New Results on the SymSum Distinguisher on Round-Reduced SHA3

Sahiba Suryawanshi, Dhiman Saha, Satyam Sachan

de.ci.phe.red Lab
Department of Electical Engineering and Computer Science
Indian Institute of Technology Bhilai, India
(sahibas,dhiman,satyams)@iitbhilai.ac.in

Abstract. In ToSC 2017 Saha *et al.* demonstrated an interesting property of SHA3 based on higher-order vectorial derivatives which led to self-symmetry based distinguishers referred to as SymSum and bettered the complexity w.r.t the well-studied ZeroSum distinguisher by a factor of 4. This work attempts to take a fresh look at this distinguisher in the light of the linearization technique developed by Guo *et al.* in Asiacrypt 2016. It is observed that the efficiency of SymSum against ZeroSum drops from 4 to 2 for any number of rounds linearized. This is supported by theoretical proofs. SymSum augmented with linearization can penetrate up to two more rounds as against the classical version. In addition to that, one more round is extended by inversion technique on the final hash values. The combined approach leads to distinguishers up to 9 rounds of SHA3 variants with a complexity of only 2^{64} which is better than the equivalent ZeroSum distinguisher by the factor of 2. To the best of our knowledge this is the best distinguisher available on this many rounds of SHA3.

Keywords: SHA3· KECCAK· Distinguisher · SymSum· ZeroSum· Higher-order Derivatives

1 Introduction

The hash function KECCAK [3] which went on to be adopted as the SHA3 [18] standard is one of the most extensively studied hash algorithms. While finding pre-images and collisions constitute the primary analysis strategies of a hash function, the paradigm of devising distinguishers give insight into the non-randomness of the construction. Further, it has been evidenced by numerous results in contemporary literature where distinguishers have been exploited to mount collision and pre-image attacks thereby amplifying their scope and impact. In case of SHA3, one of most investigated distinguisher is the ZeroSum distinguisher which is based on the fundamental result of higher-order derivatives that the $(d+1)^{th}$ derivative of a $d-$degree function leads to a zero function. This translates to obtaining a zero XOR-Sum for 2^{d+1} computations of a vectorial function. The main research is in the direction of tight-bounding the value of d which automatically leads to reduction in complexity of computing the ZeroSum. Most of the results have been reported on the internal permutation `Keccak`-f and/or `Keccak`-p. In 2009, Aumasson and Meier [1] introduced `ZeroSum` distinguisher on `Keccak`-f which penetrated up to 16 rounds by leveraging on the *inside-out* strategy. In 2011, Plasencia *et al.* [15] introduce 4 round distinguisher for Hash function rather than internal permutation function, and also give a 2 round pre-image attack and 3 round near-collision attack on `SHA3-224` and `SHA3-256` variants. The same year, Boura *et al.* [4] improvise `ZeroSum` distinguisher. They present `ZeroSum` distinguisher and high order differential derivative for the full `Keccak`-p permutation. In 2012, Duan *et al.* [6] state an advanced `ZeroSum` distinguisher full round `Keccak`-f with 2^{1579} complexity. The same year, Duc *et al.* [7] present the Unaligned Rebound Attack for 8 round distinguisher with lesser complexity. In 2013, Morawiecki *et al.* [14] present rotational cryptanalysis. It allows a preimage attack on 4-round `Keccak` with complexity 2^{506}. It also states distinguisher on 5 rounds `Keccak`-f[1600] permutation with 2^{15} complexity. In 2014 Das *et al.* analyze differential propagation properties of `Keccak` furthermore uses for 6 round Distinguisher with 2^{52} complexity. In 2015, Jean *et al.* [10] produce internal differential boomerang distinguisher. They generate boomerang pairs and analyze the differential property. Their distinguisher depends on round constant. So, according to where permutation starts, their query complexity varies. For `Keccak`-f permutation, when it starts at 0 round, with complexity 2^5, they distinguish up to 6 rounds, and with 2^{13} complexity to 7 rounds. Similarly, when permutation begins with 3rd round with complexity $2^{10.3}$, they distinguish up to 7 rounds, and with $2^{18.3}$ complexity to 8 rounds. Same year, Dinur *et al.*[5] proposed a Cube attack like a cryptanalysis technique that includes algebraic and structural analysis, which contains key recovery and MAC forgery, practical up to 6 rounds and theoretical to 9 rounds of `Keccak`. In 2016, Guo *et al.* [8] introduce the linearization technique called Linear Structure. It permits linearization up to 3 rounds of `Keccak`. It extends the `ZeroSum` distinguisher of `Keccak`-p permutation up to 15 rounds and pre-image attack up to 4 rounds.

