IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2017, No. 1, pp. 240–258. DOI:10.13154/tosc.v2017.i1.240-258

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 and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, India
{dhimans,drc}@cse.iitkgp.ernet.in
[2] Department of Mathematics, Vidyasagar University, Medinipur, India
babu.sukhendu@gmail.com

Abstract. In this work we show the existence of special sets of inputs for which the sum of the images under SHA3 exhibits a symmetric property. We develop an analytical framework which accounts for the existence of these sets. The framework constitutes identification of a generic property of iterated SPN based functions pertaining to the round-constant addition and combining it with the notion of $m-$fold vectorial derivatives for differentiation over specially selected subspaces. Based on this we propose a new distinguisher called SymSum for the SHA3 family which penetrates up to 9 rounds and outperforms the ZeroSum distinguisher by a factor of four. Interestingly, the current work is the first analysis of SHA3/Keccak that relies on round-constants but is independent of their Hamming-weights.

Keywords: distinguisher · Keccak · SHA3 · hash functions · cryptanalysis · zero-sums · self-symmetry · vectorial derivatives

1 Introduction

Keccak, the winner of the SHA3 competition [oST], has been formally standardized in August 2015 and published in FIPS 202 [Nat15]. In course of the SHA3 competition, there have been a lot of cryptanalytic results on both the internal permutation Keccak-f (denoted as Keccak-p in FIPS 202) of Keccak and also over the complete hash-function. While the main findings over Keccak concentrated on pre-images and collisions, Keccak-f received considerable attention from what is known as *distinguishing* attacks. These attacks, which aim at exhibiting non-random behavior by targeting the *hermetic sponge strategy* [BDPA07], employ various techniques like constrained-input contrained-output (CICO) [AK09], rotational symmetry [MPS12], self-symmetry [KSPC14], ZeroSum [AM09], [DL12, BC10a, BC10b], rebound [DGPW12], cube-tester [DMP+14] and recently reported internal differential boomerang [JN15]. However, among these, only the ZeroSum distinguisher is able to distinguish all 24-rounds of Keccak-f. It was first reported by Aumasson and Meier [AM09] as the *zero-sum* distinguishing property[1] for public permutations. For a given function F, the basic aim of the ZeroSum distinguisher is to show the existence of sets of input states which sum up to zero and whose images under F also sum up to zero. The generation of many such sets of inputs with an effort less than the generic complexity can be interpreted as a distinguishing property of F.

In their initial result at CHES 2009 rump session, Aumasson and Meier reported [AK09] that using the *inside-out* technique for a public permutation, zero-sums could

[1] It is also argued to be a generalization of the integral property

Received: 2016-06-01, Revised: 2016-11-23, Accepted: 2017-01-24, Published: 2017-03-08

be constructed starting from the middle. This enabled them to devise distinguishers reaching up to 16 rounds. The basic idea is of computing higher order derivatives of both forward and inverse rounds of KECCAK-f and is hence directly related to the algebraic degree. Thus better estimates for the degree of a function (and its inverse) after several rounds can lead to smaller zero-sum partitions. However, doing so precisely is believed to be a hard problem. Hence, all follow up results on ZeroSum distinguishers have tried to provide tighter bounds on the degree thereby reaching higher number of rounds and/or reducing the size of the zero-sums. The first improvement [BC10a] was shown by Boura and Canteaut who extended it to 18 rounds. Their main contribution was the analysis of the Walsh spectrum of the non-linear part (χ) leading to better bounds on degree of the rounds. In addition to that they also presented an elegant idea of one-round extension by bypassing a round in the middle by specifying input subspaces which remain same after a non-linear layer of the internal permutation. The authors call this property as multi-set property. In SAC 2010, the same authors extended their previous work to obtain multi-set property for two consecutive rounds [BC10b] that helped them penetrate up to 20 rounds of KECCAK-f. Next, Boura et al. [BCC10] presented new upper bounds on the algebraic degree of iterated permutations thereby reaching full 24-rounds of KECCAK-f with a complexity of 2^{1590}. Their main result consisted of the observation that the algebraic degree, for larger number of rounds, does not increase exponentially with the number of rounds. Duan and Lai [DL11] reduced the complexity to 2^{1579} by identifying that the product of any two output bits of inverse of χ, whose algebraic degree is 3, also has a degree 3. Finally, in FSE 2011, Boura et al. [BCC11] used the result of Duan and Lai to give the best estimates on the degree of KECCAK-f inverse resulting in construction of ZeroSum structure for full KECCAK in 2^{1575} calls to the permutation. This was also independently reported by Duan and Lai in [DL12]. An important aspect of the above results is that they **do not translate** into a distinguisher on the KECCAK/SHA3 hash function due the constraints imposed by the SPONGE mode of operation. Consequently, there have been only a few works on distinguishers that target the KECCAK hash function. New strategies have been developed in [NRM11] to construct low weight differential path. Based on this path, distinguisher on KECCAK reduced to 4 rounds have been devised which works with complexity 2^{24}. In [DM14], the authors study the propagation of difference in KECCAK and looks for biased output bits to devise a distinguisher for KECCAK reduced to 6 rounds with a complexity 2^{52}. On a final note, ZeroSum, though still applicable on KECCAK/SHA3, looses the number of penetrable rounds since the inside-out trick is invalidated.

While discussing higher order differential cryptanalysis, it is worth mentioning the closely related area of algebraic attacks like *Cube Attacks* which exploit the low algebraic degree of the round function and have been applied [Lat09] in the context of hash functions. In Eurocrypt 2015 Dinur et al. [DMP+15] augmented cube attacks with related algebraic techniques using structural properties of KECCAK-f to measure the security of keyed-SPONGE. In comparison to the classical cube attack, the modified approach (with the inversion of last χ layer when the output is sufficiently large) leads to a key recovery attack on larger number of rounds. Further one more round is penetrated by setting the cube variables in the column parity kernel in order to control the effect of θ such that the cube variables do not get multiplied in the subsequent χ layer. MAC forgery for 9 rounds also have been mounted with the help of cube tester which distinguishes the output instead of recovering underlying the key. In order to reduce the cube dimension further, some conditions have been derived in [HWX+16] by proposing conditional cube tester. While the previous attacks focused on the minimization of algebraic degree by controlling only the first round, it tries to control the propagation of cube variables for first 2 rounds by adoption of bit conditions. The paper reports key-recovery attacks on 5,6 and 7 round KECCAK-MAC with time complexity $2^{24}, 2^{40}$ and 2^{72} respectively.

In this work, we formulate an experiment on SHA3 using messages encoded in a way that maintains a certain kind of symmetry. We first show that the XOR of the digests of SHA3 over appropriate number of such messages posses a symmetric property. The work evolves around the justification of the results of this experiment. We first formalize the experiment by using a mathematical operator for higher order Boolean derivatives over *special subspaces*. Next, we establish a generic property related to round-constants and the algebraic degree of any SPN based round function. Later, we combine the arguments and adapt them in the context of members of SHA3 family. Finally, we justify the results of the experiment and propose a new distinguisher for SHA3 referred to as SymSum. Interestingly, SymSum can perform better than ZeroSum by a constant factor.

