$, we have randomly chosen a compatible input difference $\beta_1$. We then build two equation systems $E_\Delta$ and $E_M$ accordingly. $E_\Delta$ is on differences of the message pairs and $E_M$ is on values of one message. The initialization of $E_\Delta$ should abide by 1) the constraints implied by padding rules that the last $c + 1$ difference bits of initial state equal to 0, and 2) the input difference bits of nonactive S-boxes in the first round equal to 0. The initialization of $E_M$ should abide by the padding rules that the last $c + p$ value bits equal to $1^p || 0^c$. We set $p = 1$ for Keccak and $p = 6$ for SHAKE. These rules are easy to be implemented as the variable vector $x$ is an invertible linear mapping of the initial vector. Therefore, in the initialization period, we equate the corresponding bits to their enforced values in $E_\Delta$ and $E_M$.

For $E_\Delta$, we add additional equations to enforce that $\alpha_1$ is possibly deduced from $\beta_0$. Though the obvious way is to equate the 5 input difference bits to a specific value for each active S-box in $\beta_0$, this will restrict the solution space significantly. As suggested in [10], we chose one of the 2-dimensional affine subsets of input differences instead of a specific value for each active S-box. This is based on the fact that given any nonzero 5-bit output difference to a Keccak S-box, the set of possible input differences contains at least five 2-dimensional affine subspaces. After a consistent $E_\Delta$ system has been constructed, the solution space is an affine subspace of candidates for $\beta_0$. Then we continue to maintain $E_\Delta$ by
iteratively add the additional 2 equations to uniquely specify each 5-bit input difference for the active S-boxes. For all active S-boxes, once the specific input differences is determined, we add equations to $E_M$ system to enforce every active 5-bit of $x$ (input bits to active S-box) to an affine subspace corresponding to the uniquely determined $\delta_{in}$ and $\delta_{out}$. In this way, we always find a compatible $\beta_0$ from $\alpha_1$ fulfilling the constraints from the $c + p$ bits of padding and pre-set bits of capacities.

5 Search for Differential Trails

In this section, we elaborate on our searching algorithms for finding differential trails of Keccak. Our ideas greatly benefit from previous works of searching differential trails for Keccak [9,14,19]. We start by recalling several properties of the operations in the round function, followed by our considerations in finding differential trails. Then, we describe our searching algorithms which provide differential trails for practical collision attacks against Keccak[1440, 160, 5, 160], 5-round SHAKE128 and Keccak[640, 160, 5, 160] respectively, and trails for theoretical collision attack against 5-round Keccak-224 and Keccak[1440, 160, 6, 160].

5.1 Properties of $\theta$, $\rho$, $\pi$, $\iota$ and $\chi$

$\theta$, $\rho$, $\pi$, $\iota$ are linear operations while $\chi$ acts as the parallel application of 5-bit nonlinear S-boxes on the rows of the state. Since $\iota$ adds a round constant and has no essential effect on difference, we ignore it in this section. Additionally, $\rho$ and $\pi$ do not change the number of active bits in a differential trail, but only positions. Therefore, $\theta$ and $\chi$ are the crucial parts for differential analysis.

To describe the properties of $\theta$, we take definitions from [3]. The column parity (or parity for short) $P(A)$ of a value (or difference) $A$ is defined as the parity of the columns of $A$, i.e. $P(A)[i][k] = \Sigma_j A[i][j][k]$. A column is even, if its parity is 0, otherwise it is odd. A state is in CP-kernel if all its columns are even.

$\theta$ adds a pattern to the state, and this pattern is called the $\theta$-effect. The $\theta$-effect of a state $A$ is $E(A)[i][k] = P(A)[i - 1][k] + P(A)[i + 1][k - 1]$. So $\theta$ depends only on column parities. The $\theta$-gap is defined as the Hamming weight of the $\theta$-effect divided by two. Note that if the $\theta$-gap is $g$, after applying $\theta$ there are $10g$ bits flipped. Given a state $A$ in CP-kernel, the $\theta$-gap is zero and hence the Hamming weight of $A$ remains after $\theta$. Another interesting property is that $\theta^{-1}$ diffuses much faster than $\theta$. More exactly, a single bit difference can be propagated to about half state bits through $\theta^{-1}$.

Given an input difference to $\chi$, all possible output differences occur with the same probability. On the contrary, given an output difference to $\chi$, it is not the same case, but the highest probability of all possible input differences is determined. Moreover, for one-bit differences, each S-box of $\chi$ acts as identity with probability $2^{-2}$.
5.2 Representation of trails and their weights

As in previous sections, we denote the differences before and after \( i \)-th round by \( \alpha_i \) and \( \alpha_{i+1} \), respectively. Let \( \beta_i = L(\alpha_i) \). Therefore an \( n \)-round differential trail starting from 0-th round is of the following form

\[
\alpha_0 \xrightarrow{L} \beta_0 \xrightarrow{\chi} \alpha_1 \xrightarrow{L} \cdots \xrightarrow{L} \beta_{n-1} \xrightarrow{\chi} \alpha_n.
\]

For the sake of simplicity, a trail can also be represented with only \( \beta_i \)'s or \( \alpha_i \)'s.