It is evidenced from the above discussion that most of the results have been reported on `Keccak`-p that few on the hash function `SHA3`. Moreover, only a few of the distinguishers on `Keccak`-p can be extended on to any `SHA3` variant itself. However, in 2017, Saha *et al.* [17] introduced a new distinguisher called `SymSum` which examines a symmetric property of the output-sum of `SHA3` when evaluated on symmetric inputs. These distinguishers penetrate up to 9 rounds and theoretically achieve a 4-fold improvement over `ZeroSum` in terms of complexity. The prime observation was the position of the nonlinear operation χ in the sequence of sub-operations in the `Keccak`-p round function. Same year, Huang *et al.* [9] improvise a Cube attack named Conditional Cube attack, impose some conditions on specific bits and use Mixed Integer Linear Programming (MILP) to construct conditional cubes with complexity 2^{33}, 7 round cube distinguisher builds on `SHA3-224`. The same year, Qiao *et al.* [16] introduce a pre-image attack up to 5 rounds, by linearize all S-box at first round and form a 3 round differential trail for `SHAKE128` and `SHA3-224`. They put some conditions so that it satisfies for linearization and differential trail. Same year, Li *et al.* [12] proposed a cross-linear structure for a pre-image attack. They constructed a cross-linear structure for `KECCAK` [400] and found a pre-image. The complexity of their attack is 2^{150} for 3 round `SHA3-256`. In 2019, Li *et al.* [13] proposed a pre-image attack referred to as the Allocating Approach on 4 round `SHA3-256`.

In this work, we investigate the `SymSum` property introduced by Saha *et al.* further and try to augment with observations by Guo *et al.* in their work on linear structures. In particular, we achieve a one/two-round advantage by combining `SymSum` with linear structures. However, the structures we use slightly differ from the ones reported in [8] since we do not have any requirement of keeping χ^{-1} to be linear. This is attributed to the fact that we are mounting the attack on the hash-function and hence cannot leverage the inside-out technique. Consequently, we can relax the constraints that were imposed for the same. Further, we show a simple trick to gain one more round by just inverting[1] the last round χ before computing the output-sum. Using all these techniques, we are able to mount `SymSum` distinguishers on up to 9-rounds of `SHA3` variants with a complexity of only 2^{64}. We show that `SymSum` loses its 4-fold advantage over `ZeroSum` when augmented with linear structures and also furnish a proof for the same. The present `SymSum` distinguishers still have a 2-fold advantage making them the best available distinguishers on `SHA3` which are independent of the number (≥ 1) of rounds linearized. We validate most of claims by providing experimental evidence for some of the practically verifiable distinguishers. Our results are summarized in Table 1.

Table 1: Summary of the results reported

`SHA3`-variant	#Rounds	`ZeroSum`	`SymSum`	**Remarks**
`SHA3`-224	8	2^{65}	2^{64}	2R Linear
`SHA3`-256	7	2^{33}	2^{32}	2R Linear
`SHA3`-384	8	2^{33}	2^{32}	2R Linear + χ^{-1}
`SHA3`-512	8	2^{65}	2^{64}	1R Linear + χ^{-1}
`SHAKE128`	9	2^{65}	2^{64}	2R Linear + χ^{-1}
	10	2^{513}	2^{511}	χ^{-1}
`SHAKE256`	8	2^{33}	2^{32}	2R Linear + χ^{-1}
	9	2^{257}	2^{255}	χ^{-1}
	10	2^{513}	2^{511}	χ^{-1}

Organization Rest of the paper is organized as follows. Section 2 gives a brief description of the `SHA3` and `SymSum` distinguisher and linear structures of `KECCAK`-*p*. Section 3 provides proof of how the efficiency of `SymSum` reduces when we apply linearization. The new distinguishers introduced in this work are illustrated in Section 4. The experiments on round-reduced `SHA3` to validate the

[1] This applies to `SHA3` variants where at least one entire plane is available from the hash value

claims are reported in Section 5. A discussion on all the devised distinguishers is furnished in Section 5. Finally, concluding remarks are given in Section 6.