The rest of paper is organized as follows. Notations used in the work are introduced in Section 2. Section 3 illustrates an interesting experiment on SHA3. The concept of $m-$fold vectorial derivatives is proposed in relation to the previous experiment in Section 4. A property of SPN based round function is identified and justified in Section 5 which points in the direction of getting round-constant independent functions. The proposed SymSum distinguisher is devised in Section 6 which also gives a comparative study with the classical ZeroSum technique. The concluding remarks are furnished in Section 7.

2 Notations

The preliminaries on SHA3 as defined in FIPS 202 are furnished in Appendix A. The value of rate of SPONGE for a particular SHA3 variant is denoted by r while the capacity is represented by c. Since, for SHA3, the size of the permutation is always $b = 1600$, we use the following notation to refer to q rounds of KECCAK-p permutation:

$$\mathcal{K}^q \equiv \text{KECCAK-}p[1600, q]$$

The internal state of $\mathcal{K}^q$, denoted by $\mathcal{S}$, is visualized as a collection of 64 ordered *slices*[2], 64 being the lane-size. We next introduce concept of a substate which divides a state into two equal halves.

Definition 1 (Substate)**.** *A* ***Substate*** (σ) *is a collection of either the first or the last 32 slices of a state. Thus any state has two sub-states.*

$$\mathcal{S} = \sigma_1 || \sigma_2, \text{ where } \sigma_i \in \{0,1\}^{25\times 32}$$

Definition 2 (Self-Symmetry)**.** *If $\mathcal{S}^{\#}$ be the set of all states such that $\sigma_1 = \sigma_2$, then every member of $\mathcal{S}^{\#}$ is called Self-Symmetric.*

$$\mathcal{S} \in \mathcal{S}^{\#} \text{ when } \sigma_1 = \sigma_2$$

It follows from Definition 1 and 2 that $|\mathcal{S}^{\#}| = 2^{800}$. Table 1 presents an example of a Self-Symmetric state.

Table 1: A Self-Symmetric state. σ_1 is highligted.

62C05E24	62C05E24	0934258C	0934258C	49DA0D3D	49DA0D3D	2923A54B	2923A54B	8817062C	8817062C
B6C808B2	B6C808B2	24B83B05	24B83B05	20268900	20268900	738E1141	738E1141	3886D76A	3886D76A
94BA0231	94BA0231	74F13841	74F13841	ADE17841	ADE17841	411E023D	411E023D	98C34C67	98C34C67
64010A32	64010A32	8030F130	8030F130	E383F57A	E383F57A	35388C82	35388C82	61F72311	61F72311
68DD183C	68DD183C	36FB572A	36FB572A	120A313A	120A313A	1C6E105D	1C6E105D	B50D7CA2	B50D7CA2

[2]The definition of a slice is consistent with the KECCAK submission document and is defined as a 5×5 matrix of bits.

Definition 3 (β-prefix)**.** *The β-prefix (s_β) of a string $s \in \{0,1\}^*$ is the prefix of length*[3] $|s_\beta| = |s| - (|s| \bmod \beta)$.

$$\begin{aligned} s_\beta &= (b_1 b_2 \cdots b_\beta)||(b_{\beta+1} b_{\beta+2} \cdots b_{2\beta})|| \cdots ||(b_{(n-1)\beta+1} b_{(n-1)\beta+2} \cdots b_{n\beta}) \\ &= B_1||B_2|| \cdots ||B_n \end{aligned}$$

To be precise, if $(|s| \bmod \beta) = 0$, then $s_\beta = s$.

Definition 4. *The β-prefix (s_β) of a string (s) is said to be symmetric ($s^\#_\beta$) if the internal difference of all its β-bit substrings is zero.*

$$\forall B_i \in s_\beta \rightarrow [b_{(i-1)\beta+1} \cdots b_{(i-1)\beta+32}] \oplus [b_{(i-1)\beta+33} \cdots b_{i\beta}] = 0$$

An example of the symmetric 64-prefix of a string is given below in `HEX` format:

$\mathtt{s}$ =9af8dfef9af8dfeffe1135ebfe1135eb3b0a9daf3b0a9daf23d3a55823d3a558
dcb229fedcb229feb724c696b724c696cf032a7c

$\mathtt{s}^\#_{64}$ =9af8dfef9af8dfeffe1135ebfe1135eb3b0a9daf3b0a9daf23d3a55823d3a558
dcb229fedcb229feb724c696b724c696

3 An Experiment on Round-Reduced SHA3

Here we illustrate an experiment on `SHA3` by manipulating single block messages such that the input state of Keccak-p is *Self-Symmetric*. We assume that we are dealing with `SHA3-512` but the experiment can be mounted on any member of the `SHA3` family. The experiment is briefed below. A visual illustration is depicted in Figure 1 while the actual results of experiment on `SHA3-512` are furnished next.

- `MsgSet Formation:` The messages in the `MsgSet` have the following properties. The details of constructing such sets of messages are given in Appendix B.

$$\forall \mathtt{Msg} \in \mathtt{MsgSet}, \left[\mathtt{Pad(AddSuffix(Msg))}||0^c\right] \in \mathcal{S}^\#$$

$$\bigoplus_{\mathtt{Msg} \in \mathtt{MsgSet}} \mathtt{Msg} = \mathbf{0}$$

- `Output-Sum Computation`: The second step in the experiment is to use these messages in the `MsgSet` and run `SHA3-512` with reduced-round Keccak-p on them to produce the respective hash values[4]. These values constitute the `HashSet`. The next step is to compute the *symmetric-difference* over the `HashSet`.

$$\mathtt{Output\text{-}Sum} = \bigoplus_{\mathtt{Msg} \in \mathtt{MsgSet}} \mathtt{SHA3\text{-}512(Msg,\ \#Rounds)}$$

- `Self-Symmetry Verification`: The last step is to verify if the 64-prefix of the `Output-Sum` is symmetric.

$$\mathtt{Output\text{-}Sum}_{64} \xrightarrow{\mathtt{Verify}} \mathtt{Output\text{-}Sum}^\#_{64}$$

[3] It is understood that $|s| \geq \beta$

[4] This step is similar for any `SHA3` fixed length variant. For, the XOFs we truncate the hash value to r if desired hash-length $> r$

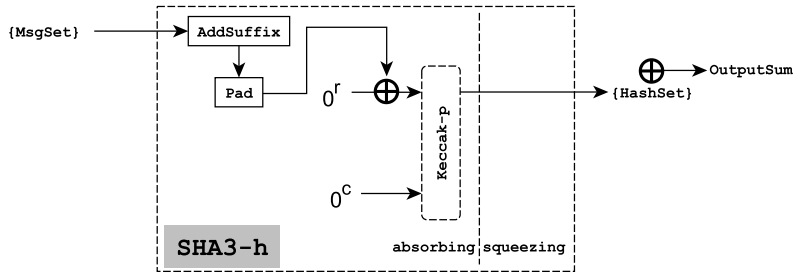

Figure 1: A generic view of the proposed experiment

3.1 Results on 4-Round SHA3-512

Based on the strategy described in the previous section MsgSet of various sizes were created (For details refer to Appendix C) and the Output-Sum was computed after running SHA3-512 (Source code from [Bau]) on every Msg of each such MsgSet. Table 2 captures the results which exhibits that the Output-Sum has some interesting phenomena.

