Cryptanalysis of 1-Round KECCAK

Rajendra Kumar[1] and Mahesh Sreekumar Rajasree[1]

Indian Institute of Technology Kanpur, India
rjndr@iitk.ac.in , mahesr@iitk.ac.in

Abstract. In this paper, we give a pre-image attack against 1-round KECCAK-512 hash function which also works for 1-round of all the variants of KECCAK.

Keywords: Cryptanalysis, KECCAK, SHA-3, Preimage attack

1 Introduction

Hash functions are used in digital signatures, message integrity and authentication. In 2006, NIST announced the "NIST hash function competition" which received 64 proposals from round the world. In October 2012, KECCAK designed by Guido Bertoni, Joan Daemen, Michal Peeters, and Gilles Van Assche [3], was selected as the winner of the competition and in 2015, it was standardized as a "Secure Hash Algorithm 3" [9].

The KECCAK hash family is based on the sponge construction[4]. Sponge construction has the property to generate a output of any length and because of this property, SHA3 standards include two extend-able output functions which are SHAKE128 and SHAKE256. These can also be used as a pseudo-random generator. Due to its vast applications, a lot of security analysis are being performed on the KECCAK hash family.

In 2010, D. J. Bernstein [1] gave an idea for second preimage for Keccak variants and in 2014, Chang et al. [5] gave a 1st and 2nd preimage attack. Both have an improvement in time complexity over the bruteforce. Morawiecki et al. [11] gave a theoretical preimage attack upto 4 round of KECCAK by using a technique called as rotational cryptanalysis. Morawiecki et al. [12] performed a preimage analysis of round reduced KECCAK by using toolkit CryptLogVer and SAT solver PrecoSAT. Mara Naya-Plasencia et al. [13] gave a preimage attack on 2-round for KECCAK-224 and KECCAK-256 by using the meet in middle approach. Dinur et al. [6] [8] gave a collision attack upto 4 round using differential and algebraic techniques, and later improved upto 5 rounds using generalized internal differential [7]. In 2016 Guo et al. [10] gave preimage attack for 2 round for KECCAK-224, 256, 384 and 512. The complexity of attack [10] for KECCAK-384 is 2^{129} and for KECCAK-512 is 2^{384}. They extended this upto 4 round for small hash length. Apart from above mentioned attacks, there are several other attacks against KECCAK.

Our contribution: In this paper, we give a preimage attack against 1 round KECCAK-512. The attack does not contain solving of system of equations. It is primarily based on exploiting the degree of freedom in the equations between hash values and the message bits, and convert these equation to simple assignments of values to message variables. Using this method, we can find a message of length less than 2880 bits corresponding to every hash value. Also, the time complexity of this attack is constant. This attack does not pose a threat to the security of 24-round KECCAK.

Organization: The rest of the paper contains the following sections. In Section 2, we briefly describe the structure of KECCAK. In Section 3, we show the preimage analysis of 1 round KECCAK-512 by using the idea of linear structure given by Guo et al. [10]. Section 4 contains the description of our pre-image attack and the appendix contains other details regarding the preimage attack that were skipped in Section 4. Section 5 contains conclusion and future works.

2 Structure of Keccak

Keccak hash function has 3 parameters: r is the bitrate, c is the capacity and n is the output length. It is based on sponge construction[4] which uses a padding function *pad*, a bitrate parameter r and a permutation function f as shown in figure 2.

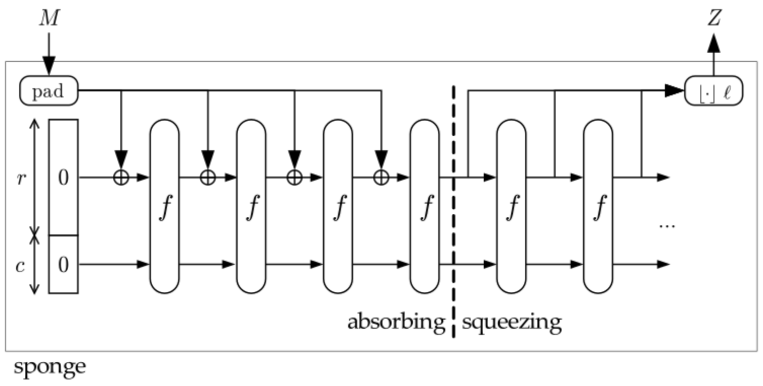


Fig. 1. Sponge function [4]

2.1 Sponge Construction

The sponge construction begins by applying the padding function *pad* on the input string M which produces M' whose length is a multiple of r. M' under goes the absorbing phase as follows.

1. M' is split into blocks of r bits namely $m_1, m_2, ...m_k$.
2. There is an initial string(o_0) which is a b bit string initialized to zero.
3. The initial r bits of o_0 is XORed with first block m_1 and is given as input to f. The output produced by f is denoted by o_1.
4. Similarly, the initial r bits of o_i is XORed with the m_{i+1} and given to f.
5. Finally, the output of the absorbing phase is o_k.

The squeezing phase consist of obtaining the output which can be of any length. Let n be the required output length such that $n = pr + q$ where $q < r$.

1. Apply the f function p more times such that $o_{k+i} = f(o_{k+i-1})$.
2. Let O be the concatenation of the first r bits of each o_{k+i} where $0 \leq i \leq p$.
3. The output of the sponge construction is the first n bits of O.