The weight of a differential \( \beta \rightarrow \alpha \) over a function \( f \) with domain \( \{0, 1\}^b \) is defined as

\[
w(\beta \rightarrow \alpha) = b - \log_2 |\{ x : f(x) \oplus f(x \oplus \beta) = \alpha \}|.
\]

In other words, the weight of a differential \( \beta \rightarrow \alpha \) is equal to \(-\log_2 \Pr(\beta \rightarrow \alpha)\). If \( \Pr(\beta \rightarrow \alpha) > 0 \), we say \( \alpha \) and \( \beta \) are compatible, otherwise the weight of \( \beta \rightarrow \alpha \) is undefined.

We denote the weight of \( i \)-th round differential by \( w_i \) where \( i \) starts from 0, and thus the weight of a trail is the sum of the weights of round differentials that constitute the trail. In addition, we use \#AS(\( \alpha \)) to represent the number of active S-boxes in a state difference \( \alpha \). According to the properties of \( \chi \), given \( \beta_i \) the weight of \( (\beta_i \rightarrow \alpha_{i+1}) \) is determined; also, given \( \beta_i \) the minimum reverse weight of \( (\beta_{i-1} \rightarrow L^{-1}(\beta_i)) \) is fixed.

As in [3], \( n-1 \) consecutive \( \beta_i \)'s, say \( (\beta_1, \cdots, \beta_{n-1}) \) is called an \( n \)-round trail core which defines a set of \( n \)-round trails \( \alpha_0 \xrightarrow{L} \beta_0 \xrightarrow{\chi} \alpha_1 \xrightarrow{L} \beta_1 \cdots \xrightarrow{L} \beta_{n-1} \xrightarrow{\chi} \alpha_n \) where the first round is of the minimal weight determined by \( \alpha_1 = L^{-1}(\beta_1) \), and \( \alpha_n \) is compatible with \( \beta_{n-1} \). The first step of mount collision attacks against \( n \)-round Keccak is to find good \( (n-1) \)-round trail cores.

5.3 Requirements for differential trails

Good trail cores are those satisfying all the requirements which we will explain as follows. The first requirement is that the difference of the output is zero, i.e. \( \alpha_d^n = 0 \) (we denote output digest difference after \( n \) rounds with \( \alpha_d^n \)). The second requirement relates to the freedom degree budget.

With the definition of weight, Equations (8) can be represented in an alternative way

\[
DF = \sum_{i=0}^{b/5-1} DF_i - (c + p) - w_1.
\]  

The first term of the formula depends on the number of S-boxes that need to be linearized and its corresponding DDT entry as depicted in Table 2. Empirically, when all S-boxes are active and linearized in the first round it is more possible to get a consistent equation system. Therefore, we heuristically set \( \frac{b}{5} \times 2 \) as a threshold for the first term in (9), and denote a threshold of the first two terms in (9) for further search conditions by

\[
TDF = \frac{b}{5} \times 2 - (c + p).
\]
To mount collision attacks against $\text{Keccak}[r, c, n_r, d]$ with methods described in Section 4, it is necessary that

$$\text{TDF} > w_1 + \cdots + w_{n_r-2} + w_{n_r-1}^d$$

(10)

where $w_{n_r-1}^d$ is the part of $w_{n_r-1}$ that relates to the digest\(^5\). The trail searching phase is performed to provide $\Delta ST$ for the connector building algorithm. However, the sufficient conditions for a good trail core is restrained by solving results of the connector, i.e. the number of freedom degrees of the solution space of $E_M$. So we take (10) as a heuristic condition for searching good trail cores which are promising for collision attacks.

Thirdly, the collision attack should be practical. Note that after we obtain a subspace of message pairs making it sure to bypass the first two rounds, the complexity for searching a collision is $2^{w_2 + \cdots + w_{n_r-1}^d}$. To make our attacks practical, we restrict $w_2 + \cdots + w_{n_r-1}^d$ to be small enough, say 48.

We summarize the requirements for differential trails as follows and list TDFs for different versions of $\text{Keccak}[r, c, n_r, d]$ in Table 3.

(1) $\alpha_{n_r}^d = 0$, i.e. the difference of output must be zero.
(2) $\text{TDF} > w_1 + \cdots + w_{n_r-1}^d$, i.e. the degree of freedom must be sufficient;
(3) $w_2 + \cdots + w_{n_r-1}^d \leq 48$, the complexity for finding a collision should be low.