Table 2: Output-Sum exhibiting self-symmetric property

|MsgSet|	Output-Sum	Remark
2^{17}	0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000	Zero-Sum
2^{16}	0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000	Zero-Sum
2^{15}	0000010000000100000000000000000000002000000020000000000000000000000000000000000000000000000000000000000000000000000000000000000000040	Symmetric-Sum
2^{14}	243f4942243f4942528c98d5528c98d57300b0d17300b0d1c0585999c0585999147b20a3147b20a3083a3900083a390009225588092255886302671c6302671c	Symmetric-Sum
2^{13}	81ed3fca81ed3dca15553dac15553dec25858e1125858e1111c9af8b11c9af8b509927bf5099273f9276901992679019ca92a3d5ca9223d54ffce7974ffc6797	Not Symmetric
2^{12}	78f523d01479a153802f16a4c8bbb67116d502ea0495823a71057dfbf18b25f22bba947d0ba094fd1240ee380a42df3899eaa56698fa64e6a21ac1328138c126	Not Symmetric

3.2 Interpretation of the Results

Zero-Sum: It can be seen that for 2^{17} and 2^{16} the Output-Sum is all zeros. This points us in the direction of the classical *ZeroSum* property which is related to degree of a function and is based on finding higher-order Boolean derivatives. However, the way we are generating symmetric inputs differs from the way a classical *ZeroSum* is computed. *So the question arises why do we still witness the* ZeroSum *property?*

Symmetric-Sum: The results suggest that for 2^{15} and 2^{14} the `Output-Sum` bears an even more interesting property: *Self-Symmetry*. Existing literature does not seem to account for this property. Hence, investigating this property and establishing theoretical arguments form the main motivations of the current work.

Non-Symmetric-Sum: It is apparent that below a certain size of the `MsgSet` the `Output-Sum` no longer shows a symmetric property. *So, is there a relation between the degree of the function and the symmetric property?*

Repeating the experiments for various base messages with different sized message sets have lead us into believing that there exists a relation between the size of the message set and the nature of the `Output-Sum`: Zero/Symmetric/Non-Symmetric. Moreover, the size of the message set giving a zero `Output-Sum` is strictly greater than the size of the set that gives a symmetric `Output-Sum`. Coupled with the fact that **the probability of an ideal hash function exhibiting such a symmetric property** is exponentially low, Symmetric-Sum property shows prospect of devising a distinguisher for the `SHA3` family with very high advantage. The contribution of the current work can be summarized as below:

Our Contribution

- Formalize the experiment mentioned above by using an operator based on classical higher order derivatives of Boolean functions and justify the reason behind getting ZeroSums.
- Establish a generic property of a class of SPN based iterated round functions by analyzing the effect of the position of round constant addition on the overall algebraic degree of the function.
- Analyze `SHA3` in the light of the property to account for the self-symmetry exhibited by the `Output-Sum`
- Propose a new distinguisher for `SHA3` variants which is better that the classical ZeroSum distinguisher by a constant factor.

Our first task is to formalize the mathematical operator in action behind the experiment. We start by using a slightly different notion of higher-order derivatives which we refer to as $m-$fold vectorial derivatives of a Boolean function. All concepts dealt with are easily extended to vectorial functions.

4 The Operator: $m-$fold Vectorial Derivatives

The operator that we put forward uses a different notion of higher-order derivatives and is analogous to computing derivatives over a subspace. The real intuition is to show that the operation it performs is actually *differentiation over a special subspace*. Now, the selection of this subspace is not straight forward and the operator tries to capture the selection of this subspace. In doing so the operator mimics the generation of the Self-Symmetric input states used in Section 3. We first give a generic definition of the operator referred to as $m-$fold vectorial derivatives. Later we show that the way variables are manipulated in the experiment before is a special case of computing these derivatives. The basic idea behind evolves from the notion of *repeated vectorial derivatives* with respect to *disjoint* sets of variables and is defined as below.

Definition 5 ($m-$Fold Vectorial Derivative)**.** *Let* $\{\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_m}, \mathbf{x_{m+1}}\}$ *be* $(m+1)$ *partitions of the Boolean variables* $(x_1, x_2, \cdots, x_n)$ *and* $f(\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_m}, \mathbf{x_{m+1}}) = f(x_1, x_2, \cdots, x_n) =$

$f(\mathbf{x})$ a Boolean function of n variables, then

$$\left.\frac{\partial^m f}{\partial \mathbf{x_m} \cdots \partial \mathbf{x_2} \partial \mathbf{x_1}}\right|_{\substack{(\mathbf{x_1},\mathbf{x_2},\cdots,\mathbf{x_m}) \\ =(\mathbf{c_1},\mathbf{c_2},\cdots,\mathbf{c_m})}} = \frac{\partial}{\partial \mathbf{x_m}} \left(\cdots \left(\frac{\partial}{\partial \mathbf{x_2}} \left(\left.\frac{\partial f}{\partial \mathbf{x_1}}\right|_{\mathbf{x_1}=\mathbf{c_1}} \right) \right|_{\mathbf{x_2}=\mathbf{c_2}} \right) \cdots \right) \Bigg|_{\mathbf{x_m}=\mathbf{c_m}}$$

is the $\mathbf{m}-$***fold vectorial derivative*** *of the Boolean function* $f(\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_m}, \mathbf{x_{m+1}})$ *with regards to the* m *partitions* $\{\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_m}\}$.

The partitioning of the variables corresponds to the selection of special subspaces. The corresponding expression for computing such a derivative is given below.

$$\left.\frac{\partial^m f}{\partial \mathbf{x_m} \cdots \partial \mathbf{x_2} \partial \mathbf{x_1}}\right|_{\substack{(\mathbf{x_1},\mathbf{x_2},\cdots,\mathbf{x_m}) \\ =(\mathbf{c_1},\mathbf{c_2},\cdots,\mathbf{c_m})}} = \bigoplus_{\substack{\{\mathbf{x_1},\mathbf{x_2},\cdots,\mathbf{x_m}\}\in\mathbf{C} \\ \mathbf{x_{m+1}}=\mathbf{c_{m+1}}}} f(\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_m}, \mathbf{x_{m+1}}) \tag{1}$$

$$\text{where, } \mathbf{C} = \begin{bmatrix} c_1, & c_2, & \cdots, & c_{m-1}, & c_m \\ c_1, & c_2, & \cdots, & c_{m-1}, & \overline{c_m} \\ c_1, & c_2, & \cdots, & \overline{c_{m-1}}, & c_m \\ c_1, & c_2, & \cdots, & \overline{c_{m-1}}, & \overline{c_m} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \overline{c_1}, & \overline{c_2}, & \cdots, & \overline{c_{m-1}}, & \overline{c_m} \end{bmatrix}_{2^m \times m} \quad \mathbf{c_i} \in \mathbb{F}_2^{|\mathbf{x_i}|}$$

Relation with previous Experiment: Now let us look back at the experiment we did earlier and map it to the above definition. It can be noticed it just describes a special partitioning of the variables. Every partition contains two variables that hold symmetric position in the input message after suffixing and padding. This leads to the following expression which is a special case of Equation 1.