2 Preliminaries

In this section, we give a brief description of the `SymSum` distinguisher and the idea of linear structure in Keccak-p.

2.1 The Keccak Hash Function

The Keccak structure follows Sponge [2] construction that applies fixed-length permutation on variable-length input and maps to variable-length output. It gives $\mathbb{F}_2^n$ length element output from $\mathbb{F}_2^m$ length input element where n and m are of any length. The permutation applied on finite-state $b = r + c$ bits, where r is rate and c is capacity. Here the finite state b of Sponge construction is the width of Keccak-f permutation. The Sponge construction has 2 phases: the absorption and squeezing phases. Firstly the input message M padded according to the padding rule that makes input message after padding $M^{'}$ multiple of r and breaks $M^{'}$ into $m_1, m_2, \ldots m_k$ each of size r. Initially, state b set to all $0's$ which is initialization vector (IV) and input of f is the `XOR`ed value of the first input message block m_1 of size r and r bits of IV then the output of f is `XOR`ed with next input message m_2 and input to f this will happen until all the message blocks get processed this is absorption phase. The required output digest collects on the squeezing phase. Suppose Z is the required digest. If $Z < r$ then, it takes first Z bits of the output of absorbing phase, otherwise, if $Z > r$ then, it needs to input to f and get more bits repeatedly until it gets Z bits output digest. Finally, the output digest Z is the output of the Sponge function.

Keccak-p **Permutation:** There are 7 Keccak-f permutations which are denoted by Keccak-$f[b, n_r]$, here n_r is the number of rounds and b is the width of Keccak-f permutation. n_r depends on b and calculated as $n_r = 12 + 2l$, here $l = \log_2(\frac{b}{25})$ where $b \in \{25, 50, 100, 200, 400, 800, 1600\}$. Keccak-$f$ permutations states can denote as $5 \times 5 \times w$ where $w = \frac{b}{25}$ such that $w \in \{1, 2, 4, 8, 16, 32, 64\}$. Here we use Keccak-$f$ [1600] that require 24 rounds. Each round has 5 mapping $R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$

θ **:** θ mapping is a linear operation that provides diffusion. In the θ mapping $A[x, y, z]$ XORed with parities of neighbouring 2 columns in the following manner:

$$A[x, y, z] = A[x, y, z] \oplus P[(x-1) \bmod 5, *, z] \oplus P[(x+1) \bmod 5, *, (\text{mod} 64)]$$

Here $P[x, *, z]$ is parity of a column that can be calculated as :

$$P[x, *, z] = \bigoplus_{j=0}^{4} A[x, j, z]$$

ρ : ρ mapping is another linear operation that rotates each lane by some predefined values. Here first column and last row represent y axis and x axis values respectively.

$$A[x, y, z] = A[x, y, z_{\lll t}] \ for \ x, y = 0, ...4$$

Here $\lll$ is a bitwise rotation

π: π mapping is another Linear operation which permutes on slices by interchanging lanes as:

$$A[y,(2x+3y)\ mod\ 5,z] = A[x,y,z] \text{ for } x,y=0,...4, z=0,...63$$

χ: χ is the only Non-linear operation that operates on rows independently as:

$$A[x,y,z] = A[x,y,z] \oplus (\sim A[x+1,y,z]) \wedge A[x+2,y,z]$$

ι: A unique RC add to lane $A[0,0]$ depend on round number.

$$A[0,0,*] = A[0,0,*] \oplus RC$$

2.2 `SymSum` Distinguishers on `SHA3`

In 2017, Saha *et al.* introduced an interesting algebraic property related to SPN round functions where the non-linear transformation preceded the round-constant addition. This was used to devise a new class of distinguishers referred to as `SymSum`. The basic result was that the round-constants could not influence the highest degree monomials which determined the upper-bound on the degree of a vectorial function. This helped them devise a round-constant independent function by computing a special type of derivative called the $m-$fold vectorial derivative. They further showed that the order of this derivative can be a factor of 4 less than the `ZeroSum` distinguisher which actually computes the $m-$fold simple derivatives. To verify this property, they used self-symmetric input states as inputs and the hypothesis was that the output sum across all hash values would also preserve the self-symmetry. Self-symmetry of Keccak can be defined as the first 32 slices are identical to the last 32 slices of the Keccak state as shown in

Fig. 1. Here σ_1 and σ_2 are identical.