In case of KECCAK hash function, f is a KECCAK-f permutation and the *pad* function appends 10*1 to input M. KECCAK-f is a specialization of KECCAK-p permutation.

$$KECCAK - f[b] = KECCAK - p[b, 12 + 2l]$$

where $l = log_2(b/25)$.

2.2 KECCAK-p permutation

KECCAK-p permutation is denoted by KECCAK-p$[b, n_r]$, where b is length of input string which is called the width of the permutation, n_r is number of rounds of internal transformation where $b \in \{25, 50, 100, 200, 400, 800, 1600\}$ and n_r being any positive integer. We can define two more quantities $w = b/25$ and $l = log_2(b/25)$. For KECCAK-512 number of rounds of internal transformation n_r is 24 and $b = 1600$. The b bit input string can be represented as a $5 \times 5 \times w$ 3-dimensional array known as state as shown in figure 2.2. A lane in a state S is denoted by $S[x][y]$ which is the substring $S[x][y][0]|S[x][y][1]| \ldots |S[x][y][w-1]$ where $|$ is the concatenation function.

The internal transformation consist of 5 step mappings θ,ρ,π,χ and ι which acts on a state. We give a brief description of each of these step mappings with A and A' being the state before and after applying a step mappings.

1. θ:

$$\begin{aligned} A'[x][y][z] = A[x][y][z] &\oplus CP[(x+1)\ mod\ 5][(z-1)\ mod\ 64] \\ &\oplus CP[(x-1)\ mod\ 5][z] \end{aligned}$$

where $CP[x][z]$ is the parity of a column, i.e,

$$CP[x][z] = A[x][0][z] \oplus A[x][1][z] \oplus A[x][2][z] \oplus A[x][3][z] \oplus A[x][4][z]$$

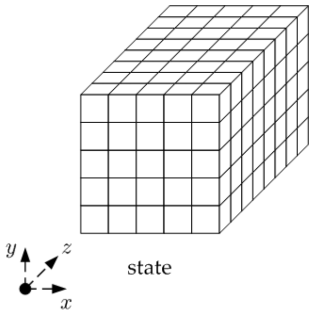


Fig. 2. Keccak state [2]

2. ρ:

$$A'[x][y] = A[x][y] << r[x][y]$$

where $<<$ means bitwise rotation towards MSB of the 64-bit word. The values of $r[x][y]$ are given in the table below.

4	18	2	61	56	14
3	41	45	15	21	8
2	3	10	43	25	39
1	36	44	6	55	20
0	0	1	62	28	27
$y \backslash x$	0	1	2	3	4

3. π:

$$A'[y][2x+3y] = A[x][y]$$

π interchanges the lanes of the state A.

4. χ:

$$A'[x][y][z] = A[x][y][z] \oplus ((A[(x+1)\ mod\ 5][y][z] \oplus 1)$$
$$.A[(x+2)\ mod\ 5][y][z])$$

χ is the only non-linear operation among the 5 step mappings.

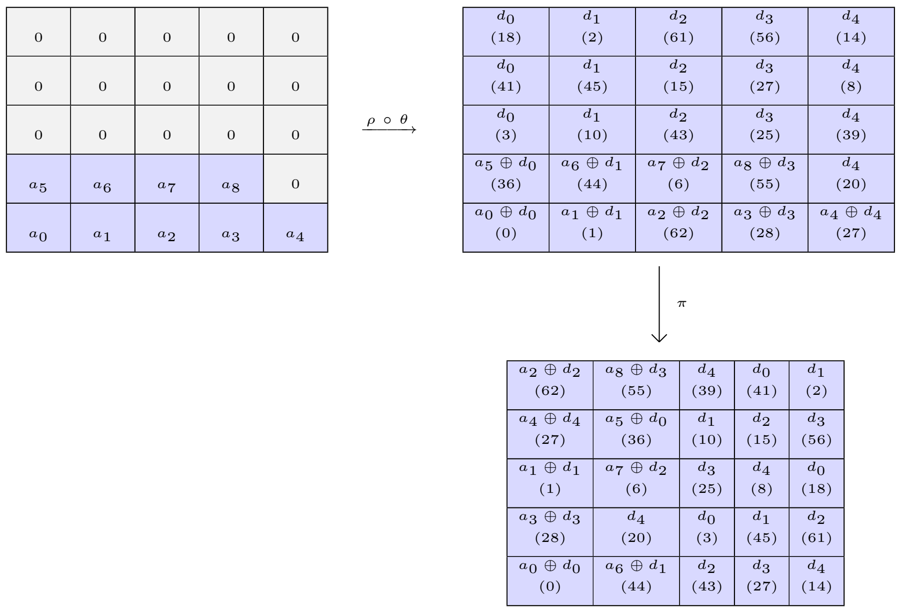
5. ι:

$$A'[0][0] = A[0][0] \oplus RC_i$$

where RC_i is a constant which depends on i where i is the round number.