$$\left.\frac{\partial^m \texttt{SHA3-512}}{\partial \mathbf{x_m} \cdots \partial \mathbf{x_2} \partial \mathbf{x_1}}\right|_{\substack{(\mathbf{x_1},\mathbf{x_2},\cdots,\mathbf{x_m}) \\ =(\mathbf{c_1},\mathbf{c_2},\cdots,\mathbf{c_m})}}$$

$$= \bigoplus_{\substack{\{\mathbf{x_1},\mathbf{x_2},\cdots,\mathbf{x_m}\}\in\mathbf{C} \\ \forall i>m \;\; \mathbf{x_i}=\mathbf{const}\in\{\mathbf{00},\mathbf{11}\}}} \texttt{SHA3-512}(\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_m}, \mathbf{x_{m+1}}, \mathbf{x_{m+2}}, \cdots, \mathbf{x_{288}})$$

$$\text{where, } \mathbf{C} = \begin{bmatrix} c_1, & c_2, & \cdots, & c_{m-1}, & c_m \\ c_1, & c_2, & \cdots, & c_{m-1}, & \overline{c_m} \\ c_1, & c_2, & \cdots, & \overline{c_{m-1}}, & c_m \\ c_1, & c_2, & \cdots, & \overline{c_{m-1}}, & \overline{c_m} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \overline{c_1}, & \overline{c_2}, & \cdots, & \overline{c_{m-1}}, & \overline{c_m} \end{bmatrix}_{2^m \times m} \quad \mathbf{c_i} \in_R \{00, 11\}$$

There are total of 288 partitions for `SHA3-512`. For the symmetric value constraint all variables must take values from $\{00, 11\}$. The partitions with respect to which the $m-$fold vectorial derivative is computed exhaust the set $\mathbf{C}$. Each of the remaining partitions are fixed to any random value[5] from $\{00, 11\}$.

So the experiment in Section 3 corresponds to computing $17, 16, 15, 14, 13-$fold vectorial derivatives of `SHA3-512` reduced to 4-rounds. This explains the first two rows of Table 2. Since, the degree of 4-rounds of Keccak-p and consequently `SHA3`[6] is ≤ 16, computing the $17-$fold vectorial derivative leads to a ZeroSum. As regards the $16-$fold vectorial

[5] Except for the partitions that need fixed value to handle padding and suffixing while maintaining the self-symmetry. (Refer Appendix B)

[6] Since it is operating on a single message block thereby requiring a single call to Keccak-p.

derivative giving the ZeroSum is justified by the fact that for the current choice of values for constant partitions maximum degree could not be reached. This partially answers our question about the nature of the `Output-Sum`. *However, it still does not account for the Symmetric-Sum property.* For this we showcase the following property which is generic for a class of SPN based iterated round functions.

5 The degree of terms involving round-constants in the ANF of SPN based functions

The following lemma is stepping stone for the next theorem that helps us to establish a relation between the algebraic degree of the monomials in the ANF of specific type of iterated round functions (permutations), that involve round constants and the ones that remain independent.

Lemma 1. *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* $(\mathcal{G})$ *can be expressed as*

$$\mathcal{G} = \mathcal{F} + c \times \mathcal{H}, \textit{where } \begin{cases} d^\circ\mathcal{G} = d^\circ\mathcal{F} \\ d^\circ\mathcal{G} > d^\circ\mathcal{H} \end{cases} \quad \textit{and} \quad \mathcal{G}, \mathcal{F}, \mathcal{H} : \mathbb{F}_2^n \to \mathbb{F}_2^n, \;\; c \to \textit{constant}$$

Proof. Let us consider a round function $\mathcal{G} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $\mathcal{G} = \mathcal{C} \circ \mathcal{N} \circ \mathcal{L}$, where, $\mathcal{C}$ represents round-constant addition, $\mathcal{N}$ is the non-linear component while $\mathcal{L}$ is the linear component. Without loss of generality we can consider the sequence of component operations as $\mathcal{C} \circ \mathcal{N} \circ \mathcal{L}$. The q-round function $\mathcal{G}^q$ can be unrolled as below:

$$\begin{aligned} \mathcal{G}^q &= (\mathcal{C}_q \circ \mathcal{N} \circ \mathcal{L}) \circ (\mathcal{C}_{q-1} \circ \mathcal{N} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_2 \circ \mathcal{N} \circ \mathcal{L}) \circ (\mathcal{C}_1 \circ \mathcal{N} \circ \mathcal{L}) \\ &= \Big[((\mathcal{C}_q \circ \mathcal{N} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_2 \circ \mathcal{N} \circ \mathcal{L})) \circ \mathcal{C}_1\Big] \circ (\mathcal{N} \circ \mathcal{L}) \end{aligned} \tag{2}$$

Equation (2) shows that due to the ordering of the operations, *first round* $\mathcal{N}$ *has no effect on the monomials involving the round constants.* In order to segregate the monomials depending on round constants (`TYPE-II` monomials) from the ones that are independent (`TYPE-I` monomials) we use the following visualization[7] of the ANF of any co-ordinate function $\mathcal{F}^q$ of the vectorial function $\mathcal{G}^q$.

$$\mathcal{F}^q = \mathcal{F}_s^q \oplus \mathcal{F}_{s'}^q \text{ where } \begin{cases} \texttt{TYPE-I} \text{ monomials} \in \mathcal{F}_s^q \\ \texttt{TYPE-II} \text{ monomials} \in \mathcal{F}_{s'}^q \end{cases}$$

We now proceed with the proof by induction:

Base Case: $q = 1$ We have $\mathcal{F} = \mathcal{C}_1 \circ \mathcal{N} \circ \mathcal{L}$. Now using the segregation made above we look at the degree of its `TYPE-I` and `TYPE-II` monomials:

$$\begin{aligned} & d^\circ\mathcal{F}_s = d^\circ(\mathcal{N} \circ \mathcal{L}) = d^\circ\mathcal{N} = \lambda \text{ (Say)} = \text{Highest degree of } \mathcal{F} \\ & d^\circ\mathcal{F}_{s'} = 0 \; [\because \mathcal{N} \text{ has no effect on } \texttt{TYPE-II}] \\ \implies & d^\circ\mathcal{F}_s > d^\circ\mathcal{F}_{s'} \end{aligned}$$

[7] For example, for the ANF of a Boolean function $f = x_1x_2x_3 + c_1c_2x_2x_3 + x_3x_4 + c_2c_3$ where, c_i is a constant, the above visualization is as follows.

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

Thus by definition, for this example `TYPE-I` monomials are $\{x_1x_2x_3,\ x_3x_4\}$ while `TYPE-II` monomials are $\{c_1c_2x_2x_3,\ c_2c_3\}$.

So, the lemma holds for $q = 1$.

Inductive hypothesis: Let us assume that the lemma is true for $q = m$ i.e., all the highest degree monomials of $\mathcal{F}^m$ are of TYPE-I and $d^\circ\mathcal{F}^m_s > d^\circ\mathcal{F}^m_{s'}$. Also assume that $d^\circ\mathcal{F}^m = \lambda^t$ where, $t \leq m$.

