New Collision Attacks on Round-Reduced KECCAK

Kexin Qiao [1,3,4] Ling Song[1,2,3] Meicheng Liu [1] Jian Guo [2]

{qiaokexin,songling,liumeicheng}@iie.ac.cn, guojian@ntu.edu.sg

[1]SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences, China

[2]Nanyang Technological University, Singapore

[3]Data Assurance and Communication Research Center,
Chinese Academy of Sciences, China

[4]University of Chinese Academy of Sciences, China

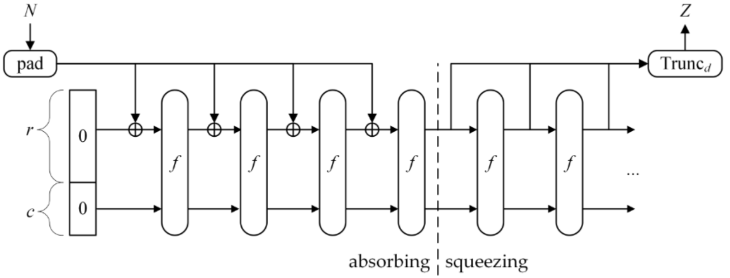
Paris, France
Eurocrypt 2017

Outlines

Outline

SHA-3 Hash Function

- Structure of Keccak–Sponge construction



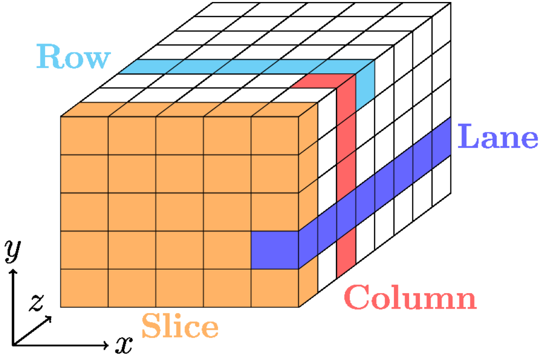
http://keccak.noekeon.org/

- Keccak-f permutation
  - 1600 bits: a 5×5 array of 64-bit lanes
  - 24 round R
  - each round consists of five steps:

