Algebraic Attacks on Round-Reduced Keccak[1]

Fukang Liu[1], Takanori Isobe[2,3], Willi Meier[4], Zhonghao Yang[5]

[1]Tokyo Institute of Technology, Tokyo, Japan

[2]University of Hyogo, Hyogo, Japan

[3]NICT, Tokyo, Japan

[4]FHNW, Windisch, Switzerland

[5]East China Normal University, Shanghai, China

PBC 2023

[1]Published at ACISP 2021

Overview

1. Background
   - Sponge
   - Keccak
   - Algebraic Properties of χ
2. Preimage Attacks
   - Linear Structure
   - Improved (Non)linear Structure
   - 3-Round Preimage Attacks
   - 4-Round Preimage Attack
   - New Improvement
3. Summary
   - Summary and Related Problems

Sponge Construction

- EUROCRYPT 2008
- application: hash function, AEAD, PRNG, ...
- concrete ciphers: Keccak, Ascon, Poseidon, ...
- Security: related to (r, c, r')

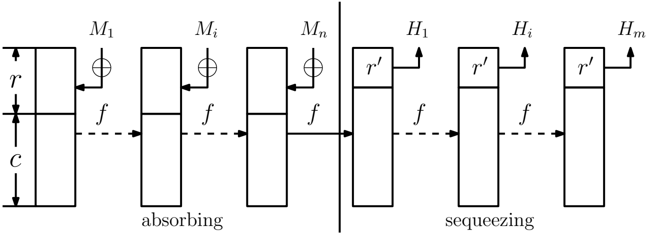
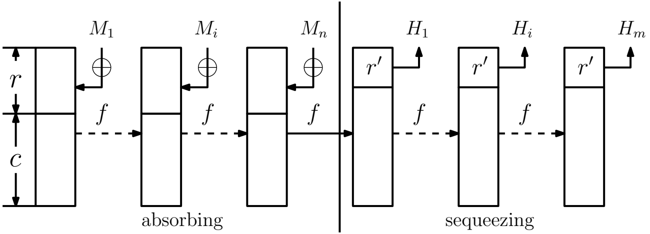


Figure: The sponge construction

Sponge Construction - Keccak

- SHA-3 standard: Keccak-224/256/384/512, SHAKE-128/256
- many other different (r, c, r') in Keccak challenges[2]



- General preimage attacks on Keccak: $\mathcal{O}(2^{r'})$
- r' increases as r decreases (for Keccak-224/256/384/512)

[2] `https://keccak.team/crunchy_contest.html`

Keccak State

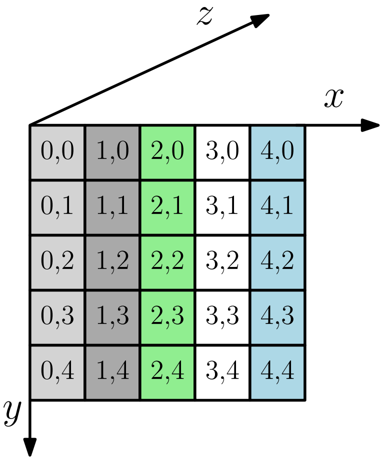
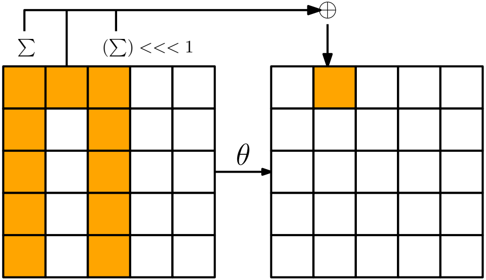


Figure: The Keccak state ($5 \times 5 \times 64$)

Round Function

- #rounds: 24 (TurboSHAKE: 12 rounds)
- $f = R^{24}$ (f : permutation; R: round function)
- $R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$

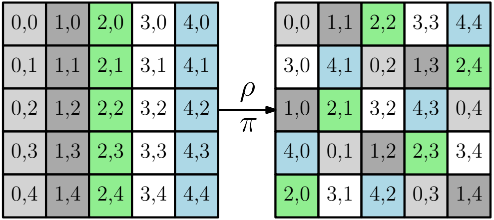


$$\theta : A[x][y] = A[x][y] \oplus \Sigma_{y=0}^{4} A[x-1][y] \oplus (\Sigma_{y=0}^{4} A[x+1][y]) <<< 1$$

