SymSum: Symmetric-Sum Distinguishers Against Round Reduced SHA3

Dhiman Saha[1], Sukhendu Kuila[2], Dipanwita Roy Chowdhury[1]

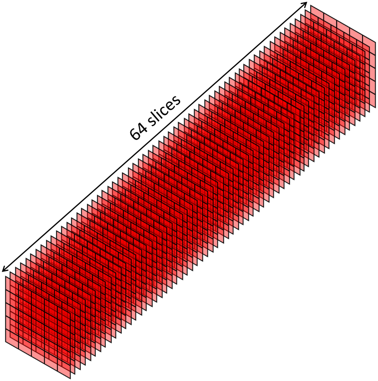
[1]Crypto Research Lab
Department of Computer Science & Engineering,
IIT Kharagpur, India
{dhimans,drc}@cse.iitkgp.ernet.in

[2]Department of Mathematics
Vidyasagar University, India
babu.sukhendu@gmail.com

FSE 2017

Tokyo, Japan

- Follows SPONGE construction
- Internal permutation called KECCAK-f/KECCAK-p
- Internal state
  - Array of 5×5 slices
  - Biggest size $\rightarrow$ 1600 bits
- Total 24 rounds
- 1 Round $=$ 5 sub-operations



$$\mathcal{R} = \iota \circ \chi \circ \pi \circ \rho \circ \theta$$

Note: Position of ι in the round function

Round-constants added at the end of a round

- SHA3 Family

$$\texttt{Fixed-Length} \rightarrow \texttt{SHA3-224/256/384/512}$$
$$\texttt{XOF} \rightarrow \texttt{SHAKE128/256}$$

- Main difference with Keccak Family:
  - Introduction of the domain separation bits prior to 10*1 padding

$$M \xrightarrow{\texttt{Add Suffix}} \begin{cases} M||01 & \texttt{Fixed-Length} \\ M||1111 & \texttt{XOF} \end{cases}$$

Distinguishing Attacks on KECCAK-f

Towards exhibiting non-random behaviour

Distinguishers on KECCAK-f

Target the Hermetic Sponge Strategy

Internal permutation of Sponge based hash function should be designed such that they **cannot be distinguished** from a randomly-chosen permutation.

- Maximum results on Keccak-f during SHA-3 competition
  - e.g., Zero-Sum, Rotational among others

Particular Attention — Zero-Sum Distinguisher

- Based on higher-order derivatives of forward/inverse rounds
- Only distinguisher to reach full 24-rounds
- Uses *inside-out* strategy

What about distinguishers on KECCAK?

Distinguishing the hash-function itself

Distinguishers on KECCAK

Distinguishers on KECCAK-f **may not** directly extend to KECCAK

- Due to restrictions imposed by SPONGE
- e.g. Zero-Sum applies
  - But **looses** number of penetrable rounds
  - Inside-out technique invalidated

Few results on distinguishers on KECCAK hash function

- 4-round KECCAK
  - Due to Naya-Plasencia, Röck, and Meier
  - Using low weight differential path
  - Complexity: 2^{24}
- 6-round KECCAK
  - Due to Das and Meier
  - Based on biased output bits
  - Complexity: 2^{52}

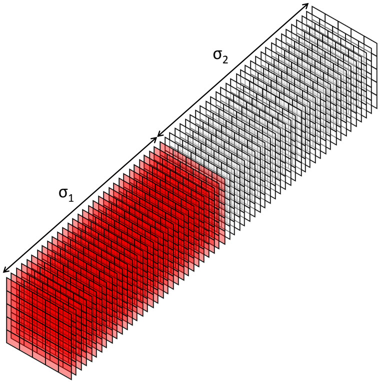
An Experiment on SHA3

Based on self-symmetry

- A **restriction** on the internal state of KECCAK-f
- 1600-bit State ($\mathcal{S}$) visualized as two 800-bit **Substates** (σ_1, σ_2)

$$\mathcal{S} = \sigma_1 || \sigma_2$$

- $\sigma_i = 5 \times 5 \times 32$ bits



The Restriction: Equal Substates

$$\sigma_1 = \sigma_2$$

- A self-symmetric state
- Represented in standard *lane* $\times$ *sheet* format
- Look at individual lanes
- The first Substate is highlighted