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$$

SHA-3 Hash Function

KECCAK-*f* permutation: the internal state

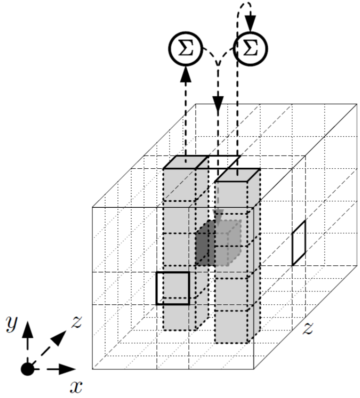


http://www.iacr.org/authors/tikz/

SHA-3 Hash Function

KECCAK permutation: $\iota \circ \chi \circ \pi \circ \rho \circ \theta$

θ step: adding two columns to current bit

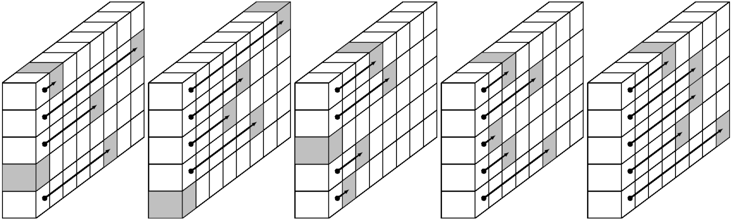


http://keccak.noekeon.org/

SHA-3 Hash Function

KECCAK permutation: $\iota \circ \chi \circ \pi \circ \rho \circ \theta$

ρ step: lane level rotations

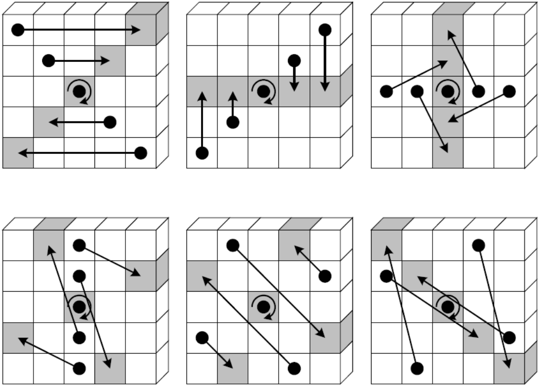


http://keccak.noekeon.org/

SHA-3 Hash Function

Keccak permutation: $\iota \circ \chi \circ \pi \circ \rho \circ \theta$

π step: permutation on lanes

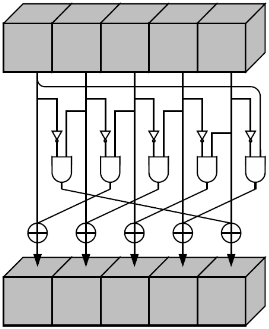


http://keccak.noekeon.org/

SHA-3 Hash Function

KECCAK permutation: $\iota \circ \chi \circ \pi \circ \rho \circ \theta$

χ step: the only nonlinear operation



http://keccak.noekeon.org/

SHA-3 Hash Function

Keccak permutation: $\iota \circ \chi \circ \pi \circ \rho \circ \theta$

ι step: adding constant

Adding one round-dependent constant to the first ”lane”, to destroy the symmetry, usually irrelevant with cryptanalysis details.

SHA-3 Hash Function

KECCAK permutation

Internal state A: a 5×5 array of 64-bit lanes

θ step $C[x] = A[x,0] \oplus A[x,1] \oplus A[x,2] \oplus A[x,3] \oplus A[x,4]$
$D[x] = C[x-1] \oplus (C[x+1] \lll 1)$
$A[x,y] = A[x,y] \oplus D[x]$

ρ step $A[x,y] = a[x,y] \lll r[x,y]$
- The constants $r[x,y]$ are the rotation offsets.

π step $B[y, 2*x+3*y] = A[x,y]$

χ step $A[x,y] = B[x,y] \oplus ((\ B[x+1,y])\&B[x+2,y])$

ι step $A[0,0] = A[0,0] \oplus RC$
- $RC[i]$ are the round constants.

The only non-linear operation is χ step.

Previous Work and Our Contribution

Collision attacks on round-reduced KECCAK

Practical Results:

- 3-round KECCAK-384 *(Dinur et al., FSE2013)*
- 3-round KECCAK-512 *(Dinur et al., FSE2013)*
- 4-round KECCAK-224 *(Dinur et al., FSE2012)*
- 4-round KECCAK-256 *(Dinur et al., FSE2012)*

Theoretical results:

- 4-round KECCAK-384: 2^{147} *(Dinur et al., FSE2013)*
- 5-round KECCAK-256: 2^{115} *(Dinur et al., FSE2013)*

Previous Work and Our Contribution

Collision attacks on round-reduced KECCAK

Practical Results:

- 3-round KECCAK-384 *(Dinur et al., FSE2013)*
- 3-round KECCAK-512 *(Dinur et al., FSE2013)*
- 4-round KECCAK-224 *(Dinur et al., FSE2012)*
- 4-round KECCAK-256 *(Dinur et al., FSE2012)*
- 5-round SHAKE128 – a member in SHA-3 *(This)*
- 5-round KECCAK$[r = 1440, c = 160, d = 160]$ *(This)*
- 5-round KECCAK$[r = 640, c = 160, d = 160]$ *(This)*

Theoretical results:

- 4-round KECCAK-384: 2^{147} *(Dinur et al., FSE2013)*
- 5-round KECCAK-256: 2^{115} *(Dinur et al., FSE2013)*
- 5-round KECCAK-224: 2^{101} *(This)*
- 6-round KECCAK$[r = 1440, c = 160, d = 160]$: $2^{70.24}$ *(This)*

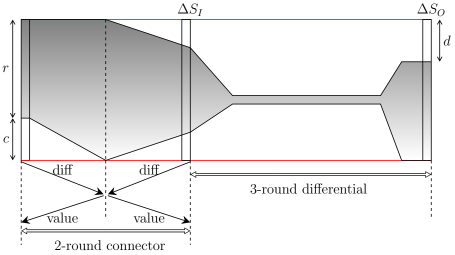
Main Idea

An extended algebraic and differential hybrid method:

1. S-box linearization in affine subspaces
2. A dedicated strategy for searching differential trails

Outline

Overview of 5-round collision attack



3-round differential: $\Delta S_I \rightarrow \Delta S_O$

2-round connector: linking ΔS_I with the initial value by linear systems

- Find (M, M')s s.t.