Figure: The θ operation

Round Function

- #rounds: 24 (TurboSHAKE: 12 rounds)
- $f = R^{24}$ (f : permutation; R: round function)
- $R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$

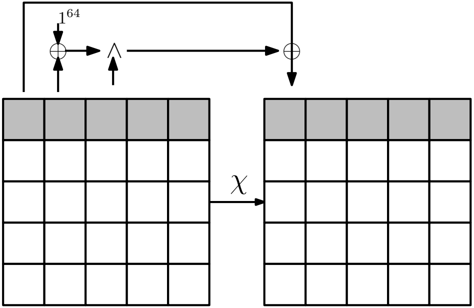


The effect of $\pi \circ \rho$

Figure: The $\pi \circ \rho$ operation

Round Function

- #rounds: 24 (TurboSHAKE: 12 rounds)
- $f = R^{24}$ (f : permutation; R: round function)
- $R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$



$$\chi : A[x][y] = A[x][y] \oplus \overline{A[x+1][y]} \wedge A[x+2][y]$$

Figure: The χ operation

Preimage Attacks on Keccak

- rotational cryptanalysis (FSE 2013)
- linear structure (linear equations, AC 2016)
- linearization technique (quadratic equations, ACISP 2021)
- polynomial method (high-degree equations, EC 2021)
- MitM method (EC 2023)
- better guessing/linearizing strategies (other improvements)

Currently, the best preimage attacks only reach up to 4 rounds.

Preimage Attacks on Keccak

- rotational cryptanalysis (FSE)
- linear structure (linear equations, AC 2017)
- linearization technique (quadratic equations, ACISP 2021)
- polynomial method (hige-degree equations, EC 2021)
- MitM method (EC 2023)
- better guessing/linearizing strategies (other improvements)

Currently, the best preimage attacks only reach up to 4 rounds.

Algebraic Properties of χ

Property 1

For

$$\chi: \; y_i = x_i \oplus (x_{i+1} \oplus 1)x_{i+2}, \; 0 \leq i \leq 4, \tag{1}$$

the following equation always holds:

$$y_i \oplus x_i = (y_{i+1} \oplus 1)x_{i+2}. \tag{2}$$

This implies that if u consecutive output bits $(y_i, \ldots, y_{i+u})$ are known where $2 \leq u \leq 4$, we can obtain $u - 1$ linear equations in $(x_0, \ldots, x_4)$.

Algebraic Properties of χ

Property 2

For

$$\chi: \; y_i = x_i \oplus (x_{i+1} \oplus 1)x_{i+2}, \; 0 \le i \le 4, \tag{3}$$

the following equation holds with probability 0.75:

$$y_i \oplus x_i = 0. \tag{4}$$

This implies that if only y_i is known, we can obtain 1 linear equation in $(x_0, \ldots, x_4)$ with probability 0.75.

Linear Structure for Keccak-512

Keccak-512: $r = 64 \times 9 = 576, r' = 512$

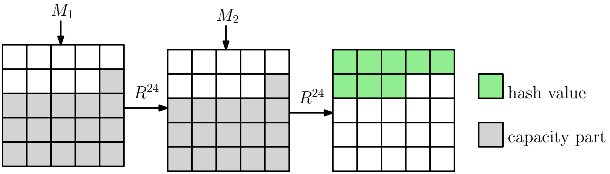


Figure: Illustration of Keccak-512

Linear Structure for Keccak-512

- many leaked linear equations are not used!!! (#eqs > #vars)