### Table 3: TDFs of different versions of $\text{Keccak}[r, c, n_r, d]$.

<table>
<thead>
<tr>
<th>$\text{Keccak}[r, c, n_r, d]$</th>
<th>TDF</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Keccak}[1440, 160, 5, 160]$</td>
<td>479</td>
<td>Challenge</td>
</tr>
<tr>
<td>$\text{Keccak}[1344, 256, 5, 256]$</td>
<td>378</td>
<td>SHAKE128</td>
</tr>
<tr>
<td>$\text{Keccak}[640, 160, 5, 160]$</td>
<td>159</td>
<td>Challenge</td>
</tr>
<tr>
<td>$\text{Keccak}[1440, 160, 6, 160]$</td>
<td>479</td>
<td>Challenge</td>
</tr>
<tr>
<td>$\text{Keccak}[1152, 448, 5, 224]$</td>
<td>191</td>
<td>$\text{Keccak}$-224</td>
</tr>
</tbody>
</table>

*‘Challenge’ means that that version is included in Keccak Crunchy Crypto Collision and Pre-image Contest [2].

5.4 Searching strategies

**Searching from light $\beta_3$’s.** Our initial goal is to find collisions for 5-round $\text{Keccak}$. To facilitate a 5-round collision of $\text{Keccak}$, we need to find 4-round differential trails satisfying the three requirements mentioned previously. However it is difficult to meet all of them simultaneously even though each of them can be fulfilled easily.

\(^5\) Suppose all the equations are independent. In later parts we will show these equations are not necessarily independent.
We explain as follows. Since we aim for practical attacks, \( w_2 + w_3 + w_4^d \) must be small enough, say 48. That is to say, the last three rounds of the trail must be light and sparse. When we restrict a 3-round trail to be lightweight and extend it backwards for one round, we almost always unfortunately get a heavy state \( \alpha_2 \) (usually \( \#AS(\alpha_2) > 120 \)) whose weight may exceed the TDF. We take Keccak-224 as an example. The TDF of Keccak-224 is 191, which indicates \( \#AS(\alpha_2) < 92 \) as the least weight for an S-box is 2. For a lightweight 3-round trail, it satisfies Requirement (1) occasionally. The greater \( d \) is, the less trails satisfy Requirement (1).

With these requirements in mind, we search for 4-round differential trail cores from light middle state differences \( \beta_3 \)'s. From light \( \beta_3 \)'s we search forwards and backwards, and check whether Requirement(1) and (2) are satisfied respectively; once these two requirements are satisfied, we compute the weight \( w_2 + w_3 + w_4^d \) for brute force, hoping it is small enough for practical attacks.

\( \alpha_3, \alpha_4 \) in CP-kernel. The designers of Keccak show in [3] that it is not possible to construct 3-round low weight differential trails which stay in CP-kernel. However, 2-round differential trails in CP-kernel are possible, as studied in [9, 14, 19].

We restrict \( \alpha_3 \) in CP-kernel. If \( \rho^{-1} \circ \pi^{-1}(\beta_3) \) is outside the CP-kernel and sparse, say 8 active bits, the active bits of \( \alpha_3 = L^{-1}(\beta_3) \) will increase due to the strong diffusion of \( \theta^{-1} \) and the sparseness of \( \beta_3 \). When \( \#AS(\alpha_3) > 10 \), the complexity for searching backwards for one \( \beta_3 \) is greater than \( 2^{31.7} \) which is too time-consuming. We had better also confine \( \alpha_4 \) to the CP-kernel. If not, the requirement \( \alpha_n^d = 0 \) may not be satisfied. As can be seen from the lightest 3-round trail for Keccak-f[1600] [14], even though the \( \theta \)-gap is only one, after \( \theta \) the difference bits are diffused among the state making a 224-bit collision impossible (a 160-bit collision is still possible). So our starting point is special \( \beta_3 \)'s which makes sure \( \alpha_3 = L^{-1}(\beta_3) \) lies in CP-kernel, and for which there exists a compatible \( \alpha_4 \) in CP-kernel. Fortunately, such kind of \( \beta_3 \)'s can be obtained with KeccakTools [6].

Steps for searching 4-round differential trails. We sketch below our steps for finding 4-round differential trail cores for Keccak and provide a description in more detail in Appendix C. To mount collision attacks on 6-round Keccak, 5-round differential trail cores are needed. In this case, we just extend our forward extension for one more round.

1. Using KeccakTools, find special \( \beta_3 \)'s with a low Hamming weight, say 8.
2. For every \( \beta_3 \) obtained, traverse all possible \( \alpha_4 \) using a tree structure, compute \( \beta_4 = L(\alpha_4) \) and test whether there exists a compatible \( \alpha_2 \) where \( \alpha_2^d = 0 \). If so, keep this \( \beta_3 \) and record its forward extension, otherwise discard it.
3. For remaining \( \beta_3 \)'s, also using a tree structure traverse all possible \( \beta_2 \) which is compatible with \( L^{-1}(\beta_3) \)'s, compute \( \#AS(\alpha_2) \) from \( \beta_2 \). If \( \#AS(\alpha_2) \) is small enough, say below 110, check whether this trail core \( (\beta_2, \beta_3, \beta_4) \) under consideration is sufficient for collision attacks.
5.5 Searching results