3 Linear Techniques

In this section, we are going to show our initial idea to attack 1 round KECCAK-512. This is based on the idea of linear structure [10] proposed by Jian Guo et al.. The attack is to invert the hash value by applying ι inverse and χ inverse at the same time convert rest of the operations on message block into linear operations. This forms a system of linear equations and the solution to these equation gives the preimage corresponding to the given hash value. In KECCAK-512, we have $n = 512$, $c = 1024$ and $r = 576$. So, the hash value occupies 8 lanes and message block is of 9 lanes. Suppose $A = a_0a_1a_2a_3a_4a_5a_6a_7a_8$ is the message block of 576 bit length where each a_i is an array of 64 bits. Figure 3 shows the state after applying θ,ρ and π on A where $d_i[k] = CP[i-1][k] + CP[i+1][k-1]$. In all the equations and figures, the value inside the brackets '()' indicate the offset by which the lane is shifted. Every operation between two lanes are bitwise and + symbol refer to XOR operation.



0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
a_5 a_6 a_7 a_8 0
a_0 a_1 a_2 a_3 a_4
$\rho \circ \theta$
d_0 (18) | d_1 (2) | d_2 (61) | d_3 (56) | d_4 (14)
d_0 (41) | d_1 (45) | d_2 (15) | d_3 (27) | d_4 (8)
d_0 (3) | d_1 (10) | d_2 (43) | d_3 (25) | d_4 (39)
$a_5 \oplus d_0$ (36) | $a_6 \oplus d_1$ (44) | $a_7 \oplus d_2$ (6) | $a_8 \oplus d_3$ (55) | d_4 (20)
$a_0 \oplus d_0$ (0) | $a_1 \oplus d_1$ (1) | $a_2 \oplus d_2$ (62) | $a_3 \oplus d_3$ (28) | $a_4 \oplus d_4$ (27)
π
$a_2 \oplus d_2$ (62) | $a_8 \oplus d_3$ (55) | d_4 (39) | d_0 (41) | d_1 (2)
$a_4 \oplus d_4$ (27) | $a_5 \oplus d_0$ (36) | d_1 (10) | d_2 (15) | d_3 (56)
$a_1 \oplus d_1$ (1) | $a_7 \oplus d_2$ (6) | d_3 (25) | d_4 (8) | d_0 (18)
$a_3 \oplus d_3$ (28) | d_4 (20) | d_0 (3) | d_1 (45) | d_2 (61)
$a_0 \oplus d_0$ (0) | $a_6 \oplus d_1$ (44) | d_2 (43) | d_3 (27) | d_4 (14)

Fig. 3. Linear structure

Suppose $H = h_0h_1h_2h_3h_4h_5h_6h_7$ is the 512-bit hash value where each h_i is of 64 bits. We know that we can invert the last row of H and obtain the exact values of $h_0^{'}, h_1^{'}, h_2^{'}, h_3^{'}$ and $h_4^{'}$(shown in figure 4) using the formula given below. The same cannot be done for the second last row.

$$h_i^{'} = h_i + (h_{i+1} + 1).(h_{i+2} + (h_{i+3} + 1).h_{i+4})$$

By equating the 3rd state of figure 3 and 2nd state of figure 4, we get the exact values of d_2, d_3, d_4 and two linear equations

$$a_0 + d_0 = h_0^{'}$$

$$a_6 + d_1 = h_1^{'}$$

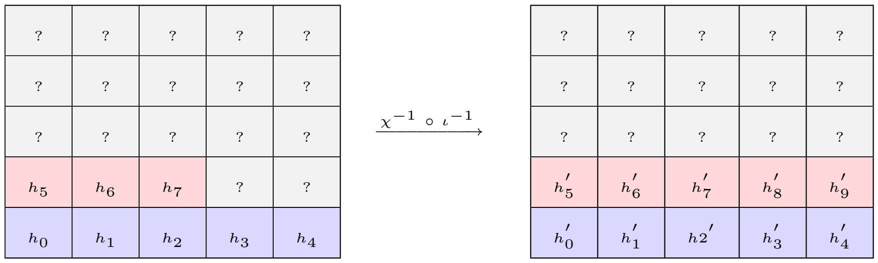


Fig. 4. Inverse operation on hash values

By applying χ on the 3rd state of figure 3, we have the following equations

$$a_3(28) + d_3(28) + (d_4(20) + 1)d_0(3) = h_5$$

$$d_4(20) + (d_0(3) + 1)d_1(45) = h_6$$

$$d_0(3) + (d_1(45) + 1)d_2(61) = h_7$$

The 2nd equation is quadratic while the other two are linear. It can be easily seen that for many cases, there isn't a solution to this system of equations. For example, take the case where $d_4 = h_4' = 0$ and $d_2 = h_2' = 0$. Then,

$$(d_0(3) + 1)d_1(45) = h_6$$

$$d_0(3) = h_7$$

The above equations cannot be solved simultaneously if $h_7 = 1$ and $h_6 = 1$. So we need a different approach to attack 1 round of KECCAK-512.