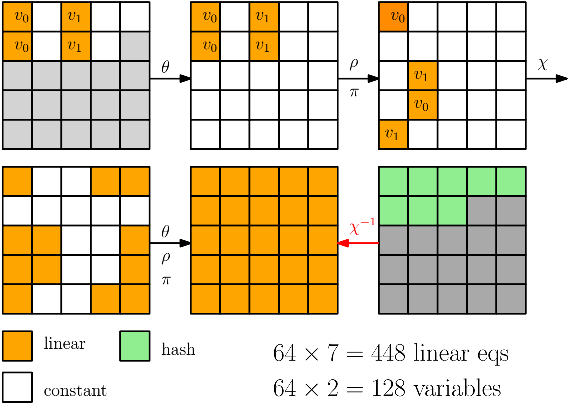


Figure: linear structure of Keccak-512

Linear Structure for Keccak-384

Keccak-384: $r = 64 \times 13 = 832$, $r' = 384$

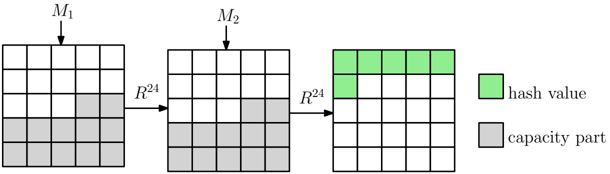


Figure: Illustration of Keccak-384

Linear Structure for Keccak-384

- many leaked linear equations are not used!!! (#eqs > #vars)

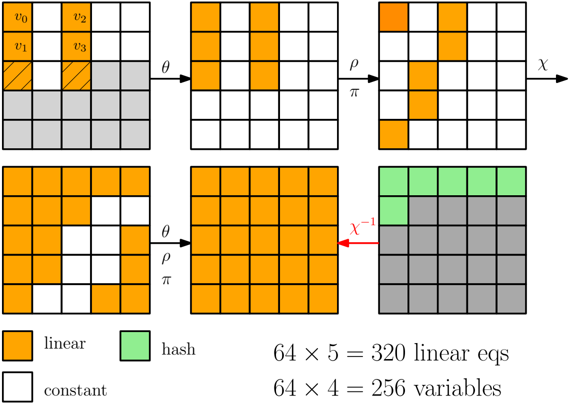


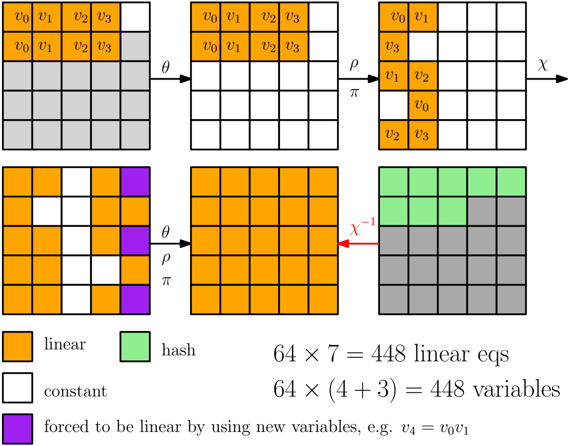
Figure: linear structure of Keccak-384

Our Motivation

Make full use of the leaked linear relations?

Improved (Non)linear Structure for Keccak-512

- Constructing linear structures consumes many degrees of freedom, i.e. many bits have to be set as constants.
- What will happen if those bits are not set as constants?

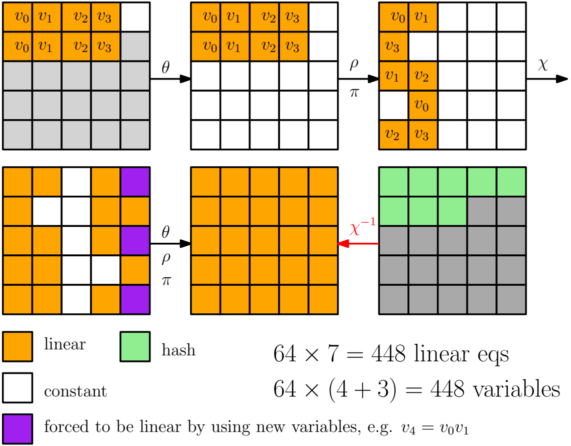


$64 \times 7 = 448$ linear eqs

$64 \times (4 + 3) = 448$ variables

Improved (Non)linear Structure for Keccak-512

- #eqs = # vars!!!
- Solving equations is equivalent to exhausting $2^{64\times4} = 2^{256}$ possible inputs, i.e. the gain is 2^{256}.



$64 \times 7 = 448$ linear eqs

$64 \times (4+3) = 448$ variables

Improved (Non)linear Structure for Keccak-384

- #eqs < # vars (no advantages?)