Inductive step: Let $q = m + 1$. Now, $\mathcal{F}^{m+1} = \mathcal{C}_{m+1} \circ \mathcal{N} \circ \mathcal{L} \circ \mathcal{F}^m$

$$\begin{aligned} d^\circ\mathcal{F}^{m+1}_{s'} &\leq d^\circ(\mathcal{N} \circ \mathcal{L}) \times d^\circ\mathcal{F}^m_{s'} \\ &< d^\circ(\mathcal{N} \circ \mathcal{L}) \times d^\circ\mathcal{F}^m_s \ [\because d^\circ\mathcal{F}^m_s > d^\circ\mathcal{F}^m_{s'} \text{ by inductive hypothesis}] \\ &\leq d^\circ\mathcal{F}^{m+1}_s \end{aligned}$$

Hence, by the principle of mathematical induction, the lemma holds $\forall q \in \mathbb{N}$. We thus have $d^\circ\mathcal{F}^q = d^\circ\mathcal{F}^q_s$ and $d^\circ\mathcal{F}^q_s > d^\circ\mathcal{F}^q_{s'}$ implying the algebraic degree of $d^\circ\mathcal{F}^q$ is determined by *only* TYPE-I monomials which in turn implies independence from the round constants. □

To properly utilize the result of lemma Lemma 1, it is vital to get a upper bound on the highest degree of the TYPE-II monomials and its relation to the highest degree of TYPE-I monomials i.e., $d^\circ\mathcal{F}^q$. This is established by the following Theorem.

Theorem 1. *The upper-bound on the degree of TYPE-II monomials is given by the following expression:* $d^\circ\mathcal{F}^q_{s'} \leq d^\circ\mathcal{F}^q - d^\circ\mathcal{N}$.

Proof. Let $d^\circ\mathcal{N} = \lambda$. A TYPE-II monomial reaches its highest degree if in the q^{th} round non-linear operation, $(\lambda - 1)$ number of TYPE-I monomials from some $(\lambda - 1)$ coordinate functions of $\mathcal{F}^{q-1}_s$ (Which by lemma Lemma 1 reach the highest degree for $q - 1$ rounds) get mixed with one TYPE-II monomial of $\mathcal{F}^{q-1}_{s'}$. The mix is completely determined by the linear diffusion layer that precedes the non-linear layer.

Let $\left[\mathcal{F}^{q-1}\right]_j$ be the j^{th} coordinate of the vectorial function $\mathcal{G}^{q-1}$. Then $\left[\mathcal{F}^{q-1}_s\right]_j$ has all TYPE-I monomials of $\left[\mathcal{F}^{q-1}\right]_j$ while $\left[\mathcal{F}^{q-1}_{s'}\right]_l$ has all TYPE-II monomials of $\left[\mathcal{F}^{q-1}\right]_l$. Now the statement made above translates into the following where the set of indices S and the index l are determined by the linear diffusion layer.

$$\begin{aligned} d^\circ\mathcal{F}^q_{s'} &= d^\circ\left(\left(\prod_{\substack{j\in S;|S|=\lambda-1 \\ S\subset\{1,2,\cdots,n\}}} \left[\mathcal{F}^{q-1}_s\right]_j\right) \times \left[\mathcal{F}^{q-1}_{s'}\right]_l\right) \\ &\leq \left[(\lambda-1) \times d^\circ\mathcal{F}^{q-1}\right] + k, \qquad \text{where } k = d^\circ\mathcal{F}^{q-1}_{s'} < d^\circ\mathcal{F}^{q-1} \\ &= \lambda \times d^\circ\mathcal{F}^{q-1} - d^\circ\mathcal{F}^{q-1} + k \\ &= d^\circ\mathcal{F}^q - \left[d^\circ\mathcal{F}^{q-1} - k\right] \end{aligned}$$

Now,

$$[d^\circ\mathcal{F}^{q-1} - k] > 0 \quad \left[\because k < d^\circ\mathcal{F}^{q-1}\right]$$

So to maximize $d^\circ\mathcal{F}^q_{s'}$, we need to minimize $[d^\circ\mathcal{F}^{q-1} - k]$. Again, since k and $d^\circ\mathcal{F}^{q-1}$ are both multiples of λ, the least value of $[d^\circ\mathcal{F}^{q-1} - k]$ is λ. Thus we have,

$$\begin{aligned} d^\circ\mathcal{F}^q_{s'} &\leq d^\circ\mathcal{F}^q - \left[d^\circ\mathcal{F}^{q-1} - k\right] \\ &\leq d^\circ\mathcal{F}^q - \lambda \\ &= d^\circ\mathcal{F}^q - d^\circ\mathcal{N} \end{aligned}$$

□

Corollary 1. *For q rounds of the SHA3 permutation* KECCAK*-p, the maximum degree of a monomial involving a round-constant is $d^{\circ}\mathcal{K}^q - 2$*

The above Theorem points us in the direction of getting a round-constant independent function. This is achieved by a lemma which brings into context the $m-$fold vectorial derivatives introduced earlier.

A Round-Constant Independent Function

Lemma 2. *The $(d^{\circ}\mathcal{F} - d^{\circ}\mathcal{N} + 1)-$fold vectorial derivative of $\mathcal{F}^q$, is a function that is independent of round constants.*

This easily follows from Theorem 1 as the $(d^{\circ}F - d^{\circ}\mathcal{N} + 1)-$fold vectorial derivative of $\mathcal{F}^q$ will give a function that has no TYPE-II monomials.

Corollary 2. *For q rounds of* KECCAK*-p the $(d^{\circ}\mathcal{K}^q - 1)-$fold vectorial derivative is a round-constant independent function.*

The corollary brings us one step closer to explaining the results of Table 2. However, it is still not sufficient because the property by itself is generic. We need to connect it with some specific characteristic of KECCAK-p. Herein, comes the *Translation Invariance Property* which is associated with the symmetry of the internal state of KECCAK-p.

Translation Invariance of $\boldsymbol{\theta, \rho, \pi, \chi}$ All step mappings with the exception of ι are translation invariant in the z-axis [BDPA11]. In the context of the current work it implies that

$$\forall \mathcal{S} \in \mathcal{S}^{\#}, \;\; \chi \circ \pi \circ \rho \circ \theta(\mathcal{S}) \in \mathcal{S}^{\#}$$

Thus the Self-Symmetric structure of the internal state is preserved. Since, Corollary 2 establishes the round-constant independence for $(d^{\circ}\mathcal{K}^q - 1)-$fold vectorial derivative of KECCAK-p, this should in turn give a translation invariant function. Consequently, the $(d^{\circ}\mathcal{K}^q - 1)-$fold vectorial derivative will preserve the symmetry of the inputs. The same can be verified using self-symmetric inputs. This conforms to the constraint on the inputs that we impose in our experiment in Section 3. Finally, we are now in a position to make a proposition that completely justifies the findings of Table 2 and consequently forms the basis of a new type of distinguisher for SHA3 referred to as SymSum.