Some of the best differential trail cores we obtained are listed in Table 4. As can be seen that Trail cores No. 1～3 are all suitable for collision attacks against $\text{Keccak}[1440,160,5,160]$, and Trail cores No. 1 and 2 for $\text{SHAKE128}$. Trail core No. 4 is sufficiently good for collision attacks against $\text{Keccak}[640, 160, 5, 160]$. However, to mount collision attacks on $\text{Keccak}-224$, all the first three trail cores are not good enough. Fortunately, a doubled version of Trail core No. 4 can make our two-round attack possible because $85 \times 2 = 170 < 191$. For $\text{Keccak}[1440, 160, 6, 160]$, we also find a trail core ripe for collision attacks except that Requirement (3) is not satisfied. Details of these differential trail cores are provided in Appendix D.

Table 4: Differential trail cores for $\text{Keccak}[r, c, n, d]$.

<table>
<thead>
<tr>
<th>No.</th>
<th>$r + c$</th>
<th>$#AS(\alpha_2-\beta_2-\beta_3-\beta_4)$</th>
<th>$w_1-w_2-w_3-w_4^d$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1600</td>
<td>102-8-8-2</td>
<td>240-19-16-4</td>
<td>256</td>
</tr>
<tr>
<td>2</td>
<td>1600</td>
<td>88-8-7-0</td>
<td>195-21-15-0</td>
<td>256</td>
</tr>
<tr>
<td>3</td>
<td>1600</td>
<td>85-9-10-2</td>
<td>190-25-20-3</td>
<td>224</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
<td>38-8-8-0</td>
<td>85-20-16-0</td>
<td>160</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No.</th>
<th>$r + c$</th>
<th>$#AS(\alpha_2-\beta_2-\beta_3-\beta_4-\beta_5^d)$</th>
<th>$w_1-w_2-w_3-w_4-w_5^d$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1600</td>
<td>145-6-6-10-14</td>
<td>340-15-12-22-23</td>
<td>160</td>
</tr>
</tbody>
</table>

6 Experiments and Results

In this section, we employ 4-round (5-round) trail cores to mount collision attacks against 5-round (6-round) $\text{Keccak}[r, c, n, d]$. Our attack consists of two main stages:
– **Connecting stage.** Find a subspace of messages bypassing the first two rounds.
– **Brute-force searching stage.** Find a colliding pair from this subspace by brute force.

In the first stage, with \( \alpha_2 \) fixed by the trail core, we choose compatible \( \beta_1 \) where \( \alpha_1 = L^{-1}(\beta_1) \) and all the S-boxes in \( \alpha_1 \) are active. In order to save freedom degrees, we also restrict that \( \beta_1 \rightarrow \alpha_2 \) should be of least weight. When \( \beta_1 \) is chosen, we run the two-round connector. If a certain number of failures is reached, we select another \( \beta_1 \) until a solution is found, i.e. a subspace of message pairs definitely reaching to \( \alpha_2 \) is obtained. If the number freedom degrees of this subspace is large enough, the first stage succeeds. Once the first stage succeeds, we move on to the second stage for finding a colliding message pair.

### 6.1 Collision Attack of Keccak[1440, 160, 5, 160]

We apply Trail core No. 2 to the collision attack of 5-round Keccak[1440, 160, 5, 160]. In this case, we choose compatible \( \beta_1 \)s randomly. After solving the two-round problem in 9.6 seconds, the degree of freedom is 162, which is enough for collision search of the remaining 3 rounds with probability \( 2^{-40} \). The searching time for the collision is 2.48 hours. We give one example of collisions in Table 10, with which we solve a challenge of Keccak Crunchy Crypto Collision and Pre-image Contest [2].

### 6.2 Collision Attack of 5-round SHAKE128

We apply Trail core No. 1 to the collision attack of 5-round SHAKE128. As the capacity of SHAKE128 is much larger than that of Keccak[1440, 160, 5, 160], which means about 100 more freedom degrees are needed, we just choose compatible \( \beta_1 \)s where \( \beta_1 \rightarrow \alpha_2 \) is of least weight. We also follow this rule in later collision attacks. After solving the two-round problem with 25 minutes, the degree of freedom is 94 and the search for 3-round collision with probability \( 2^{-39} \) costs half an hour. We give an instance of collision in Table 11.

### 6.3 Collision Attack of Keccak[640, 160, 5, 160]

We apply Trail core No. 5 to the collision attack of Keccak[640, 160, 5, 160]. The methods used in this case are similar to those of 5-round SHAKE128. The first stage succeeds in 30 minutes. The second stage takes 2 hours 40 minutes to find a collision which happens with probability \( 2^{-35} \). An example of collision is provide in Table 12, with which we solve another challenge of Keccak Crunchy Crypto Collision and Pre-image Contest [2].