4 Preimage-Attack

In this section, we discuss in detail the preimage-attack against 1-Round KECCAK-512. For this attack, we consider the message to be of length 2880 bits which will include the padding bits, i.e., the message will contain five blocks A, B, C, D and E. In the analysis, we will ensure that the last bit of E will be 1, so that the padded bits are of form {10*1}. Let S be the initial state of 1600 bits where each bit is zero.

$$S \oplus A \xrightarrow{\iota\circ\chi\circ\pi\circ\rho\circ\theta} X$$

$$X \oplus B \xrightarrow{\iota\circ\chi\circ\pi\circ\rho\circ\theta} Y$$

$$Y \oplus C \xrightarrow{\iota\circ\chi\circ\pi\circ\rho\circ\theta} Z$$

$$Z \oplus D \xrightarrow{\iota\circ\chi\circ\pi\circ\rho\circ\theta} W$$

$$W \oplus E \xrightarrow{\iota\circ\chi\circ\pi\circ\rho\circ\theta} S'$$

The first 512 bits of S' denoted by $h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7$ represents the hash value of the message. Now we invert the last round by applying $\rho^{-1} \circ \pi^{-1} \circ \chi^{-1} \circ \iota^{-1}$.

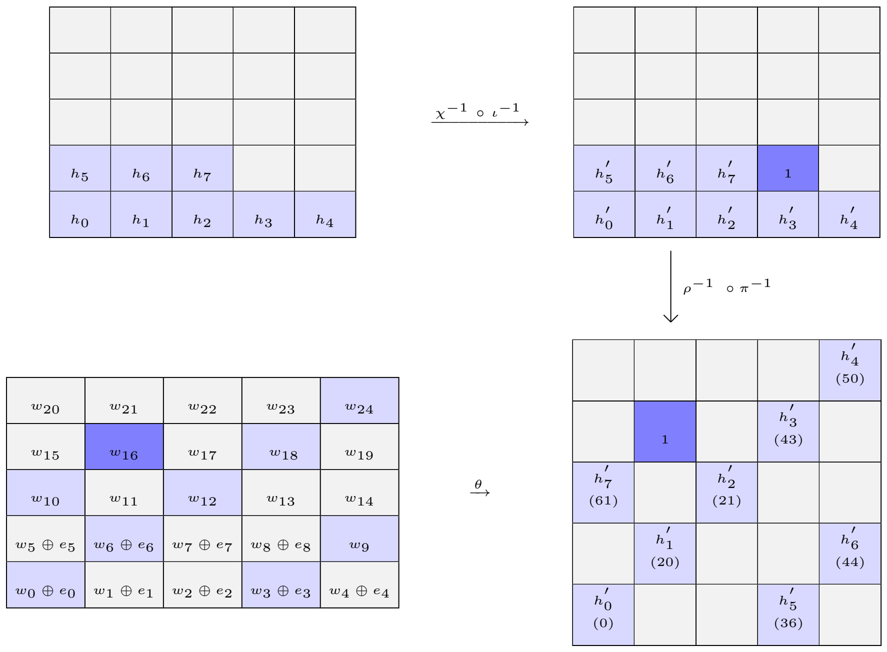


Fig. 5. Inverting last round

In above figure, we do not worry about the blank lanes because we will ensure that θ will act as a identity. Notice that the value of one of the lanes is 1 in-order to invert the χ operation. The following are the values of $h_i^{'}$.

$$h_7^{'} = h_7$$
$$h_6^{'} = \overline{(h_6 + h_7)}$$
$$h_5^{'} = h_5 + (h_6 + h_7)h_7$$
$$h_0^{'} = h_0 + RC_0 + \overline{(h_1)}(h_2 + \overline{h_3}h_4)$$
$$h_1^{'} = h_1 + \overline{(h_2)}(h_3 + \overline{h_4}(h_0 + RC_0))$$
$$h_2^{'} = h_2 + \overline{(h_3)}(h_4 + \overline{(h_0 + RC_0)}h_1)$$

$$h_3^{'} = h_3 + \overline{(h_4)}(h_0 + RC_0 + \overline{h_1}h_2)$$

$$h_4^{'} = h_4 + \overline{(h_0 + RC_0)}(h_1 + \overline{h_2}h_3)$$

4.1 Making θ ineffective

In-order to nullify the effect of θ in each round, we just have to ensure that the parity of every column in the state after being XORed with the message block is zero(or one). This can be ensured because we have the freedom to choose the message block. If the message block $M = m_0m_1m_2m_3m_4m_5m_6m_7m_8$ is being XORed with state S, we can set

$$m_4 = \bigoplus_{i=0}^{4} S[4][i]$$

Similarly, for other columns, we have two lanes as variables, from which we will keep one as a variable and the other to make parity zero.

$$m_j = m_{(j+5 \mod 10)} \bigoplus_{i=0}^{4} S[j][i]$$

4.2 Finding first four message blocks

In $S \oplus A$, to make all column parities zero we have the following assignments

$$a_0 = a_5, a_1 = a_6, a_2 = a_7, a_3 = a_8 \text{ and } a_4 = 0$$

This gives a compact representation of state after applying 1 round over message block A. In Appendix, we have shown how to get the equation of state at the end of single f function in the terms of input state and message block. By making sure the column parity of each round as zero, we get

$$w_{10} = h_7'(61), w_{16} = 1, w_{12} = h_2'(21), w_{18} = h_3'(43), w_{24} = h_4'(50) \text{ and } w_9 = h_6'(44)$$

We now give the equation for w_{12},w_{18},w_{24},w_9,w_{10} and w_{16}.