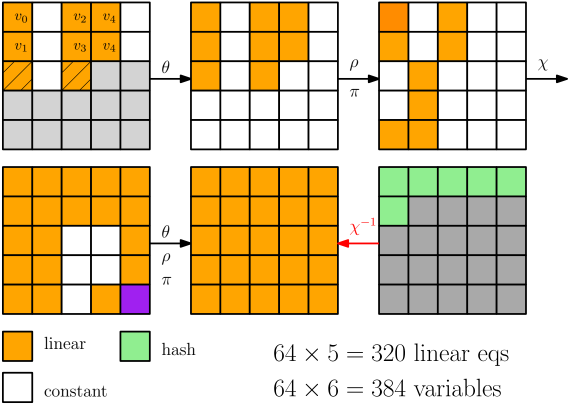


Figure: (non)linear structure of Keccak-384

Improved (Non)linear Structure for Keccak-384

- #eqs = # vars (using probabilistic linear equations)
- gain: $2^{64\times5-27} = 2^{64\times4+37} = 2^{293}$ (time com: $2^{384-293} = 2^{91}$)

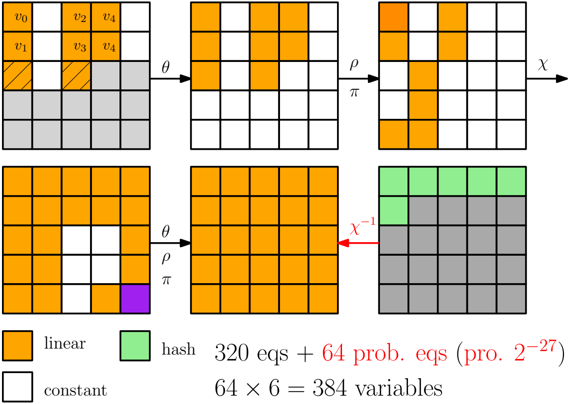


Figure: (non)linear structure of Keccak-384

Summary of the 2-Round Attacks

- Introduce more variables in the inputs to make full use of leaked linear equations
- Allow the first round to be quadratic
- replace quadratic part with new variables

How to use similar ideas for 3-round preimage attacks?

Improved (Non)linear Structure for 3-Round Keccak-512

- Based on the original linear structure
- Use conditions (1st round) to slow down the diffusion
- linearization technique is costly for too many quadratic terms

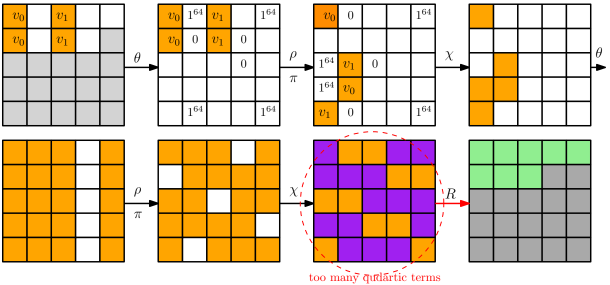


Figure: (non)linear structure of 3-Round Keccak-512

Improved (Non)linear Structure for 3-Round Keccak-512

- Based on the original linear structure
- Use conditions (1st round) to slow down the diffusion
- Further guess equations (2nd round) to slow down the diffusion, i.e. making the linearization technique effective

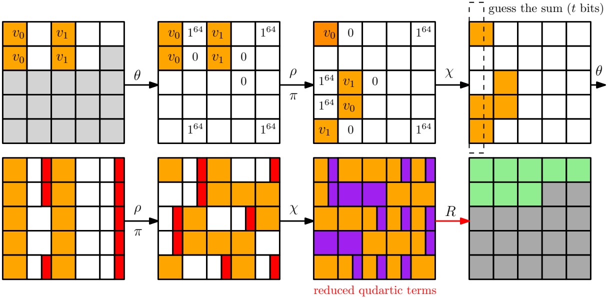


Figure: (non)linear structure of 3-Round Keccak-512

Improved (Non)linear Structure for 3-Round Keccak-512

Determine the required number of guessed equations (t)

- Goal: # eqs $\geq$ # vars
- #eqs: $t + 64 \times 7 = 448 + t$
- #vars: $128 + (64 - t) \times (3 + 1 + 3 + 1 + 3) + 256 = 1088 - 11t$
- $t = 54$: # eqs: 502, # vars: 494 (gain: $2^{64\times 2-54} = 2^{74}$)

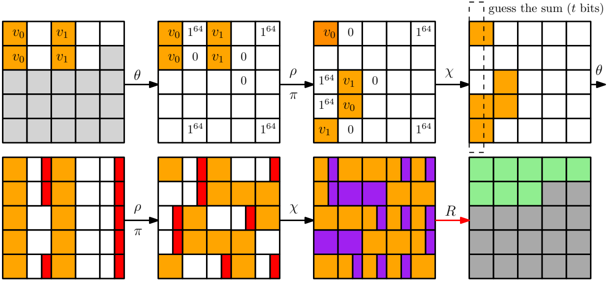


Figure: (non)linear structure of 3-Round Keccak-512

Improved (Non)linear Structure for 3-Round Keccak-512

How to satisfy conditions?