0934258C0934258C
24B83B0524B83B05
74F1384174F13841
8030F1308030F130
36FB572A36FB572A

62C05E2462C05E24	0934258C0934258C	49DA0D3D49DA0D3D	2923A54B2923A54B	8817062C8817062C
B6C808B2B6C808B2	24B83B0524B83B05	2026890020268900	738E1141738E1141	3886D76A3886D76A
94BA023194BA0231	74F1384174F13841	ADE17841ADE17841	411E023D411E023D	98C34C6798C34C67
64010A3264010A32	8030F1308030F130	E383F57AE383F57A	35388C8235388C82	61F7231161F72311
68DD183C68DD183C	36FB572A36FB572A	120A313A120A313A	1C6E105D1C6E105D	B50D7CA2B50D7CA2

Pad(AddSuffix(Message)) $\rightarrow$ Self-Symmetric Internal State

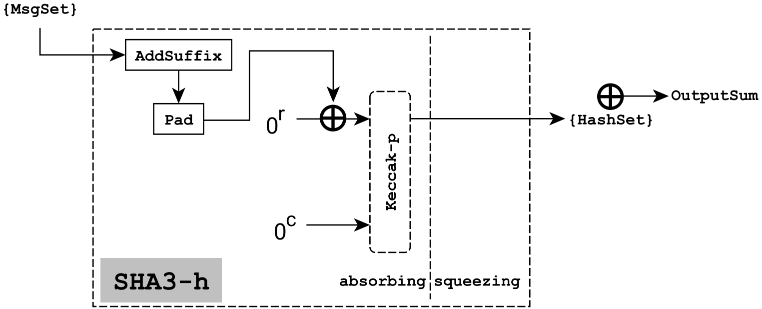
8cd812d28cd812d2
****0*9b****0*9b
0000000000000000

- Single block messages
- Similar to ZeroSum computation
- But with additional **restriction** of preserving symmetry
- By construction, $\bigoplus_{\texttt{Msg}\in\texttt{MsgSet}} \texttt{Msg} = \mathbf{0}$

4a36ea584a36ea58	8cd812d28cd812d2	88e61fc788e61fc7	f3372eaff3372eaf	ea3f0b51ea3f0b51
ce168c02ce168c02	****0*9b****0*9b	b934cb9fb934cb9f	866ac262866ac262	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000	0000000000000000	0000000000000000

Zeros at end indicate value of capacity bits

- Run SHA3 (Round-Reduced) over the Message Set
- Compute Output-Sum



What is the nature of the Output-Sum?

|MsgSet|	Output-Sum	Remark
2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum

|MsgSet|	Output-Sum	Remark
2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{16}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum

The Output-Sum

|MsgSet|	Output-Sum	Remark
2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{16}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{15}	000001000000010000000000000000000000200000002000 000000000000000000000000000000000000000000000000 00000000000000000000004000000040	Symmetric-Sum

|MsgSet|	Output-Sum	Remark
2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{16}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{15}	000001000000010000000000000000000000200000002000 000000000000000000000000000000000000000000000000 00000000000000000000004000000040	Symmetric-Sum
2^{14}	243f4942243f4942528c98d5528c98d57300b0d17300b0d1 c0585999c0585999147b20a3147b20a3083a3900083a3900 09225588092255886302671c6302671c	Symmetric-Sum

|MsgSet|	Output-Sum	Remark
2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{16}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{15}	000001000000010000000000000000000000200000002000 000000000000000000000000000000000000000000000000 00000000000000000000004000000040	Symmetric-Sum
2^{14}	243f4942243f4942528c98d5528c98d57300b0d17300b0d1 c0585999c0585999147b20a3147b20a3083a3900083a3900 09225588092255886302671c6302671c	Symmetric-Sum
2^{13}	81ed3fca81ed3dca15553dac15553dec25858e1125858e11 11c9af8b11c9af8b509927bf5099273f9276901992679019 ca92a3d5ca9223d54ffce7974ffc6797	Not Symmetric

The Output-Sum

|MsgSet|	Output-Sum	Remark
2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{16}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{15}	000001000000010000000000000000000000200000002000 000000000000000000000000000000000000000000000000 00000000000000000000004000000040	Symmetric-Sum
2^{14}	243f4942243f4942528c98d5528c98d57300b0d17300b0d1 c0585999c0585999147b20a3147b20a3083a3900083a3900 09225588092255886302671c6302671c	Symmetric-Sum
2^{13}	81ed3fca81ed3dca15553dac15553dec25858e1125858e11 11c9af8b11c9af8b509927bf5099273f9276901992679019 ca92a3d5ca9223d54ffce7974ffc6797	Not Symmetric
2^{12}	78f523d01479a153802f16a4c8bbb67116d502ea0495823a 71057dfbf18b25f22bba947d0ba094fd1240ee380a42df38 99eaa56698fa64e6a21ac1328138c126	Not Symmetric

What to make of these results?