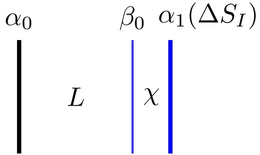
$$\mathtt{R}^2(\overline{M}\|0^c) + \mathtt{R}^2(\overline{M'}\|0^c) = \Delta S_I, \quad (\mathtt{R}^i : \text{i iterations of } \mathtt{R})$$

- E_Δ – solution is the difference of two messages
- E_M – solution space is the message/searching space

Property of Keccak S-box

1. Given $(\delta_{in}, \delta_{out})$, $V = \{x : \mathtt{S}(x) + \mathtt{S}(x + \delta_{in}) = \delta_{out}\}$ an affine subspace.
2. Given δ_{out}, $\{\delta_{in} : \mathtt{DDT}(\delta_{in}, \delta_{out}) > 0\}$ contains at least five 2-dimensional affine subspaces.

1-round connector



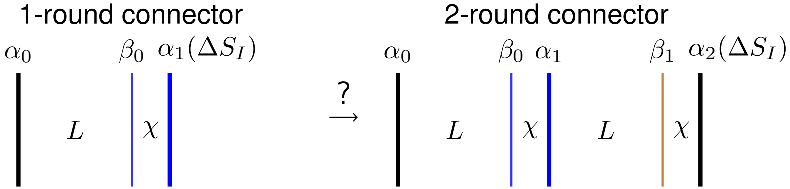
Dinur *et al.*'s target difference algorithm: find $(M, M')s$ s.t.

$$\mathrm{R}^1(\overline{M}||0^c) + \mathrm{R}^1(\overline{M'}||0^c) = \Delta S_I$$

- **Difference phase**: find exact input difference β_0 to the χ layer
  - For each active S-box, choose an affine subspace with 4 potential input differences
  - A more flexible approach
- **Value phase**: obtain the actual message pairs that lead to the target difference ΔS_I

  - Given β_0, the value phase reduces to solving linear equations.

Extension the 1-round connector to 2-round



S-box linearization

Definition (Linearizable affine subspace, LAS)

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 $\mathtt{S}(\cdot)$, $\forall x \in V, \mathtt{S}(x) = A \cdot x + b$, where A is a matrix and b is a constant vector.

Example (Linearizable affine subspace)

$V = \{00000, 00001, 00100, 00101\}$, $\mathtt{S}(V) = \{00000, 01001, 00101, 01100\}$, S-box is equivalent to linear transformation

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

Linearizable Affine Subspace and DDT

Observation (Linear Affine Subspaces in DDT)

Consider the DDT *of* KECCAK *S-box,* $V = \{x : \mathtt{S}(x) + \mathtt{S}(x + \delta_{in}) = \delta_{out}\}$

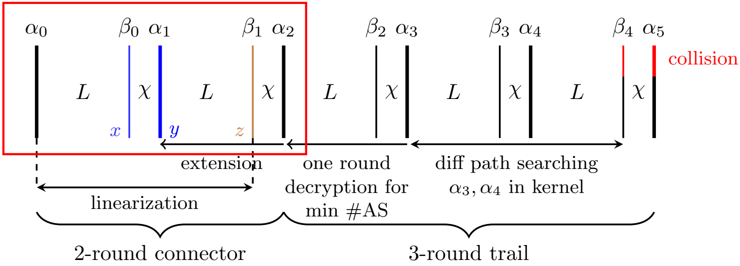
1. *if* $\mathtt{DDT}(\delta_{in}, \delta_{out}) = 2$ *or* 4*, then* V *is a linearizable affine subspace.*
2. *if* $\mathtt{DDT}(\delta_{in}, \delta_{out}) = 8$*, then there are six 2-dimensional subsets* $W_i \subset V, i = 0, 1, \cdots, 5$ *such that* $W_i (i = 0, 1, \cdots, 5)$ *are linearizable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