---

6 We also utilized Trail core No. 2, but Trail core No. 1 produces a colliding pair in a relatively shorter time.
6.4 Collision Attack of Keccak[1440, 160, 6, 160]

We found four trail cores for which there exist zero 160-bit output differences. The one with the best probability is Trail core No. 5 which is displayed in Table 9. From $\beta_4$ there are 24 trails to zero $\alpha_6^d$. Taking all these trails into consideration, we get a complexity of $2^{67.24} \sim 2^{70.24}$ for the second stage. If we let $\#AS(\alpha_2) (u_2)$ be the smallest, the complexity for the second stage is $2^{70.24} (2^{67.24})$. In the experiments, we let $\#AS(\alpha_2)$ be the smallest. In one hour our two-round algorithm returns a subspace of messages with freedom degree 135, and in 20 minutes we get a message pair shown in Table 13 that follows the first four rounds of the differential trail, which demonstrates that in time complexity of $2^{70.24}$ a collision for 6-round Keccak[1440, 160, 6, 160] will be found with great confidence.

6.5 Collision Attack of 5-round Keccak-224

For the collision attack of 5-round Keccak-224, all the 4-round trail cores we found for Keccak-f[1600] are not good enough, i.e. the weight of the trail cores exceeds TDF too much and even $u_2 > TDF$. However, our two-round connector is still likely to work. For one hand, from Trail core No. 4 for Keccak-f[800] we can construct a 4-round trail core for Keccak-f[1600] with weight pattern (170-40-32-0) which makes our two-round connector possible. From the other, as the capacity increases, it is probable that equations added in connecting phase are not always mutually independent, which means the assumption of freedom degrees of our connector may be less than TDF. The applicability of our connector in this case is verified with experiments. With Trail core No. 4, the two round connector returns a subspace of messages of freedom degree 11 and 2 or 3 for Trail core No. 3. Since the message subspaces derived are too small to mount collision attacks against 5-round Keccak-224, we turn to two-block messages. Once we get $c$ bits from the first block, we set corresponding $c$ bit constants in $E_M$ to the value we obtained and then solve the system to find a subspace of messages for the second block. Now the attack proceeds in the following way.

- **Connecting stage.**
  - Use the two-round connector to find a message subspace with freedom degree $s$ as large as possible, hoping that $t = (c+p) + \text{rank}(E_M \setminus E_{(c,p)}) - \text{rank}(E_M)$ is as small as possible.

- **Brute-force searching stage.**
  - Choose the first message randomly and compute the $c$-bit value for the second block. Replace the corresponding $c$ bit constants in $E_M$ and check whether it is still consistent. If it is consistent, we obtain another subspace with size $2^s$.
  - Search for collision with the subspace.
  - Repeat until we find a two-block collision.

In our experiment, using Trail core No. 3 the connector returns a message subspace with freedom degree $s = 2$, and $t = 55$. Then the complexity for find a two-block collision is $2^{55+(48-2)} = 2^{101}$. 

20
6.6 Re-launch 4-round Collision Attacks of Keccak-224 and Keccak-256

Though the 4-round collisions of Keccak-224 and Keccak-256 have already been found [10], we use our method to optimize the complexity. We start from the same 2-round differential trail in Dinur et al.’s work [10] and build a two-round connector. The time spent on building and solving the two-round connectors is 2 minutes 15 seconds for Keccak-224 and 7 minutes for Keccak-256. Then the complexity for brute forth searching is reduced to $2^{12}$ and cost 0.325 seconds and 0.28 seconds respectively which outperforms $2^{24}$ on-line complexity in [10].

Besides, it is pointed out in [10] that even though they got subsets with more than $2^{30}$ message pairs from their target difference algorithm, they were not able to find collisions within some of these subsets. The reason was suspected to be the incomplete diffusion within the first two rounds and the closely related message pairs within a subset. While in our algorithm, we did not encounter such a problem. In other words, we always find collisions from the subsets deduced from the two-round connector. Thus once we succeed in the 2-round connector building phase with a large enough subset, we never need to repeat it.

7 Conclusion

In conclusion, we observed that the Keccak S-box can be re-expressed as linear transformations under some restricted input subspaces. With this property, we linearized all S-boxes of the first round, and extended the existing connector by one round. Implementations confirmed our idea, and found us real examples of 5-round SHAKE128, and two instances of Keccak challenges. Theoretical results on 5-round Keccak-224 and a 6-round Keccak challenge version are projected.

It is noted that the algorithm for solving the two-round connectors are heuristic, further work includes finding the theoretical bounds of this algorithm and factors deciding the complexities for possible improvements. Note, any relaxation on the restrictions of $\Delta S_f$ might lead us to better differential trails in the searching phase.

Acknowledgement. The authors would like to thank anonymous reviewers and Joan Daemen for their helpful comments and suggestions. The work of this paper was supported by the National Key Basic Research Program of China (2013CB834203) and the National Natural Science Foundation of China (Grants 61472417, 61472415, 61402469, and 61672516).

References

A 2-dimensional Linearizable Affine Subspaces of Keccak S-box

There are totally 80 2-dimensional linearizable affine subspaces for Keccak S-box as listed in Table 5.