- They can always be satisfied by assigning proper values to constant parts of the second message block
- Left degrees of freedom: 128 bits
- The number of the 1st message blocks: $2^{512-128+2} = 2^{386}$.

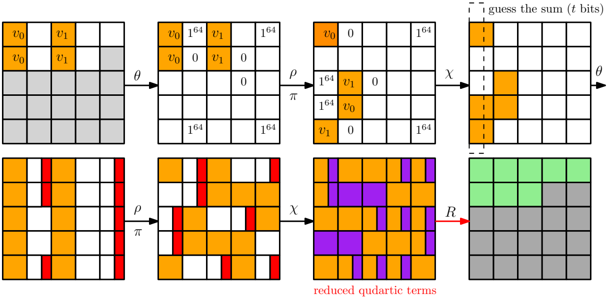


Figure: (non)linear structure of 3-Round Keccak-512

Improved (Non)linear Structure for 3-Round Keccak-512

Details of how to satisfy conditions:

$$
\begin{aligned}
&B_0 = A^0[0][2] \oplus A^0[0][3] \oplus A^0[0][4],\\
&B_2 = A^0[2][2] \oplus A^0[2][3] \oplus A^0[2][4],\\
&B_3 = A^0[3][2] \oplus A^0[3][3] \oplus A^0[3][4],\\
&B_4 = A^0[4][1] \oplus A^0[4][2] \oplus A^0[4][3] \oplus A^0[4][4],\\
&A^0[1][0] \oplus (B_0 \oplus C_0) \oplus (B_2 \oplus C_1) \lll 1 = 1^{64},\\
&A^0[1][1] \oplus (B_0 \oplus C_0) \oplus (B_2 \oplus C_1) \lll 1 = 0,\\
&A^0[1][4] \oplus (B_0 \oplus C_0) \oplus (B_2 \oplus C_1) \lll 1 = 1^{64},\\
&A^0[3][1] \oplus (B_2 \oplus C_1) \oplus (B_4 \oplus A^0[4][0]) \lll 1 = 0,\\
&A^0[3][2] \oplus (B_2 \oplus C_1) \oplus (B_4 \oplus A^0[4][0]) \lll 1 = 0,\\
&A^0[4][0] \oplus (A^0[3][0] \oplus A^0[3][1] \oplus B_3) \oplus (B_0 \oplus C_0) \lll 1 = 1^{64},\\
&A^0[4][4] \oplus (A^0[3][0] \oplus A^0[3][1] \oplus B_3) \oplus (B_0 \oplus C_0) \lll 1 = 1^{64}.
\end{aligned}
$$

Improved (Non)linear Structure for 3-Round Keccak-512

Details of how to satisfy conditions:

$$
\begin{aligned}
A^0[4][0] &= A^0[4][4], \\
A^0[3][1] &= A^0[3][2], \\
C_1 &= A^0[3][2] \oplus (B_4 \oplus A^0[4][0]) \lll 1 \oplus B_2. \\
A^0[1][0] &= A^0[1][4], \\
A^0[1][1] &= A^0[1][4] \oplus 1^{64}, \\
C_0 &= A^0[1][4] \oplus (B_2 \oplus C_1) \lll 1 \oplus B_0 \oplus 1^{64}. \\
A^0[3][0] &= A^0[4][4] \oplus (B_0 \oplus C_0) \lll 1 \oplus (A^0[3][1] \oplus B_3) \oplus 1^{64}.
\end{aligned}
$$

Improved (Non)linear Structure for 3-Round Keccak-384

- Based on the original linear structure
- Use conditions (1st) to slow down the diffusion
- Further guess equations (2nd round) to slow down the diffusion, i.e. making the linearization technique effective