$$\begin{aligned}
w_9 =& z_{22}(61) + \overline{(z_3(28) + d_3(28))}z_9(20)\\
=& z_{22}(61) + \overline{(z_3(28) + d_3(28))}[\overline{a_1(57)}a_2(62) + a_3(10)]\\
w_{10} =& z_1(1) + d_1(1) + \overline{(z_7(6) + d_7(6))}(z_{13}(25))\\
w_{12} =& a_2(23) + [\overline{a_0(60)}a_1(40) + b_4(60) + 1][a_3(33) + b_5(5)]+\\
& [b_2(41) + \overline{(a_3(62) + b_8(34))a_1(19)}a_2(24) + 1][b_6(6) + c_1(26)]+\\
& [y_{23}(0) + \overline{(y_4(35) + c_4(35))}(y_5(44) + c_5(44)) + 1]\\
& [y_2(16) + c_2(16) + \overline{(y_8(9) + c_8(9))}y_{14}(57)]\\
w_{16} =& z_5(36) + d_5(36) + \overline{z_{11}(10)}z_{17}(15)\\
w_{18} =& b_7(31) + \overline{a_1(51)}a_0(5) + (a_2(46) + a_2(20) + 1)(\overline{a_3(0)})(b_2(5))+\\
& [\overline{a_0(60)}a_1(40) + b_4(60) + (\overline{a_3(33) + b_5(5)})a_2(49)+\\
& (a_3(13) + b_8(49) + 1)(\overline{a_2(2)}a_3(59) + c_2(54)) + 1][z_4(27) + d_4(27)]\\
w_{24} =& a_0(10) + a_0(21) + b_3(21) + c_8(57)+\\
& (a_2(57) + (\overline{a_1(22) + b_1(42)})b_7(47) + 1)\\
& (\overline{a_0(6)}a_1(50) + b_4(6) + (a_3(43) + b_5(15) + 1)a_2(59))+\\
& \overline{(z_2(62) + d_2(62))}(z_8(55) + d_8(55))
\end{aligned}$$

We want a solution of these non-linear equations by using message blocks A, B, C and D. Following are the observations and analysis done on the above equations to get the values of A, B, C and D

1. In state X, all the elements in 2$^{\text{nd}}$ column are zero (refer Appendix A) and so we must have $b_7 = b_2$ to make the column parity 0.
2. We assign $a_1 = 0$ and $b_1 = 0$ which will be helpful later.
3. We assign $a_2(23) = w_{12}$. By using this assignment, we can see that apart from $a_2(23)$, the rest of the terms in the equation of w_{12} must vanish. To achieve this, we set
   - $b_5(5) = a_3(33)$
   - $b_8(34) = a_3(62)$ which is same as $b_8(49) = a_3(13)$
   - $b_2(41) = a_2(24)$

   Now, we have

$$\begin{aligned}
w_{12} =& a_2(23) + [b_6(6) + c_1(26)]+\\
& [y_{23}(0) + \overline{(y_4(35) + c_4(35))}(y_5(44) + c_5(44)) + 1]\\
& [y_2(16) + c_2(16) + \overline{(y_8(9) + c_8(9))}y_{14}(57)]
\end{aligned}$$

   Later, we will assign c_1 with appropriate value so that only $a_2(23)$ remains.

4. By looking at the equation for w_9, since $a_1 = 0$, if we set $a_3(10) = a_2(62)+1$, we get
$$w_9 = z_{22}(61) + z_3(28) + d_3(28) + 1$$
5. We are only left with a_0 for A. This can be achieved from the equation of w_{18} by assigning properly the value of c_2 which will be done later. So, we get
$$a_0(5) = w_{18} + b_7(31) + [a_2(46) + a_2(20) + 1]\overline{a_3(0)}b_2(5)$$
6. Since we now have the first message block A, by applying one round of f on it, we can obtain X. Since we know the values of b_5,b_1,b_2, b_7 and b_8, we can fix the other lanes of message block B such that column parities in $X \oplus B$ are zero and then compute Y.
7. As seen earlier, to get the value of a_0, we must properly assign the value of c_2, which is
$$c_2(54) = 1 + \overline{a_2(2)}a_3(59) + b_4(60) + \overline{[a_3(33) + b_5(5)]}a_2(49)$$
8. To find the value of c_8, we use the equation of w_{24} and setting $d_2 = \overline{z_2}$, i.e, we get
$$c_8(57) = w_{24}+a_0(10)+a_0(21)+b_3(21)+\overline{[a_2(57) + b_7(47)]}[b_4(6)+\overline{(a_3(43) + b_5(15))}a_2(59)]$$
9. We randomly choose c_0 and find c_5, c_7, c_3, c_4 such that the column parity of their respective column is zero.
10. Finally, to find c_1, we again use the equation of w_{12}.
$$u_1 = y_{23}(56) + \overline{(y_4(27) + c_4(27))}(y_5(36) + c_5(36))$$
$$u_2 = y_2(62) + c_2(62) + \overline{(y_8(55) + c_8(55))}y_{14}(39)$$
$$c_1(26) = b_6(6) + \overline{u_1(8)}u_2(18)$$
Using c_1, we can also find the value of c_6 to make the column parity is zero.
11. Now,we know message block C, so we can apply the 1 round of KECCAK-f permutation on $Y \oplus C$ and compute Z.
12. From step 4, we can set
$$d_3(28) = w_9 + z_{22}(61) + z_3(28) + 1$$
13. We can find the value of d_7, using d_2 and Z ensuring that the column parity is zero. From the equation of w_{10}, we get
$$d_1(1) = w_{10} + z_1(1) + (z_7(6) + d_7(6) + 1)z_{13}(25)$$
and from w_{16}, we have
$$d_5(36) = w_{16} + z_5(36) + \overline{z_{11}(10)}z_{17}(15)$$
14. We can now find d_0, d_6, d_8, d_4 to make the column parity zero.