<table>
<thead>
<tr>
<th>Table 5: Linearizable affine subspaces of Keccak S-box</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 1, 4, 5]</td>
</tr>
<tr>
<td>[1, 2, 9, A]</td>
</tr>
<tr>
<td>[0, 1, C, D]</td>
</tr>
<tr>
<td>[4, 7, C, F]</td>
</tr>
<tr>
<td>[0, 2, 10, 12]</td>
</tr>
<tr>
<td>[1, 5, 10, 14]</td>
</tr>
<tr>
<td>[10, 11, 14, 15]</td>
</tr>
<tr>
<td>[C, E, 14, 16]</td>
</tr>
<tr>
<td>[D, F, 15, 17]</td>
</tr>
<tr>
<td>[8, A, 18, 1A]</td>
</tr>
<tr>
<td>[9, B, 19, 1B]</td>
</tr>
<tr>
<td>[9, D, 18, 1C]</td>
</tr>
<tr>
<td>[10, 11, 1C, 1D]</td>
</tr>
<tr>
<td>[8, F, 1A, 1E]</td>
</tr>
<tr>
<td>[9, F, 19, 1F]</td>
</tr>
<tr>
<td>[D, F, 1D, 1F]</td>
</tr>
</tbody>
</table>

B Differential Distribution Table of Keccak S-box [14]

C Steps for Finding Differential Trials

In this section, we describe more at length about the steps for finding differential trails.

1. Generate $\beta_3$s each of which makes sure $\alpha_3 = L^{-1}(\beta_3)$ lies in CP-kernel, and for each $\beta_3$ there exists a compatible $\alpha_4$ in CP-kernel using TrailCoreInKernelAtC of KeccakTools [6] where the parameter $aMax\text{Weight}$ is set to be 52. As a result, 503 such $\beta_3$s are obtained.

2. For every $\beta_3$ obtained, if $\#\text{AS}(\beta_3) < 16$ we traverse all possible $\alpha_4$ using a tree structure, compute $\beta_4 = L(\alpha_4)$ and test whether there exists a compatible $\alpha_5$ where $\alpha_5^d = 0$. If so, keep this $\beta_3$ and record its forward extension, otherwise discard it. For $d = 224$, 351 $\beta_3$s are left while 495 $\beta_3$s are left for $d = 160$.

3. For remaining $\beta_3$s, we also use a tree structure to search backwards. For each $\beta_3$ compute $\alpha_3 = L^{-1}(\beta_3)$. If $\#\text{AS}(\alpha_3) < 10$, traverse all possible $\beta_2$s which is compatible with $\alpha_3$, compute $\#\text{AS}(\alpha_2)$ from $\beta_2$. If $\#\text{AS}(\alpha_2)$ is small enough, say below 110, check whether this trail core ($\beta_2, \beta_3, \beta_4$) under consideration is sufficient for collision attacks.

Parameters and conditions in our algorithm can be changed and we just set them as described for the sake of practicality. For example, in the third step if
Table 6: The differential distribution table of the $\chi$ when viewed as S-box. The first bit of a row is viewed as the least significant bit. Given input difference $\Delta_{in}$ and output difference $\Delta_{out}$, the number in the table shows the size of the solution set $\{ v \mid \chi(v) + \chi(v + \Delta_{in}) = \Delta_{out} \}$. Differences are in hex number.

<table>
<thead>
<tr>
<th>$\Delta_{in}$</th>
<th>00</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
<th>0A</th>
<th>0B</th>
<th>0C</th>
<th>0D</th>
<th>0E</th>
<th>0F</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>01</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
</tr>
<tr>
<td>02</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
</tr>
<tr>
<td>03</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>04</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>05</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>06</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
</tr>
<tr>
<td>07</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
</tr>
<tr>
<td>08</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>09</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>0A</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>0B</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
<td>08</td>
</tr>
<tr>
<td>0C</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
<td>04</td>
</tr>
<tr>
<td>0D</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>0E</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>0F</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

$\#AS(\alpha_3) \geq 10$, it costs too much time for one $\beta_3 (\geq 2^{31.7})$. For each $\beta_3$, the backward search costs more time than the forward search because of the property of $\chi$. Since the corresponding $\alpha_3$ has the same number of active bits with $\beta_3$, the numbers of active S-boxes are the same for both extension. Further, active bits are rather sparse in both $\alpha_3$ and $\beta_3$, and the active S-boxes in them are almost all with one active bit. Note that given a one-bit input difference of an S-box, there are only 4 possible output difference, while there are 9 possible input differences given an one-bit output difference. This is the reason why the backward search is more time-consuming.

To find a 5-round trail core for Keccak[1440, 160, 6, 160], we adapt the second step as follows.