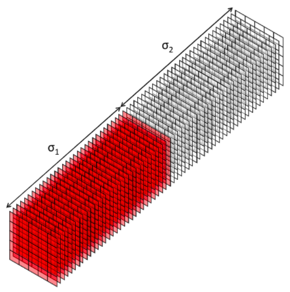


Fig. 1: Self-Symmetric State of Keccak [11]

For brevity, the main results are mentioned below, where `TYPE-II` as defined in [17] are monomials that are dependent on round-constants:

Lemma 1 *[17] For SPN round function $\mathcal{G}$, if the ordering of components is in such a way that the non-linear function precedes from round constant addition* *then $\mathcal{G}$ can express as: $\mathcal{G} = \mathcal{F} + C \times \mathcal{H}$ where $d^\circ\mathcal{G} = d^\circ\mathcal{F}$ and $d^\circ\mathcal{G} > d^\circ\mathcal{H}$ where $\mathcal{G}, \mathcal{F}, \mathcal{H} : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and C is a constant*

Theorem 1. *[17] The upper-bound on the degree of `TYPE-II` monomials is given by the following expression: $d^\circ\mathcal{F}^q_{s'} \leq d^\circ\mathcal{F}^q - d^\circ\mathcal{N}$*

Lemma 2 *[17] 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 constant*

Here $d^\circ\mathcal{F}$, $d^\circ\mathcal{N}$ are the upper bounds on the degrees of function $\mathcal{G}$ and non-linear function $\mathcal{N}$ respectively and $\mathcal{F}^q$ is function after q rounds. Using the above mentioned lemmas the authors furnished a proof that `SymSum` distinguisher is better than `ZeroSum` distinguisher for `SHA3` by a factor of 4.

2.3 Linear Structures

The idea of linearization as introduced by Guo *et al.* is basically a lane-wise restriction on the input space so as to handle the linear θ and non-linear χ operations of the KECCAK-p round function. The authors demonstrate linearization of the KECCAK-p permutation up to 3 rounds: 1 round backward and 2 rounds forward. It extends the `ZeroSum` distinguisher and also leads to new pre-image attacks. To understand the technique, one needs to look at the Boolean expression of the χ function. The primary observation is that if two consecutive variables never come together in a row then, then all output co-ordinate functions of χ become linear. The operation θ which relies on the column parity of spatially adjacent columns can be handled so that it does does not diffuse the state by keeping the column parity constants across calls to KECCAK-p. The idea is captured in Fig. 2. As evident from the figure, to handle the effect of θ on the variables, the following condition is imposed where α is any constant:

$$A[1,0] \oplus A[1,1] \oplus A[1,2] \oplus A[1,3] \oplus A[1,4] = \alpha$$

This can be equivalently written as $A[1,4] = \bigoplus_{j=0}^{3} \mathrm{A[1,\ j]} \oplus \alpha$. This results, in 1 round linearization of KECCAK with degree of input up to 256.

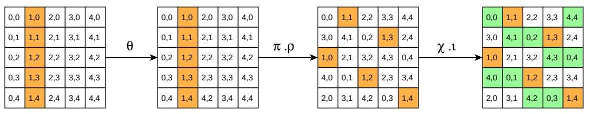


Fig. 2: KECCAK state configuration for 1-round linearization with degrees of freedom 256 [8]. Here white cells are constants, orange cells are variable with degree 1, and green cells have degree at most 1.

To increase the degree of freedom, it is possible to take variables at different columns as shown in Fig. 3 as $A[i,4] = \bigoplus_{j=0}^{3} A[i,j] \oplus\ \alpha_i$ where $i = 0, 2$, $j = 0, 1, 2, 3$.