4.3 Finding last message block

Now we want state $W \oplus E$ should have column parities as zero and 1^{st} , 4^{th} ,7^{th} lane is equal to $h_0^{'}$, $h_5^{'}(36)$, $h_1^{'}(20)$ respectively.

$$e_0 = w_0 + h_0^{'}$$

$$e_6 = w_6 + h_1^{'}$$

$$e_3 = w_3 + h_5^{'}$$

Choose e_2 randomly and fix e_5, e_1, e_7, e_8, e_4 such that column parity is zero.

4.4 Ensuring last bit is 1

We still have to ensure that the last bit of E is 1. e_8 depends on the parity of the fourth column of W. The equation for the parity of this column contains c_0 which is chosen randomly, and hence we can expect that the last bit of e_8 will be 1. If not, we repeat the attack.

5 Conclusion and Future Works

Our approach gives a preimage attack to all the variants of 1 round KECCAK hash functions. This is currently the fastest preimage attack known. In future, we need to find whether this idea can be extended to 2 rounds KECCAK-384 and KECCAK-512. This idea is difficult to apply for more rounds because we cannot nullify the θ step mapping and the degree of the equation increase by two after each round. So, it is difficult to find the simpler equation with the extra degree of freedom.

References

1. Daniel J Bernstein. Second preimages for 6 (7?(8??)) rounds of keccak. *NIST mailing list*, 2010.
2. G Bertoni, J Daemen, M Peeters, and GV Assche. The keccak reference. online at http://keccak. noekeon. org/keccak-reference-3.0. pdf, 2011.
3. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak specifications. *Submission to NIST (Round 2)*, 2009.
4. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Cryptographic sponges. *online] http://sponge. noekeon. org*, 2011.
5. Donghoon Chang, Arnab Kumar, Pawell Morawiecki, and Somitra Kumar Sanadhya. 1st and 2nd preimage attacks on 7, 8 and 9 rounds of keccak-224,256,384,512. In *SHA-3 workshop (August 2014)*, 2014.
6. Itai Dinur, Orr Dunkelman, and Adi Shamir. New attacks on keccak-224 and keccak-256. In *FSE*, volume 12, pages 442–461. Springer, 2012.

7. Itai Dinur, Orr Dunkelman, and Adi Shamir. Collision attacks on up to 5 rounds of sha-3 using generalized internal differentials. In *International Workshop on Fast Software Encryption*, pages 219–240. Springer, 2013.
8. Itai Dinur, Orr Dunkelman, and Adi Shamir. Improved practical attacks on round-reduced keccak. *Journal of cryptology*, 27(2):183–209, 2014.
9. Morris J Dworkin. Sha-3 standard: Permutation-based hash and extendable-output functions. *Federal Inf. Process. Stds.(NIST FIPS)-202*, 2015.
10. Jian Guo, Meicheng Liu, and Ling Song. Linear structures: Applications to cryptanalysis of round-reduced keccak. In *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 22*, pages 249–274. Springer, 2016.
11. Paweł Morawiecki, Josef Pieprzyk, and Marian Srebrny. Rotational cryptanalysis of round-reduced keccak. In *International Workshop on Fast Software Encryption*, pages 241–262. Springer, 2013.
12. Paweł Morawiecki and Marian Srebrny. A sat-based preimage analysis of reduced keccak hash functions. *Information Processing Letters*, 113(10-11):392–397, 2013.
13. María Naya-Plasencia, Andrea Röck, and Willi Meier. Practical analysis of reduced-round keccak. In *INDOCRYPT*, volume 7107, pages 236–254. Springer, 2011.

A The equations of the state at the end of *f* function in the terms of input state and message block

By following section 4.1, we can make every θ step as an identity operation. We will find the equations of the state after each KECCAK-f function. Let us look at the state X at the end of the first round.

$$S \oplus A \xrightarrow{KECCAK-f} X$$

In-order to make the θ step ineffective in the first round, we adopt the following assignments.

$$a_5 = a_0, a_6 = a_1, a_7 = a_2, a_8 = a_3 \text{ and } a_4 = 0$$

So, at the end of the first round, the state X will be as given below.