2. We first extend forwards from $\beta_3$ for one round using KeccakFTrailExtension of KeccakTools [6] with weight up to 36. Then for $\beta_4$ of each trail core, if $\#AS(\beta_4) < 16$, traverse all possible $\alpha_5$, compute $\beta_5 = L(\alpha_5)$ and test whether there exists a compatible $\alpha_6$ where $\alpha_6^d = 0$. If so, we keep the $\beta_3$ and record the three-round trail core $(\beta_3, \beta_4, \beta_5)$, otherwise we discard the $\beta_3$.

In the end four trail cores remain. In order to reduce the weight, we take multiple trails from $\beta_4$ to $\alpha_5$ into consideration and the trail core in Table 9 is the best among the four trail cores.
D Differential Trails

In this section, we give details of differential trails of Keccak mentioned in Section 5. Actually we present trail cores. For example, a 4-round tail core \((\beta_2, \beta_3, \beta_4)\) consisting three state differences represents a set of 4-round differential trails

\[
\alpha_1 \xrightarrow{L} \beta_1 \xrightarrow{X} \alpha_2 \xrightarrow{L} \beta_2 \xrightarrow{X} \alpha_3 \xrightarrow{L} \beta_3 \xrightarrow{X} \alpha_4 \xrightarrow{L} \beta_4 \xrightarrow{X} \alpha_5
\]

where \(\alpha_5\) is compatible with \(\beta_4\) and \(\beta_1 \rightarrow \alpha_2\) is of the least weight determined by \(\beta_2\). In our collision attacks of 5-round Keccak, 4-round trail cores are needed. In this section, we not only present trail cores used in collision attacks, but also two 5-round trail cores we found.

Each state difference is represented with a matrix of \(5 \times 5\) lanes, ordered from left to right, where ‘|’ is used as a separator between lanes; each lane is given in hexadecimal using the little-endian format and ‘0’ is replaced with ‘-’.

E Collisions for Keccak\([r, c, n_r, d]\)

In this section, we give instances of collisions against Keccak\([1440, 160, 5, 160]\), 5-round SHAKE128 and Keccak\([640, 160, 5, 160]\) respectively. Note that we denote two colliding messages with \(M_1, M_2\). For 5-round Keccak-224 and Keccak\([1440, 160, 6, 160]\), we are unable to find collisions for them because of the limitation of computation power. However, we can demonstrate the soundness of our method by providing instances of massage pairs that follow first 4 rounds of the trail of Keccak\([1440, 160, 6, 160]\) and first 2 rounds of the trail of 5-round Keccak-224.
Table 7: Trail core No. 1 ∼ 3 used in the collision attacks

<table>
<thead>
<tr>
<th>Trail core No. 1, used in the collision attack of 5-round SHAKE128</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_2 )</td>
</tr>
<tr>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trail core No. 2, used in the collision attack of Keccak[1440, 160, 5, 160]</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_2 )</td>
</tr>
<tr>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trail core No. 3, used in the collision attack of 5-round Keccak-224</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_3 )</td>
</tr>
<tr>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

Table 8: Trail core No. 4, used in the collision attack of Keccak[640, 160, 5, 160] and its doubled version can be used in the collision attack of Keccak-224. \( \beta_4 \) has two choices.

<table>
<thead>
<tr>
<th>Trail core No. 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_2 )</td>
</tr>
<tr>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trail core No. 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_3 )</td>
</tr>
<tr>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

26
Table 9: Trail cores No. 5, used in the collision attack of Keccak[1440, 160, 6, 160].

<table>
<thead>
<tr>
<th>( \beta_4 )</th>
<th>( \beta_3 )</th>
<th>( \beta_2 )</th>
<th>( \beta_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
</tr>
<tr>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
</tr>
<tr>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
</tr>
<tr>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
<td>1-1-1-1-1-1-1-1</td>
</tr>
</tbody>
</table>

---

Table 10: Collision for Keccak[1440, 160, 5, 160].

<table>
<thead>
<tr>
<th>Digest</th>
<th>M&lt;sub&gt;1&lt;/sub&gt;</th>
<th>M&lt;sub&gt;2&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>09C5501A19C3C1C7</td>
<td>81D7F806D907E6569334</td>
<td>89182C870A0387A0</td>
</tr>
<tr>
<td>89182C870A0387A0</td>
<td>980A9D8F82C40A90</td>
<td>9305194AEBBC1C17</td>
</tr>
<tr>
<td>9305194AEBBC1C17</td>
<td>D41440AE0C84F2E0</td>
<td>035C1981BFF84755C</td>
</tr>
<tr>
<td>035C1981BFF84755C</td>
<td>E40E0749ED35BB5</td>
<td>37E7FA11AAD390EB</td>
</tr>
<tr>
<td>E40E0749ED35BB5</td>
<td>35C1981BFF84755C</td>
<td>19485675C7530B8E</td>
</tr>
<tr>
<td>35C1981BFF84755C</td>
<td>37E7FA11AAD390EB</td>
<td>042893444D9EC364</td>
</tr>
<tr>
<td>37E7FA11AAD390EB</td>
<td>19485675C7530B8E</td>
<td>6D7DFE249ED35BB5</td>
</tr>
<tr>
<td>19485675C7530B8E</td>
<td>042893444D9EC364</td>
<td>6D317B9B40DE874C</td>
</tr>
<tr>
<td>042893444D9EC364</td>
<td>6D7DFE249ED35BB5</td>
<td>E2EC2A3613678DDA</td>
</tr>
<tr>
<td>6D7DFE249ED35BB5</td>
<td>E2EC2A3613678DDA</td>
<td>3939A7F72AC98F61</td>
</tr>
<tr>
<td>E2EC2A3613678DDA</td>
<td>3939A7F72AC98F61</td>
<td>1440, 160, 6, 160</td>
</tr>
</tbody>
</table>