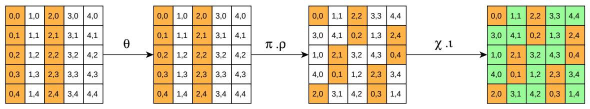


Fig. 3: Keccak state configuration for linearization with degree of freedom 512 [8]

For 2-round linearization, the input state should be taken as shown in the Fig.4, here light gray cells and dark grey has value 0 and 1 respectively. To handle θ at 1 round variables need to satisfy the following condition.

$$A[1,0] \oplus A[1,1] \oplus A[1,2] \oplus A[1,3] = A[1,4] \oplus \texttt{0xf...f}$$

$$A[2,0] \oplus A[2,1] \oplus A[2,2] \oplus A[2,3] = \texttt{0xf...f}$$

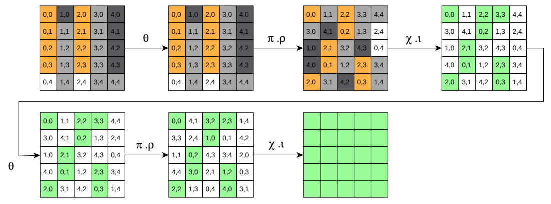


Fig. 4: State configuration to handle χ at 2^{nd} round [8]

At second-round θ, ρ, π permute variables, so to handle 2 round θ variables have to satisfy the following conditions.

$$A[2,0]_{\lll 62} = A[0,0] \oplus A[2,2]_{\lll 43}$$

$$A[2,1]_{\lll 6} = S[0,1]_{\lll 36} \oplus A[2,3]_{\lll 15}$$

$$A[2,2]_{\lll 43} = A[0,2]_{\lll 3}$$

$$A[2,3]_{\lll 15} = A[0,3]_{\lll 41} \oplus A[2,0]_{\lll 62}$$

3 Investigating Effect of Linear Structures on SymSum

Our first study constitutes analyzing the effect of using linear structures in conjunction with the `SymSum` property. We extend the `ZeroSum` distinguisher and `SymSum` distinguisher by applying linearization technique up to 2 rounds. However, we argue that because of linearization, the difference in complexities for obtaining `SymSum` and `ZeroSum` decreases from the factor of 4 to 2. We next try to furnish theoretical arguments to support this claim. Thus, we first need to look at a more general result that compares the behaviour of a SPN round function (as observed in [17]) with and without linearization. For the SPN round function without applying linear structures, the behaviour is described by lemma 1. The following lemma captures the same while incorporating the effect of linearization.

Lemma 3 *For any SPN round function $\mathcal{G}$ iterated for n_r rounds, if $l_r(\leq n_r)$ rounds are linearized, the degrees of the linearized version ($\mathbb{G}$) and unlinearized versions ($\mathbb{G}'$) are related by the degree (λ) of the non-linear component function by the following relation:*

$$d^\circ\mathbb{G} \leq \lambda^{l_r} \times d^\circ\mathbb{G}' \textit{ where } \begin{cases} \mathbb{G} = \mathcal{G}^{n_r} \\ \mathbb{G}' = \mathcal{G}^{n_r - l_r} \circ \mathcal{G}'^{l_r} \\ \mathcal{G}' \leftarrow \textit{Linearized version of } \mathcal{G} \end{cases}$$

Here $d^\circ\mathbb{G}$, $d^\circ\mathbb{G}'$ are the upper bounds on the degrees of $\mathbb{G}, \mathbb{G}'$ respectively.

Proof. Let us write down the degree of the unlinearized version $\mathbb{G}$. Since the degree grows exponentially (before asymptotically converging on the highest possible degree which is determined by the number of independent input variables) in the degree of the non-linear component, we can write the following expression:

$$\begin{aligned} \mathbb{G} &= \mathcal{G} \circ \mathcal{G} \circ \mathcal{G} \circ \cdots n_r \text{ times} \\ \implies d^\circ\mathbb{G} &\leq (d^\circ\mathcal{G})^{n_r} \\ &= \lambda^{n_r} \end{aligned} \tag{1}$$

Now let us write the expression for the linearized version:

$$\begin{aligned} \mathbb{G}' &= \{\mathcal{G} \circ \mathcal{G} \circ \mathcal{G} \circ \cdots (n_r - l_r) \text{ times}\} \circ \{\mathcal{G}' \circ \mathcal{G}' \circ \mathcal{G}' \circ \cdots l_r \text{ times}\} \\ \implies d^\circ\mathbb{G}' &\leq (d^\circ\mathcal{G})^{n_r - l_r} \times (d^\circ\mathcal{G}')^{l_r} \\ &= \lambda^{n_r - l_r} \quad [\because d^\circ\mathcal{G}'^{l_r} = 1] \end{aligned} \tag{2}$$