6 SymSum Distinguishers against SHA3

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

The physical significance of this is that the $(d^{\circ}\texttt{SHA3} - 1)-$fold vectorial derivative of SHA3 is independent of the effect of round constant addition which is the only symmetry disturbing operation in KECCAK-p. Thus it preserves the symmetry induced by the input states. In Table 2, we have seen that 16-fold vectorial derivative yields a Zero-Sum signifying that for this case $(d^{\circ}\texttt{SHA3-512} = 15)$. Thus, by virtue of the above proposition, we have $15-$fold and the $14-$fold vectorial derivatives exhibiting a Symmetric-Sum. Moreover, the proposition does not guarantee such a property for $m < (d^{\circ}\texttt{SHA3} - 1)$. Thus we see that the 13 and $12-$fold vectorial derivative are no longer symmetric in nature. This experiment has been repeated with various message sets on all SHA3 variants and the results were found to conform to the theoretical arguments. We now formally define the SymSum distinguisher.

Algorithm 1 `SymSum` $(\mathbb{H}, \mathit{d}, \mathit{mode}, v)$

Require:
- $\mathbb{H} : (\mathbb{F}_2^r)^* \to \mathbb{F}_2^h/\mathbb{F}_2^v$ for $\mathit{mode} =$ `FXD/VAR`, $h \in$ `{224, 256, 384, 512}`
- $\mathit{d} = d^\circ\mathbb{H} \to$ (Estimated) Algebraic degree of $\mathbb{H}$
- $\mathit{mode} \in \{$`FXD`, `VAR`$\}$
- $v \xrightarrow{\texttt{Optional}}$ Hash-length for $\mathit{mode} =$ `VAR`

Ensure:
- `TRUE` if $\mathbb{H} \leftrightarrow$ `SHA3` variant
- `FALSE` Otherwise

1: Choose $\mathcal{M} : \Big[\texttt{Pad(AddSuffix(}\mathcal{M}\texttt{))}\Big|\Big|0^c\Big] \in \mathcal{S}^{\#}$ ▷ Base Message
2: Choose any $\mathit{d}-1$ bits from $\mathcal{M}$ ▷ Except fixed bits (Refer Appendix B)
3: **for** $i \in \{0,1\}^{\mathit{d}-1}$ **do**
4: $\quad m_i = \mathcal{M} \xleftarrow[\texttt{and symmetric conterparts}]{\texttt{Assign } \mathit{d}-1 \texttt{ bits}} i$ ▷ Generating individual messages
5: $\quad h_i \leftarrow \mathbb{H}\Big(m_i\Big)$
6: **end for**
7: `Output-Sum` $\leftarrow \bigoplus_{i=0}^{2^{\mathit{d}-1}} h_i$ ▷ Computing $(\mathit{d}-1)-$fold vectorial derivative
8: **if** $\big(\mathit{mode} =$ `VAR`$\big)\&\big(v > r\big)$ **then**
9: $\quad$ `Output-Sum` $\leftarrow$ Trunc$\Big[$`Output-Sum`$\Big]_r$ ▷ Truncate to r bits for `SHA3 XOFs`
10: **end if**
11: **if** $\Big($`Output-Sum`$_{64} \to$ `Output-Sum`$^{\#}_{64}\Big)$ **then**
12: $\quad$ **return** `TRUE`
13: **else**
14: $\quad$ **return** `FALSE`
15: **end if**

Definition 6 (Symmetric Sum (`SymSum`))**.** *Let us consider the* ***SHA3*** *fixed-length hash functions* ***SHA3-h*** $: (\mathbb{F}_2^r)^* \to \mathbb{F}_2^h$ *or* ***XOFs SHAKE128/256*** $: (\mathbb{F}_2^r)^* \to \mathbb{F}_2^*$*. 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****.*

One can confirm that the padded messages used in the experiment in Section 3 conform to the above definition and hence the message set constitutes a `SymSum` structure. It is also evident that one can construct many such structures. Algorithm 1 elaborates how `SymSum` can be used as a distinguisher for `SHA3`. Here the argument *mode* differentiates between the fixed and variable length variants. v denotes the output hash-length for `XOFs`. For the sake of consistency with Definition 6, we assume that $v \geq 64$ bits. For `XOFs`, if desired hash-length is greater than the rate, then the `Output-Sum` is truncated to r bits. Next, we find the distinguishing advantage of `SymSum` by assessing the probability of the `SymSum` property arising at random.

Advantage of `SymSum` Let us randomly select a string from a space of dimension equal to the hash-length (h). The probability that the 64-prefix of such a string turns out to be symmetric can be computed as below:

- For $x \in_R \{0,1\}^{64}$
$$\Pr\Big[x[0\cdots31] \oplus x[32\cdots63] = 0\Big] = \frac{1}{2^{32}}$$
- For $X \in_R \{0,1\}^h$, number of 64-bit substrings $\in X_{64} = \lfloor \frac{h}{64} \rfloor$

- Probability that 64-prefix of X is symmetric is given by:

$$\Pr\left[X_{64} = X_{64}^{\#}\right] = \prod_{\forall x \in X_{64}} \Pr\left[x[0 \cdots 31] \oplus x[32 \cdots 63] = 0\right] = 2^{-32 \times \lfloor \frac{h}{64} \rfloor}$$

Thus the advantage of SymSum is given by the following expression which shows that the SymSum distinguisher is almost deterministic.

$$\text{Adv}_{\texttt{SymSum}} = 1 - 2^{-32 \times \lfloor \frac{h}{64} \rfloor} \approx 1$$

6.1 Degrees of Freedom

By degrees of freedom we refer to the number of variables against which higher order derivatives can be computed. Generally, while computing $m-$fold *simple* derivative one has maximum degrees of freedom since all variables are *free* in absence of any constraint. However, this is apparent that this degrees of freedom decreases while computing $m-$fold *vectorial* derivatives owing to the partitioning of the variables into disjoint sets.

Since, the classical ZeroSum technique corresponds to finding $m-$fold simple derivatives, it enjoys maximum degrees of freedom. For ZeroSum computation on SHA3 this can be achieved if message-size $= (r-4)$ for fixed-length and $(r-6)$ for XOFs. The reduction of bits is attributed to Suffix = 01 (fixed-length), Suffix = 1111(XOFs) and Padding (Minimum) = 11. As for SymSum, which has been established above as equivalent to computing $m-$fold vectorial derivatives, a loss in the degrees of freedom is incurred due to the Self-Symmetry constraint on inputs in addition to the reduction of bits above. The figures for SymSum stand at $2^{\frac{r-8}{2}}$ for fixed-length and $2^{\frac{r-12}{2}}$ for the variable-length members of SHA3 family. Table 3 presents a comparison after plugging the value of r. In the next subsection we put the ZeroSum and SymSum distinguishers into perspective and compare their impact on SHA3/Keccak.

Table 3: Comparison of degrees of freedom of classical ZeroSum and SymSum for SHA3 fixed-length and XOF members.