Example (Linear Affine Subspaces in DDT)

$\mathtt{DDT}(01, 01) = 8$, $V = \{10, 11, 14, 15, 18, 19, 1C, 1D\}$, w_i's are

$$\{10, 11, 14, 15\}, \{10, 11, 18, 19\}, \{10, 11, 1C, 1D\},$$
$$\{14, 15, 18, 19\}, \{14, 15, 1C, 1D\}, \{18, 19, 1C, 1D\}.$$

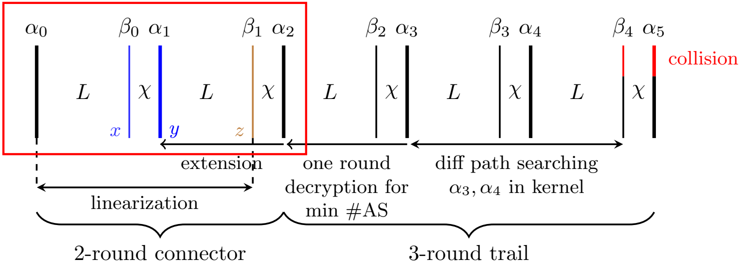
Build a 2-round connector



E_Δ and E_M are built on x variables before χ layer in the first round.

- Initialize E_Δ and E_M concerning the initial state.
- $\alpha_2(\Delta S_I) \xrightarrow{\$} \beta_1 \xrightarrow{L^{-1}} \alpha_1 \xrightarrow[\textit{by Dinur et al.}]{\textit{target difference algorithm}} \beta_0$

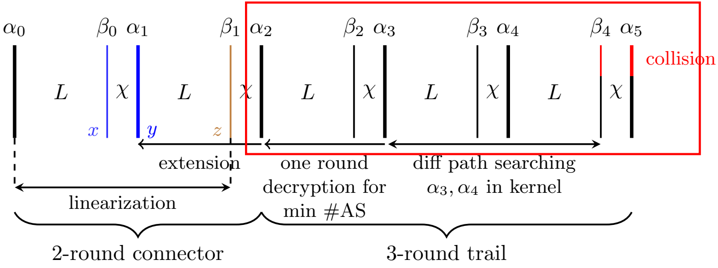
Build a 2-round connector



- Constrain x to linearizable affine subspaces by linear equations$\Rightarrow$
  - $\Pr(\beta_0 \to \alpha_1)$=1
  - y is linear to x
- Constrain z to subspaces by linear equations $\Rightarrow \Pr(\beta_1 \to \alpha_2)$=1
- Convert constrains on z to those on x
  - All are linear equation system constraints!

Outline

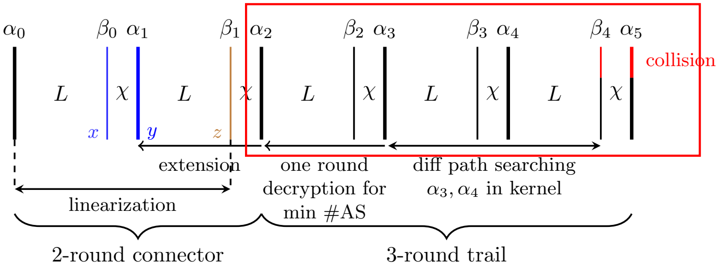
Premaries



- $\alpha_0 \xrightarrow{L} \beta_0 \xrightarrow{\chi} \alpha_1 \xrightarrow{L} \cdots \alpha_{n-1} \xrightarrow{L} \beta_{n-1} \xrightarrow{\chi} \alpha_n$.
- $w_i = w(\beta_i \to \alpha_{i+1}) = b - \log_2 |\{x : f(x) \oplus f(x \oplus \beta_i) = \alpha_{i+1}\}|$.
- n-round **trail core** $(\beta_1, \cdots, \beta_{n-1})$: a set of n-round trails

$$\alpha_0 \xrightarrow{L} \beta_0 \underset{\text{minimum weight}}{\xrightarrow{\chi}} \alpha_1 \xrightarrow{L} \beta_1 \cdots \xrightarrow{L} \beta_{n-1} \underset{\text{compatible}}{\xrightarrow{\chi}} \alpha_n$$

Requirements for differential trails



(1) $\alpha^d_{n_r} = 0$, i.e. the difference of output must be zero.