$a_2(62)$	$a_3(56)$	0	$a_2(62)$	$\overline{a_2(62)}a_3(55)$
0	$a_0(36)$	0	0	$a_0(36)$
$a_1(1)$	$a_2(6)$	0	$a_1(1)$	$\overline{a_1(1)}a_2(6)$
$a_3(28)$	0	0	$a_3(28)$	0
$a_0 \oplus RC_0$	$a_1(44)$	0	a_0	$\overline{a_0}a_1(44)$

Fig. 6. State X

Similarly, let's look at state Y at the end of second round. The figure given below shows the state obtained by XORing the first 576 bits of X with message block B and applying $\pi \circ \rho \circ \theta$ to it.

x_{20} (18)	x_{21} (2)	x_{22} (61)	x_{23} (56)	x_{24} (14)
x_{15} (41)	x_{16} (45)	x_{17} (15)	x_{18} (27)	x_{19} (8)
x_{10} (3)	x_{11} (10)	x_{12} (43)	x_{13} (25)	x_{14} (39)
$b_5 \oplus x_5$ (36)	$b_6 \oplus x_6$ (44)	$b_7 \oplus x_7$ (6)	$b_8 \oplus x_8$ (55)	x_9 (20)
$b_0 \oplus x_0$ (0)	$b_1 \oplus x_1$ (1)	$b_2 \oplus x_2$ (62)	$b_3 \oplus x_3$ (28)	$b_4 \oplus x_4$ (27)

$$\xrightarrow{\pi\circ\rho\circ\theta}$$

$b_2 \oplus x_2$ (62)	$b_8 \oplus x_8$ (55)	x_{14} (39)	x_{15} (41)	x_{21} (2)
$b_4 \oplus x_4$ (27)	$b_5 \oplus x_5$ (36)	x_{11} (10)	x_{17} (15)	x_{23} (56)
$b_1 \oplus x_1$ (1)	$b_7 \oplus x_7$ (6)	x_{13} (25)	x_{19} (8)	x_{20} (18)
$b_3 \oplus x_3$ (28)	x_9 (20)	x_{10} (3)	x_{16} (45)	x_{22} (61)
$b_0 \oplus x_0$ (0)	$b_6 \oplus x_6$ (44)	x_{12} (43)	x_{18} (27)	x_{24} (14)

Fig. 7. Applying $\pi \circ \rho \circ \theta$

On further applying $\iota \circ \chi$, we get state Y which is as follows

$$\begin{aligned}
y_0 &= x_0(0) + b_0(0) + (\overline{x_6(44) + b_6(44)})x_{12}(43) + RC_0 \\
y_1 &= x_6(44) + b_6(44) + \overline{x_{12}(43)}x_{18}(21) \\
y_2 &= x_{12}(43) + \overline{x_{18}(21)}x_{24}(14) \\
y_3 &= x_{18}(21) + \overline{x_{24}(14)}(x_0(0) + b_0(0)) \\
y_4 &= x_{24}(14) + (\overline{x_0(0) + b_0(0)})(x_6(44) + b_6(44)) \\
y_5 &= x_3(28) + b_3(28) + \overline{x_9(20)}x_1 0(3) \\
y_6 &= x_9(20) + \overline{x_{10}(3)}x_{16}(45) \\
y_7 &= x_{10}(3) + \overline{x_{16}(45)}x_{22}(61) \\
y_8 &= x_{16}(45) + \overline{x_{22}(61)}(x_3(28) + b_3(28)) \\
y_9 &= x_{22}(61) + (\overline{x_3(28) + b_3(28)})x_9(20) \\
y_{10} &= x_1(1) + b_1(1) + (\overline{x_7(6) + b_7(6)})x_{13}(25) \\
y_{11} &= x_7(6) + b_7(6) + \overline{x_{13}(25)}x_{19}(8) \\
y_{12} &= x_{13}(25) + \overline{x_{19}(8)}x_{20}(18) \\
y_{13} &= x_{19}(8) + \overline{x_{20}(18)}(x_1(1) + b_1(1)) \\
y_{14} &= x_{20}(18) + (\overline{x_1(1) + b_1(1)})(x_7(6) + b_7(6)) \\
y_{15} &= x_4(27) + b_4(27) + (\overline{x_5(36) + b_5(36)})x_{11}(10) \\
y_{16} &= x_5(36) + b_5(36) + \overline{x_{11}(10)}x_{17}(15)
\end{aligned}$$

$$\begin{aligned}
y_{17} &= x_{11}(10) + \overline{x_{17}(15)}x_{23}(56)\\
y_{18} &= x_{17}(15) + \overline{x_{23}(56)}(x_4(27) + b_4(27))\\
y_{19} &= x_{23}(56) + (\overline{x_4(27) + b_4(27)})(x_5(36) + b_5(36))\\
y_{20} &= x_2(62) + b2(62) + (\overline{x_8(55) + b_8(55)})x_{14}(39)\\
y_{21} &= x_8(55) + b_8(55) + \overline{x_{14}(39)}x_{15}(41)\\
y_{22} &= x_{14}(39) + \overline{x_{15}(41)}x_{21}(2)\\
y_{23} &= x_{15}(41) + \overline{x_{21}(2)}(x_2(62) + b_2(62))\\
y_{24} &= x_{21}(2) + (\overline{x_2(62) + b_2(62)})(x_8(55) + b_8(55))
\end{aligned}$$

Similarly we will find the output state Z in terms of Y and C and the output state W in terms of Z and D. Now, we shall write w_9,w_{10},w_{12},w_{16},w_{18} and w_{24} in their simplified form by substituting those variables whose values are known.