- Results
  - Partly intuitive
  - Partly inexplicable
  - Definitely worth investigating (Our Motivation)

First Question

What is the underlying operator in the experiment?

Intuition

We must be computing some kind of higher order derivative.

- But not simple higher order derivatives (as in case of classical Zero-Sum)
  - Recall: Multiple variables change values per call
  - Also, the self-symmetry constraint

So, What is the underlying operator?

Answer: $m-$fold vectorial derivatives[1]

- **Slightly different notion of higher-order derivatives**
- Analogous to computing derivatives over a subspace
- Partitions the inputs variables

$$\text{The Experiment} \equiv \begin{array}{c}\text{Computing } m-\text{fold vectorial derivatives}\\ \text{with specially selected subspaces}\end{array}$$

Specially selected subspace $\rightarrow$ Self-Symmetry constraint

[1] Refer paper for mathematical form

Why do we witness ZeroSum?

2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{16}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum

The Experiment

Corresponds to computing $17, 16, 15, 14, 13-$fold vectorial derivatives of `SHA3-512` reduced to 4-rounds.

2^{17}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum
2^{16}	000000000000000000000000000000000000000000000000 000000000000000000000000000000000000000000000000 00000000000000000000000000000000	Zero-Sum

- Note: deg 4-Round SHA3-512 ≤ 16
  - So computing the $17-$fold vectorial derivative leads to a ZeroSum
  - For $16-$fold case, highest degree could not be reached due to choice of constant partitions

Why do we witness **symmetry** in the Output-Sum?

2^{15}	0000010000000100000000000000000000020000000200 0 000000000000000000000000000000000000000000000000 0000000000000000000000400000040	Symmetric-Sum
2^{14}	243f4942243f4942528c98d5528c98d57300b0d17300b0d1 c0585999c0585999147b20a3147b20a3083a3900083a3900 092255880922558863026 71c6302671c	Symmetric-Sum

Lemma

For an iterated SPN round function ($\mathcal{G}$) if the ordering of the component transformations is such that the non-linear operation **precedes** *the round constant addition, then highest-degree monomials are "***not affected***" by round-constants.*

$$\begin{aligned}\mathcal{G}^q &= (\mathcal{C}_q \circ \mathcal{N} \circ \mathcal{L}) \circ (\mathcal{C}_{q-1} \circ \mathcal{N} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_2 \circ \mathcal{N} \circ \mathcal{L}) \circ (\mathcal{C}_1 \circ \mathcal{N} \circ \mathcal{L}) \\ &= \Big[((\mathcal{C}_q \circ \mathcal{N} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_2 \circ \mathcal{N} \circ \mathcal{L})) \circ \mathcal{C}_1\Big] \circ (\mathcal{N} \circ \mathcal{L}) \qquad (1)\end{aligned}$$

Intuition

Notice effect of the first round non-linear operation

- Segregate monomials in ANF based on dependence on round-constants

Example

$$
\begin{aligned}
f &= x_1x_2x_3 + c_1c_2x_2x_3 + x_3x_4 + c_2c_3 \\
&= (x_1x_2x_3 + x_3x_4) + (c_1c_2x_2x_3 + c_2c_3) \\
&= f_s + f_{s'}
\end{aligned}
$$

- Show difference in highest-degree attained

What does this mean for SHA3?

Corollary

For q rounds of the SHA3 permutation $\textsc{Keccak}$-p*, the maximum degree of a monomial involving a round-constant is* $d^{\circ}\mathcal{K}^{q} - 2$

- Recall the sequence of operations in Keccak-f

$$\mathcal{R} = \iota \circ \chi \circ \pi \circ \rho \circ \theta$$

- Note ι after χ, the non-linear operation
- First round χ has no effect on terms involving round-constants.
- Note: $\deg \chi = 2$

Corollary

For q rounds of KECCAK-p *the* $(d^{\circ}\mathcal{K}^{q} - 1)-$*fold vectorial derivative is a* **round-constant independent function**.