From Equation 1 and Equation 2 it follows that: $d^\circ\mathbb{G} \leq \lambda^{l_r} \times d^\circ\mathbb{G}'$ □

With the above proof in place, we revisit lemma 1 in the light of linearization. We argue that lemma 1 still holds for the linearized version $\mathbb{G}'$ of $\mathbb{G}$. This implies that the degree of $\mathbb{G}'$ will be determined by monomials which are independent of round constants (`TYPE-I`) from monomials that involve round constants (`TYPE-II`). We use the same terminology as stated in [17] and redo the proof of lemma 1.

Lemma 4 *Lemma 1 holds under linearization.*

Proof. Let us consider the SPN round function ($\mathcal{G}$) as stated above with the restriction that the non-linear operation precedes the round-constant addition (as required by lemma 1). So, let $\mathcal{G} = \mathcal{C} \circ \mathcal{N} \circ \mathcal{L}$ where $\mathcal{C}$ represents the round constant addition, $\mathcal{N}$ is non-linear component, and $\mathcal{L}$ is the linear component. So for n_r rounds, $\mathbb{G}$ can be written as:

$$\begin{aligned} \mathbb{G} &= (\mathcal{C}_{n_r} \circ \mathcal{N} \circ \mathcal{L}) \circ (\mathcal{C}_{n_r - 1} \circ \mathcal{N} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_1 \circ \mathcal{N} \circ \mathcal{L}) \\ &= \Big[(\mathcal{C}_{n_r} \circ \mathcal{N} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_2 \circ \mathcal{N} \circ \mathcal{L}) \circ \mathcal{C}_1\Big] \circ (\mathcal{N} \circ \mathcal{L}) \end{aligned} \tag{3}$$

However, if *linear structures* are applied for l_r rounds then $\mathbb{G}'$ can be expressed as:

$$\begin{aligned}
\mathbb{G}' &= (\mathcal{C}_{n_r} \circ \mathcal{N} \circ \mathcal{L}) \circ (\mathcal{C}_{n_r-1} \circ \mathcal{N} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_{n_r-l_r} \circ \mathcal{N}_{n_r-l_r} \circ \mathcal{L}_{n_r-l_r}) \\
&\quad \circ (\mathcal{C}_{l_r} \circ \mathcal{L}^{'} \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_1 \circ \mathcal{L}^{'} \circ \mathcal{L}) \\
&= \Big[(\mathcal{C}_{n_r} \circ \mathcal{N} \circ \mathcal{L}_q) \circ \cdots \circ (\mathcal{C}_{n_r-l_r} \circ \mathcal{N} \circ \mathcal{L}) \circ (\mathcal{C}_{l_r} \circ \mathcal{L}^{'} \circ \mathcal{L}) \circ \cdots \circ \mathcal{C}_1\Big] \circ (\mathcal{L}^{'} \circ \mathcal{L}) \qquad (4)
\end{aligned}$$

Here $\mathcal{L}^{'}$ is a linearized version of $\mathcal{N}$ thus $d^\circ \mathcal{L}^{'}$ will be reduced by $(\lambda - 1)$. Due to Equation (3) and (4), it can be observed that after 1 round, the round constant $\mathcal{C}_1$ has no effect of $\mathcal{N}$ (or $\mathcal{L}^{'}$ in case of linearization). Now, using the strategy described in [17] to segregate monomials which are independent of round constants (`TYPE-I`) from monomials that involve round constants (`TYPE-II`) we can visualize any co-ordinate function of $\mathbb{G}'$ as $\mathcal{F}^{n_r}$:

$$\mathcal{F}^{n_r} = \mathcal{F}^{n_r}_{c'} \oplus \mathcal{F}^{n_r}_{c} \text{ where } \begin{cases} \texttt{TYPE-I} \in \mathcal{F}^{n_r}_{c'} \\ \texttt{TYPE-II} \in \mathcal{F}^{n_r}_{c} \end{cases}$$

(a) To Prove: $d^\circ \mathcal{F}^m_{c'} > d^\circ \mathcal{F}^m_c$ (Proof by induction)