	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}			

6.2 Comparison with ZeroSum

The ZeroSum distinguisher has been one of the most extensively studied distinguishers on Keccak-f and is the only distinguisher that is able to penetrate all the 24 rounds of Keccak-f using the *inside-out* strategy. The efficiency of this distinguisher is measured in terms of the size of the ZeroSum structure which is directly determined by the algebraic degree(d) of the underlying function. This directly follows from the fact that ZeroSum property is exhibited by any $m-$fold simple derivative of the function where $m > d$. Thus, the research on the ZeroSum distinguisher has been concentrated on improving the

upper bounds on the degree of KECCAK-f permutation which automatically translated into reduction of the size of ZeroSum structures. This resulted in findings that show that the algebraic degree of KECCAK-f stops growing exponentially after 10 rounds. Due to the prospect of the *inside-out* technique there have been studies on improving the bounds of the inverse of the KECCAK-f round function. However, ZeroSum has this fundamental limitation that its complexity is *lower-bounded* by the (estimated) algebraic degree of KECCAK-f. Moreover, results on KECCAK-f/KECCAK-p do not directly apply on KECCAK/SHA3 since the inside-out analysis is no longer valid on the over-all hash-function. Thus, though ZeroSum for KECCAK-p reaches 24 round, the same for SHA3 manages to go up to 10 rounds (Refer Table 4) before exhausting the degrees of freedom (Refer Table 3). In addition to that it is worth mentioning that since ZeroSum is based on the evolution of the degree of the round-function, it only exploits the properties of χ and/or χ^{-1} among the different component mappings.

Table 4: Complexity of classical ZeroSum Vs SymSum

		Complexity	
#Rounds (n_r)	Bound on d°SHA3	ZeroSum ($2^{d^{\circ}\text{SHA3}+1}$)	SymSum ($2^{d^{\circ}\text{SHA3}-1}$)
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[BCC11]	$\star$	$\star$

† Not applicable for SHA3-512 and SHA3-384 $\star$ Exceeds degrees of freedom (Table 3)

SymSum seems to bring to the table interesting avenues despite being also dependent on the algebraic degree as it also computes higher order derivatives (of a special form). The first significant improvement over ZeroSum is the **4** *times* reduction in the size of a structure exhibiting SymSum property in comparison to ZeroSum since the order of the derivatives can be reduced a couple of times for SymSum. A comparison of the complexities of finding ZeroSum and SymSum structures is furnished in Table 4. It was mentioned in Subsection 6.1 that SymSum, due to the self-symmetry constraint imposed on its input set, looses degrees of freedom. However, this does not have a significant impact since it is evident from Table 4 that with regards to SymSum, ZeroSum is able to reach one extra round (Round 10) for four variants. For all the other rounds SymSum performs better that ZeroSum. It can be noted that the SymSum property holds only if the degree grows maximally. Hence it cannot be exploited for higher rounds as it has been shown in literature that the algebraic degree of KECCAK stops growing maximally after 10 rounds. However, in the current work our motivation was to present a property that holds despite the degree growing maximally. This is confirmed by the fact that the degrees of freedom of SymSum allows it be used up to a maximum of 9 rounds. Interestingly, as shown in Table 4 ZeroSum too exhausts its degrees of freedom after 10 rounds.

Most of the results on ZeroSum work on the KECCAK permutation but cannot be adapted to the hash function as we pointed out in the introduction of the work. The only result that works is a better bound on the degree. This only happens when the degree stops growing maximally. So, ZeroSum gives a generic result for maximal growth of degree for the hash function. On the other hand, SymSum gives a fourfold better result *despite* the maximal growth. Finally, SymSum not only exploits the properties of KECCAK-p round function pertaining to its algebraic degree but it also capitalizes on the translation invariance property (due to θ, ρ, π, χ) and round-constant independence (due to position of ι in the round function). Additionally, SymSum constitutes the first property of KECCAK/SHA3 that relies on the round-constants but is *independent* of their hamming-weights.

7 Conclusion

In this work we investigated an interesting symmetric property exhibited by the sum of SHA3 message digests computed over a specially constructed set of inputs. We put forward a mathematical framework to explain the results which consists of a operator that tries to select a specific subspace over which it computes higher order derivatives and a relation that estimates the degree of round-constant dependent terms in ANF for SPN based functions. Putting all results together a new distinguisher SymSum is proposed for the SHA3 family which corresponds to computing $m-$fold vectorial derivative of the hash-function using self-symmetric input states. SymSum which has a high distinguishing advantage is shown to penetrate up to 9 rounds of SHA3 and outperform classical ZeroSum distinguisher by a factor of four.

Acknowledgement

We would like to thank the anonymous reviewers for their valuable comments.

References

[AK09] Jean-Philippe Aumasson and Dmitry Khovratovich. First Analysis of Keccak. Available online, 2009. http://131002.net/data/papers/AK09.pdf.

[AM09] Jean-Philippe Aumasson and Willi Meier. Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. NIST mailing list, 2009. http://www.131002.net/data/papers/AM09.pdf.

[Bau] Renaud Bauvin. Keccak implementation in Python, supporting the FIPS 202 standard instances. Available online. http://keccak.noekeon.org/KeccakInPython-3.2.zip.

[BC10a] Christina Boura and Anne Canteaut. A zero-sum property for the Keccak-f permutation with 18 rounds. In *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*, pages 2488–2492, 2010.

[BC10b] Christina Boura and Anne Canteaut. Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak-f and Hamsi-256. In *Selected Areas in Cryptography - 17th International Workshop, SAC 2010, Waterloo, Ontario, Canada, August 12-13, 2010, Revised Selected Papers*, pages 1–17, 2010.

[BCC10] Christina Boura, Anne Canteaut, and Christophe De Cannière. Higher-order differential properties of Keccak and Luffa. *IACR Cryptology ePrint Archive*, 2010:589, 2010.

[BCC11] Christina Boura, Anne Canteaut, and Christophe De Cannière. Higher-Order Differential Properties of Keccak and *Luffa*. In *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, pages 252–269, 2011.

[BDPA07] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. Sponge functions. Ecrypt Hash Workshop 2007, May 2007.

[BDPA11] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. The Keccak reference. Submission to NIST (Round 3), 2011. http://keccak.noekeon.org/Keccak-reference-3.0.pdf.

[DGPW12] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei. Unaligned Rebound Attack: Application to Keccak. In *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, pages 402–421, 2012.

[DL11] Ming Duan and Xuajia Lai. Improved zero-sum distinguisher for full round Keccak-f permutation. Cryptology ePrint Archive, Report 2011/023, 2011.

[DL12] Ming Duan and XueJia Lai. Improved zero-sum distinguisher for full round keccak-f permutation. *Chinese Science Bulletin*, 57(6):694–697, 2012.

[DM14] Sourav Das and Willi Meier. Differential Biases in Reduced-Round Keccak. In David Pointcheval and Damien Vergnaud, editors, *AFRICACRYPT 2014*, volume 8469 of *Lecture Notes in Computer Science*, pages 69–87. Springer, 2014.

[DMP+14] Itai Dinur, Pawel Morawiecki, Josef Pieprzyk, Marian Srebrny, and Michal Straus. Practical Complexity Cube Attacks on Round-Reduced Keccak Sponge Function. Cryptology ePrint Archive, Report 2014/259, 2014. http://eprint.iacr.org/.

[DMP+15] Itai Dinur, Pawel Morawiecki, Josef Pieprzyk, Marian Srebrny, and Michal Straus. Cube Attacks and Cube-Attack-Like Cryptanalysis on the Round-Reduced Keccak Sponge Function. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 733–761, 2015.

[HWX+16] Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao. Conditional Cube Attack on Reduced-Round Keccak Sponge Function. Cryptology ePrint Archive, Report 2016/790, 2016. http://eprint.iacr.org/2016/790.

[JN15] Jeremy Jean and Ivica Nikolic. Internal Differential Boomerangs: Practical Analysis of the Round-Reduced Keccak-f Permutation. Cryptology ePrint Archive, Report 2015/244, 2015. http://eprint.iacr.org/.