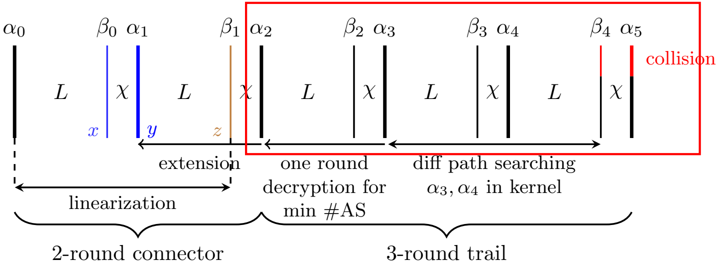
(2) $\mathtt{DF} > w_2 + \cdots + w^d_{n_r-1}$, i.e. the degree of freedom must be sufficient;

- Estimation of the degree of freedom of the 2-round connector:

$$\mathtt{DF} = \frac{b}{5} \times 2 - (c + p) - w_1.$$

(3) $w_2 + \cdots + w^d_{n_r-1} \leq 48$, the complexity for finding a collision should be low.

Search strategies



1. Search for lightweight β_3s s.t. α_3 and α_4 are in CP-kernel
2. *Forward*: Test whether there exists $\alpha_5^d = 0$ (requirement (1))
3. *Backward*: For lightweight α_3, traverse all compatible β_2. In the trail core (β_2,β_3,β_4) with lightweight α_2, check requirment (2) and (3).

Searching results

Table: Differential trail cores for KECCAK$[r, c, n_r, d]$.

No.	$r+c$	$\#AS(\alpha_2\text{-}\beta_2\text{-}\beta_3\text{-}\beta_4^d)$	$w_1\text{-}w_2\text{-}w_3\text{-}w_4^d$	d
1	1600	102-8-8-2	240-19-16-4	256
2	1600	88-8-7-0	195-21-15-0	256
3	1600	85-9-10-2	190-25-20-3	224
4	800	38-8-8-0	85-20-16-0	160
No.	$r+c$	$\#AS(\alpha_2\text{-}\beta_2\text{-}\beta_3\text{-}\beta_4\text{-}\beta_5^d)$	$w_1\text{-}w_2\text{-}w_3\text{-}w_4\text{-}w_5^d$	d
5	1600	145-6-6-10-14	340-15-12-22-23	160

Outline

Summary of Attacks on KECCAK

Table: Collision attack results

Target $[r, c, d]$	n_r	Searching Complexity	Degree of freedom	Searching Time	Solving Time[2]
SHAKE128	5	2^{39}	94	30 min	25 min
KECCAK[1440,160,160]	5	2^{40}	162	2.48 hr	9.6 sec
	6	$2^{70.24}$	135	N.A. [1]	1 hr
KECCAK[640,160,160]	5	2^{35}	56	2.67 hr	30 min
KECCAK-224	5	2^{101}	11/2/3	N.A.	N.A.

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

Outline

Future work

1. 3-round connectors
   – Practical 6-round collisions on a challenge version have already been found

Future work

1. 3-round connectors
   – Practical 6-round collisions on a challenge version have already been found
2. The S-box linearization can be viewed as a "row-level" linear approximation.
   – Linear cryptanalysis: bit-level linear approximation
   Does linearization on alternative levels exist and how to find them?

Future work

1. 3-round connectors
   – Practical 6-round collisions on a challenge version have already been found
2. The S-box linearization can be viewed as a "row-level" linear approximation.
   – Linear cryptanalysis: bit-level linear approximation
   Does linearization on alternative levels exist and how to find them?
3. Will system of higher degree work? Systems of degree 2 can also be applied to build connectors.

Thanks for your attention.