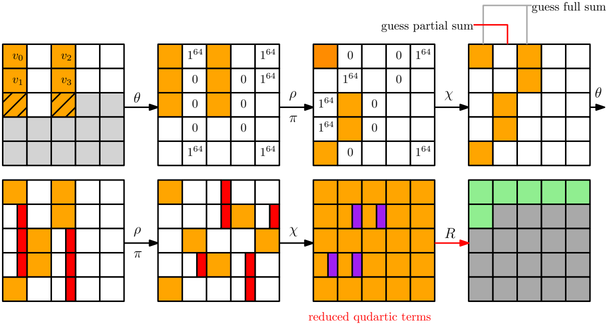


Figure: (non)linear structure of 3-Round Keccak-384

Improved (Non)linear Structure for 3-Round Keccak-384

- Optimal $t = 13$, #eqs: 461, #vars: 460
- Left degrees of freedom: 256
- The number of the 1st message block: $2^{384-256+128+2} = 2^{258}$
- Gain: $2^{64\times4-64\times2-13-2} = 2^{113}$

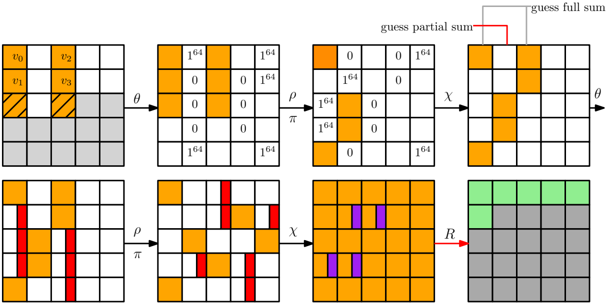


Figure: (non)linear structure of 3-Round Keccak-384

Summary of the 3-Round Attacks

- Use additional conditions (1st round) to slow down the propagation of variables
- Further guessing linear equations (2nd round) to slow down the propagation of variables
- construct overdefined quadratic equations that can be solved by the linearization technique

Preimage Attack on 4-Round Keccak-384

- Make the first 2 rounds linear (only a few variables), inspired from the conditional cube attack
- solve a system of quadratic Boolean equations
- The gain is small

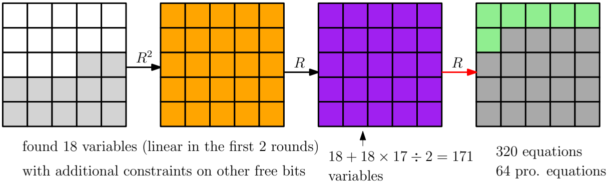
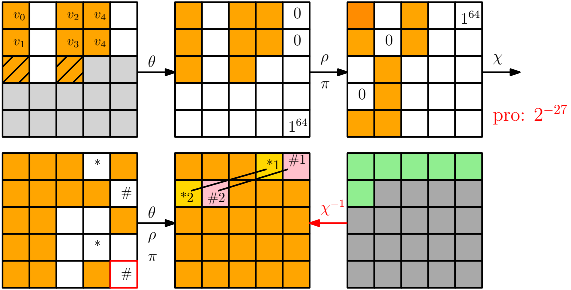


Figure: (non)linear structure of 3-Round Keccak-384

New Improvement (Others' Work)

Practical preimages for 2-round Keccak-384 (eprint 2022/788)

- Slow propagation of variables by adding conditions
- Properly choose constant bits to make $\#1 = 1^{64}$
- Properly choose constant bits to make $*2$ match the hash
- match the full 384-bit hash with only 256 variables!!!



Using Polynomial Method (EC 2021)

- 4-round Keccak-384: degree-4 eqs in 256 vars (2^{374} bit ops.)
- 4-round Keccak-512: degree-8 eqs in 512 vars (2^{502} bit ops.)

Drawback: too large memory complexity

Using Crossbred Algorithm (ICISC 2021)

- 4-round Keccak-224: overdefined quadratic eqs (2^{182} calls.)
- 4-round Keccak-256: overdefined quadratic eqs (2^{214} calls.)

based on the 2-round linear structure at EC 2019

Summary

- Slow down the propagation of variables seems important
- Nonlinear equations are better than linear ones
- Advanced techniques for nonlinear equations in Keccak?
- Construct advanced technique friendly (non)linear structures?
- Optimize the polynomial method for Keccak?