$$\begin{aligned}
w_9 =& z_{22}(61) + \overline{(z_3(28) + d_3(28))}z_9(20)\\
=& z_{22}(61) + \overline{(z_3(28) + d_3(28))}[y_{22}(17) + \overline{(y_3(48) + c_3(48))}y_9(40)]\\
=& z_{22}(61) + \overline{(z_3(28) + d_3(28))}[x_{14}(56) + \overline{x_{15}(48)}x_{21}(19) +\\
& \overline{[x_{18}(5) + \overline{x_{24}(62)}(x_0(48) + b_0(48)) + c_3(48)]}[x_{22}(37) + \overline{(x_3(4) + b_3(4))}x_9(60)]]\\
=& z_{22}(61) + \overline{(z_3(28) + d_3(28))}[\overline{a_1(57)}a_2(62) + a_3(10)]\\
w_{10} =& z_1(1) + d_1(1) + \overline{(z_7(6) + d_7(6))}(z_{13}(25))\\
w_{12} =& z_{13}(25) + \overline{z_{19}(8)}z_{20}(18)\\
=& y_{19}(33) + \overline{y_{20}(43)}(y_1(26) + c_1(26)) + \overline{[y_{23}(0) + \overline{(y_4(35) + c_4(35))}(y_5(44) + c_5(44))]}\\
& [y_2(16) + c_2(16) + \overline{(y_8(9) + c_8(9))}y_{14}(57)]\\
=& x_{23}(25) + \overline{(x_4(60) + b_4(60))}(x_5(5) + b_5(5)) + \overline{[x_2(41) + b_2(41) + \overline{(x_8(34) + b_8(34))}x_{14}(18)]}\\
& [x_6(6) + b_6(6) + \overline{x_{12}(5)}x_{18}(47) + c_1(26)] + \overline{[y_{23}(0) + \overline{(y_4(35) + c_4(35))}(y_5(44) + c_5(44))]}\\
& [y_2(16) + c_2(16) + \overline{(y_8(9) + c_8(9))}y_{14}(57)]\\
=& a_2(23) + [\overline{a_0(60)}a_1(40) + b_4(60) + 1][a_3(33) + b_5(5)] +\\
& [b_2(41) + \overline{(a_3(62) + b_8(34))a_1(19)}a_2(24) + 1][b_6(6) + c_1(26)] +\\
& [y_{23}(0) + \overline{(y_4(35) + c_4(35)}(y_5(44) + c_5(44)) + 1]\\
& [y_2(16) + c_2(16) + \overline{(y_8(9) + c_8(9))}y_{14}(57)]\\
w_{16} =& z_5(36) + d_5(36) + \overline{z_{11}(10)}z_{17}(15)
\end{aligned}$$

$$
\begin{aligned}
w_{18} =& z_{17}(15) + \overline{z_{23}(56)}(z_4(27) + d_4(27)) \\
=& y_{11}(25) + \overline{y_{17}(30)}y_{23}(7) + \overline{[y_{15}(33) + \overline{y_{21}(58)}(y_2(54) + c_2(54))]}[z_4(27) + d_4(27)] \\
=& x_7(31) + b_7(31) + \overline{x_{13}(50)}x_{19}(33) + \overline{(x_{11}(40) + \overline{x_{17}(45)}x_{23}(22))}(x_{15}(48) + \overline{x_{21}(9)}(x_2(5) + b_2(5))) + \\
& [x_4(60) + b_4(60) + \overline{(x_5(5) + b_5(5))}x_{11}(43) + \overline{(x_8(49) + b_8(49) + \overline{x_{14}(33)}x_{15}(35))} \\
& (x_{12}(33) + \overline{x_{18}(11)}x_{24}(4) + c_2(54)) + 1](z_4(27) + d_4(27)) \\
=& b_7(31) + \overline{a_1(51)}a_0(5) + (a_2(46) + a_2(20) + 1)(\overline{a_3(0)})(b_2(5)) + \\
& [\overline{a_0(60)}a_1(40) + b_4(60) + (\overline{a_3(33) + b_5(5)})a_2(49) + \\
& (a_3(13) + b_8(49) + 1)(\overline{a_2(2)}a_3(59) + c_2(54)) + 1][z_4(27) + d_4(27)]
\end{aligned}
$$

$$
\begin{aligned}
w_{24} =& z_{21}(2) + \overline{(z_2(62) + d_2(62))}(z_8(55) + d_8(55)) \\
=& y_8(57) + c_8(57) + \overline{y_{14}(41)}y_{15}(43) + + \overline{(z_2(62) + d_2(62))}(z_8(55) + d_8(55)) \\
=& x_{16}(38) + \overline{x_{22}(54)}(x_3(21) + b_3(21)) + c_8(57) + \\
& \overline{[x_{20}(59) + (\overline{x_1(42) + b_1(42)})(x_7(47) + b_7(47))]}[x_4(6) + b_4(6) + (\overline{x_5(15) + b_5(15)})x_{11}(53)] + \\
& \overline{(z_2(62) + d_2(62))}(z_8(55) + d_8(55)) \\
=& a_0(10) + a_0(21) + b_3(21) + c_8(57) + \\
& (a_2(57) + (\overline{a_1(22) + b_1(42)})b_7(47) + 1) \\
& (\overline{a_0(6)}a_1(50) + b_4(6) + (a_3(43) + b_5(15) + 1)a_2(59)) + \\
& \overline{(z_2(62) + d_2(62))}(z_8(55) + d_8(55))
\end{aligned}
$$