[KSPC14] Sukhendu Kuila, Dhiman Saha, Madhumangal Pal, and Dipanwita Roy Chowdhury. Practical Distinguishers against 6-Round Keccak-f Exploiting Self-Symmetry. In *Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings*, pages 88–108, 2014.

[Lat09] J Lathrop. Cube attacks on cryptographic hash functions. Master's thesis, 2009. Available at http://www.cs.rit.edu/ jal6806/thesis/.

[MPS12] Pawel Morawiecki, Josef Pieprzyk, and Marian Srebrny. Rotational cryptanalysis of round-reduced Keccak. Cryptology ePrint Archive, Report 2012/546, 2012. http://eprint.iacr.org/.

[Nat15] National Institute of Standards and Technology. SHA-3 Standard: Permutation-based hash and extendable-output functions, August 2015. http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf.

[NRM11] María Naya-Plasencia, Andrea Röck, and Willi Meier. Practical Analysis of Reduced-Round Keccak. In *Progress in Cryptology - INDOCRYPT 2011 - 12th International Conference on Cryptology in India, Chennai, India, December 11-14, 2011. Proceedings*, pages 236–254, 2011.

[oST] National Institute of Standards and Technology. SHA-3 : Cryptographic hash algorithm competition. http://csrc.nist.gov/groups/ST/hash/sha-3/index.html.

A Description of SHA3

After the completion of the standardization process of KECCAK, NIST published the new standard as FIPS PUB 202 [Nat15]: *'SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions'*. The SHA3 standard defines four fixed length hash-function along with two variable-length variants dubbed as XOFs or extendable-output functions.

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

The main difference between KECCAK family and the standard functions reported in FIPS 202 lies in the introduction of the domain separation bits which are appended to the message before the standard 10^*1 padding. More specifically,

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

The main motivation on domain separation is that the outputs of SHA3-h hash functions and SHAKE functions be unrelated given the same input to the functions. The SHA3 standard preserves all other properties pertaining to the internal permutation KECCAK-p with some minor notational changes. For instance, in FIPS 202, the internal permutation KECCAK-f $[b]$ of KECCAK is denoted as:

$$\text{KECCAK-}p[b, n_r]$$

where b denoted the width of the permutation while n_r is the number of rounds. For the six SHA3 hash functions, the underlying permutation is KECCAK-p $[1600, 24]$ which is equivalent to KECCAK-f $[1600]$. The SPN based round function of KECCAK-p is a composition of five step mappings operating on a state that is expressed as a three-dimensional array of bits. A particular bit of a state a is indexed using $a[x][y][z]$ where, $x, y \in \mathbb{Z}_5$ while $z \in \mathbb{Z}_{64}$. $a[*][*][z]$ is referred to as the z^{th} slice while $a[x][y][*]$ denotes the (x, y)-lane. Operations involving x, y are performed modulo 5 while those related to z are done modulo 64. The operations are briefed below:

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

$\theta \rightarrow$ The operation θ is linear and it produces diffusion. The parity of two neighboring columns of each bit are added with it. Additions are performed over $GF(2)$.

$$\theta : a[x][y][z] \leftarrow a[x][y][z] + \sum_{y'=0}^{4} a[x-1][y'][z] + \sum_{y'=0}^{4} a[x+1][y'][z-1]$$

$\rho \rightarrow$ Another linear operation ρ translates each lane by some predetermined values $T(x, y)$ called ρ offsets (Refer Page 13, FIPS 202).

$$\rho : a[x][y][z] \leftarrow a[x][y][z + T(x, y)]$$

$\pi \rightarrow$ The operation π is a permutation on a slice and it is defined as

$$\pi : a[x][y][z] \leftarrow a[x'][y'][z]; \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$\chi \rightarrow$ The only nonlinear function χ operates on every row independently.

$$\chi : a[x][y][z] \leftarrow a[x][y][z] + ((\neg a[x+1][y][z]) \wedge a[x+2][y][z])$$

$\iota \rightarrow$ At the end of every round a unique 64-bit round constant is added with the lane $a[0][0][*]$.

B Message-Set Construction for SymSum

The messages used in the SymSum experiment have the following properties:

1. *Single Block*: Every message is of a single block i.e., the size of the message block after adding suffix (as per FIPS 202 specification) and padding is exactly equal to the rate. This implies that there is exactly one call to the KECCAK-p permutation in the absorbing phase while processing a single message.

$$|\texttt{Pad(AddSuffix(Msg))}| = r = 576(\texttt{SHA3-512})$$

2. *Self-Symmetric*: The messages are chosen in such a fashion that they lead to a Self-Symmetric input state for KECCAK-p.

$$\forall \texttt{Msg} \in \texttt{MsgSet}, \left[\texttt{Pad(AddSuffix(Msg))}||0^c\right] \in \mathcal{S}^{\#}$$

3. *Base Message*: For a particular instance of the experiment every message is derived from a base message. The base message conforms to the above mentioned properties. This message has two parts: *symmetric* and *fixed*. The fixed part corresponds to symmetric positions of the bits that would be fixed after appending suffix and padding. The remaining part of the message is variable with symmetric bits having equal values. This will be clear from the following example of a typical base message for SHA3-512:

   58ea364a58ea364ad212d88cd212d88cc71fe688c71fe688af2e37f3af2e37f3
   510b3fea510b3fea028c16ce028c16ce9b2806999b2806999fcb34b99fcb34b9
   62c26a8662c26a

   One can easily identify the symmetric 64-prefix of the message. In this example, 86 represents the fixed part while the other bytes are symmetric. After adding the suffix and padding bits, the last part becomes 62c26a8662c26a86 thereby ensuring the second property above.

To generate 2^d messages from the base message one can choose any[8] set of d bits and vary them *assigning all values from* $\{0,1\}^d$ preserving the symmetric property. All generated messages constitute the Message Set: MsgSet. By construction the *symmetric-difference* over the MsgSet is $\mathbf{0}$.

$$\bigoplus_{\texttt{Msg}\in\texttt{MsgSet}} \texttt{Msg} = \mathbf{0}$$

C Message Sets used for SymSum experiment on SHA3-512

The message set used in the Experiment on SHA3-512 are given below. Here the ***** represent the nibbles from which bits are chosen to be varied to generate individual messages. Recall, to preserve symmetry ***** and ***** must have the same value for a particular message.

MsgSet ={58ea364a58ea364ad212d88cd212d88cc71fe688c71fe688af2e37f3af2e37f3
510b3fea510b3fea028c16ce028c16ce9b0*****9b0*****9fcb34b99fcb34b9
62c26a8662c26a}

As stated in Appendix B these messages after suffixing and padding lead to inputs to the internal permutation that are self-symmetric. Table 5 shows the representative state that generates the set of inputs of KECCAK-p with respect to the above message set.

[8] Except the fixed part.

Table 5: The state representing the self-symmetric input states of KECCAK-p

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

By increasing the number (d) of bits to be varied, `MsgSet` of various sizes (2^d) were generated for the `SymSum` experiment in Section 3 where the `Output-Sum` was computed after running `SHA3-512` on every `Msg` of each such `MsgSet`.