- Recall ι is the only operation that breaks symmetry
- And θ, ρ, π, χ are translation invariant in the z-axis

Implication

A Round-Constant Independent Function

$$\Longrightarrow$$

A Translation Invariant Function

The SymSum Proposition

Proposition

The $(d^\circ SHA3 - 1)-$fold vectorial derivative of SHA3 evaluated using **only self-symmetric input states** *will preserve the symmetric property.*

- Explains the symmetry in the Output-Sum

2^{15}	0000010000000100000000000000000000002000000020000 00000000000000000000000000000000000000000000000000 000000000000000000000040000000040	Symmetric-Sum
2^{14}	243f4942243f4942528c98d5528c98d57300b0d17300b0d1 c0585999c0585999147b20a3147b20a3083a3900083a3900 092255880922558863026 71c6302671c	Symmetric-Sum

- Recall: Highest degree attained for this particular case was < 16

SymSum:
A new distinguishing property for SHA3

Definition (Symmetric Sum (SymSum))

Let us consider the SHA3 fixed-length hash functions SHA3−h : $(\mathbb{F}_2^r)^* \rightarrow \mathbb{F}_2^h$ *or XOFs SHAKE128/256* : $(\mathbb{F}_2^r)^* \rightarrow \mathbb{F}_2^*$. *A* **Symmetric Sum** *or SymSum is defined as a set of inputs* $\{x_1, x_2, \cdots, x_k\} \in \mathbb{F}_2^r$ *for which the input-sum is* **zero** *while the* **64-prefix** *of the output-sum is* **symmetric**.

Step 1: Compute $(d^\circ\mathtt{SHA3} - 1)-$fold vectorial derivative of SHA3 by generating self-symmetric input states

Step 2: Check for the SymSum property in the Output-Sum

SymSum Advantage $h =$ hash-length

$$\mathrm{Adv}_{\mathtt{SymSum}} = 1 - 2^{-32\times\left\lfloor\frac{h}{64}\right\rfloor} \approx 1$$

ZeroSum Vs SymSum

	Degrees of freedom			Degrees of freedom	
SHA3 variant Fixed-Length	ZeroSum (2^{r-4})	SymSum $(2^{\frac{r-8}{2}})$	SHA3 variant XOFs	ZeroSum (2^{r-6})	SymSum $(2^{\frac{r-12}{2}})$
SHA3-224	2^{1148}	2^{572}	SHAKE-128	2^{1338}	2^{666}
SHA3-256	2^{1084}	2^{540}			
SHA3-384	2^{828}	2^{412}	SHAKE-256	2^{1082}	2^{538}
SHA3-512	2^{572}	2^{284}			

- SymSum **looses** degrees of freedom

Does this have an adverse effect on its performance?

Actually, No (See next slide)

Comparison with ZeroSum

		Complexity	
#Rounds (n_r)	Bound on d°SHA3	ZeroSum ($2^{d^\circ \texttt{SHA3}+1}$)	SymSum ($2^{d^\circ \texttt{SHA3}-1}$)
1	2	2^3	2^1
2	4	2^5	2^3
3	8	2^9	2^7
4	16	2^{17}	2^{15}
5	32	2^{33}	2^{31}
6	64	2^{65}	2^{63}
7	128	2^{129}	2^{127}
8	256	2^{257}	2^{255}
9	512	2^{513}	$2^{511\dagger}$
10	1024	$2^{1025\dagger}$	$\star$
11	1408 (*Boura et al.*)	$\star$	$\star$

† Not applicable for SHA3-512 and SHA3-384

⋆ Exceeds degrees of freedom

Epilogue

- We investigated an interesting symmetric property exhibited by the sum of SHA3 message digests

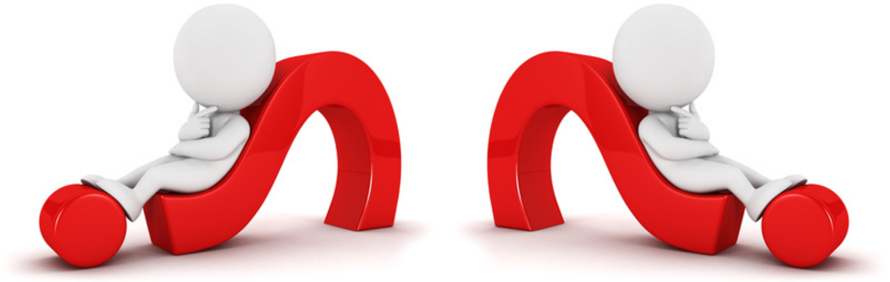
- Put forward a mathematical framework to explain the property
  - A operator that tries to select a specific subspace over which it computes higher order derivatives
  - A relation that estimates the degree of round-constant dependent terms in ANF for SPN based functions.

- Capitalizing on this a new distinguisher SymSum is proposed
  - Has high distinguishing advantage
  - Better that ZeroSum by a factor of four

- First property that relies on round-constants but independent of their Hamming-weights

Thanks!

Related info on `http://de.ci.phe.red` shortly.



Queries

`crypto@dhimans.in`