---

Table 11: Collision for 5-round SHAKE128.

<table>
<thead>
<tr>
<th>Digest</th>
<th>M&lt;sub&gt;1&lt;/sub&gt;</th>
<th>M&lt;sub&gt;2&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>A6E173C9FC3E8EF</td>
<td>0A3E44EBE62104A0</td>
<td>0CDDD5D25A8BD7CA</td>
</tr>
<tr>
<td>0A3E44EBE62104A0</td>
<td>1E8617C352E80FBC</td>
<td>F71D259EE445A4D8</td>
</tr>
<tr>
<td>1E8617C352E80FBC</td>
<td>B69A38114369962F</td>
<td>EB84F177D51C9D45</td>
</tr>
<tr>
<td>B69A38114369962F</td>
<td>1237F5EEA8045DAB</td>
<td>0A70C1FC50024C24</td>
</tr>
<tr>
<td>1237F5EEA8045DAB</td>
<td>D41440AC64E22044C</td>
<td>7108096E3F024F63</td>
</tr>
<tr>
<td>D41440AC64E22044C</td>
<td>0000000000000000</td>
<td>0000000000000000</td>
</tr>
<tr>
<td>0000000000000000</td>
<td>0000000000000000</td>
<td>0000000000000000</td>
</tr>
</tbody>
</table>

---

Table 12: Collision for Keccak[640, 160, 5, 160].

<table>
<thead>
<tr>
<th>Digest</th>
<th>M&lt;sub&gt;1&lt;/sub&gt;</th>
<th>M&lt;sub&gt;2&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>FE431BE34F8371D8</td>
<td>09C5501A19C3C1C7</td>
<td>0CDDD5D25A8BD7CA</td>
</tr>
<tr>
<td>09C5501A19C3C1C7</td>
<td>81D7F806D907E6569334</td>
<td>F71D259EE445A4D8</td>
</tr>
<tr>
<td>81D7F806D907E6569334</td>
<td>89182C870A0387A0</td>
<td>EB84F177D51C9D45</td>
</tr>
<tr>
<td>89182C870A0387A0</td>
<td>980A9D8F82C40A90</td>
<td>0A70C1FC50024C24</td>
</tr>
<tr>
<td>980A9D8F82C40A90</td>
<td>035C1981BFF84755C</td>
<td>7108096E3F024F63</td>
</tr>
<tr>
<td>035C1981BFF84755C</td>
<td>E40E0749ED35BB5</td>
<td>37E7FA11AAD390EB</td>
</tr>
<tr>
<td>E40E0749ED35BB5</td>
<td>35C1981BFF84755C</td>
<td>19485675C7530B8E</td>
</tr>
<tr>
<td>35C1981BFF84755C</td>
<td>37E7FA11AAD390EB</td>
<td>042893444D9EC364</td>
</tr>
<tr>
<td>37E7FA11AAD390EB</td>
<td>19485675C7530B8E</td>
<td>6D7DFE249ED35BB5</td>
</tr>
<tr>
<td>19485675C7530B8E</td>
<td>042893444D9EC364</td>
<td>6D317B9B40DE874C</td>
</tr>
<tr>
<td>042893444D9EC364</td>
<td>6D7DFE249ED35BB5</td>
<td>E2EC2A3613678DDA</td>
</tr>
<tr>
<td>6D7DFE249ED35BB5</td>
<td>E2EC2A3613678DDA</td>
<td>3939A7F72AC98F61</td>
</tr>
<tr>
<td>E2EC2A3613678DDA</td>
<td>3939A7F72AC98F61</td>
<td>1440, 160, 6, 160</td>
</tr>
</tbody>
</table>
Table 13: Collision attack of Keccak[1440, 160, 6, 160]. From a message space of freedom degree 135 which definitely leads to $\beta_2$ of Trail core No. 5, we present a pair of messages following the first 4 rounds. To find a collision of Keccak[1440, 160, 6, 160], it costs a time complexity of $2^{70.24}$.

Table 14: Collision attack of 5-round Keccak-224. From a message space of freedom degree 2 which definitely leads to $\beta_2$ of Trail core No. 3, we present a pair of messages following the first 2 rounds. To find a collision of 5-round Keccak-224, it costs a time complexity of $2^{101}$. 

28