Base case: Let $n_r = 1$ which implies $\mathbb{G} = \mathcal{G}^1 = \mathcal{C}_1 \circ \mathcal{N} \circ \mathcal{L}$,
However, we have to take into account the linearization. So let $l_r = 1$. Therefore $\mathbb{G}' = \mathcal{G}'^1 = \mathcal{C}_1 \circ \mathcal{L}^{'} \circ \mathcal{L}$. Hence the degree of `TYPE-I` and `TYPE-II` monomials are:

$$\begin{aligned}
d^\circ \mathcal{F}_{c'} &= d^\circ(\mathcal{L}^{'} \circ \mathcal{L}) = \lambda - (\lambda - 1) = 1 \\
d^\circ \mathcal{F}_c &= 0 \big[\because \mathcal{C}_1 \text{is independent of } \mathcal{L}^{'} \big]
\end{aligned}$$

Thus $d^\circ \mathcal{F}_{c'} > d^\circ \mathcal{F}_c$ Hence lemma hold for base condition i.e., at $n_r = 1$

Inductive hypothesis: Let us assume the lemma hold for $n_r = m$ i.e., $d^\circ \mathcal{F}^m_{c'} > d^\circ \mathcal{F}^m_c$

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

$$\begin{aligned}
d^\circ \mathcal{F}^{m+1}_c &\leq d^\circ(\mathcal{N} \circ \mathcal{L}) \times d^\circ \mathcal{F}^m_c \\
&< d^\circ(\mathcal{N} \circ \mathcal{L}) \times d^\circ \mathcal{F}^m_{c'} \big[\because d^\circ \mathcal{F}^m_{c'} > d^\circ \mathcal{F}^m_c \big] \\
&\leq d^\circ \mathcal{F}^{m+1}_{c'}
\end{aligned}$$

Hence, by induction, the lemma holds $\forall n_r \in \mathbb{N}$. □

Our next claim is that the difference in the degrees of `TYPE-I` and `TYPE-II` monomials as stated by Saha *et al.* in [17] no longer holds as we linearize the SPN. For any value of $l_r \geq 1$, the following theorem holds instead. One can note that unlike [17], the following result is independent of the degree of the non-linear component.

Theorem 2. *With at least one round linearized, the upper-bound on the degree of `TYPE-II` monomials in terms of `TYPE-I` monomials is given by:*

$$d^\circ \mathcal{F}_c^{n_r} \leq d^\circ \mathcal{F}_{c'}^{n_r} - 1$$

Proof. We start by segregating the `TYPE-II` monomials further. The new sub-type is referred to as `TYPE-III` and represents a `TYPE-II` monomial which is independent of any variables and constitutes only constants terms as stated below:

$$\prod C_i \text{ where } C_i \text{ is any constant term}$$

Suppose our function is in the linear form up to l_r rounds ($l_r \geq 1$) then using notations used above:

$$\mathcal{G}'^{l_r} = (\mathcal{C}_{l_r} \circ \mathcal{L}' \circ \mathcal{L}) \circ (\mathcal{C}_{l_r-1} \circ \mathcal{L}' \circ \mathcal{L}) \circ \cdots \circ (\mathcal{C}_2 \circ \mathcal{L}' \circ \mathcal{L}) \circ (\mathcal{C}_1 \circ \mathcal{L}' \circ \mathcal{L})$$

Since there is no non-linear function, the degree of `TYPE-I` and `TYPE-II` monomials never change. Also for `TYPE-II` monomials, only `TYPE-III` monomials occur. Thus the degree of `TYPE-I` monomials and `TYPE-II` monomials should 1 and 0 (because of `TYPE-III`), respectively i.e., $d^\circ \mathcal{F}_c^{l_r} = 0$ and $d^\circ \mathcal{F}_{c'}^{l_r} = 1$. Now, we prove by induction.

Base case: Let $n_r = l_r + 1$, i.e. $\mathbb{G}' = \mathcal{G} \circ \mathcal{G}'^{l_r}$ implying a single non-linear function and $d^\circ \mathcal{N} = \lambda$,.

Now, `TYPE-I` monomials will reach the highest degree after the current round when λ `TYPE-I` monomials mix together under $\mathcal{N}$ in the current round. This final degree of `TYPE-I` is expressed as:

$$\begin{aligned} d^\circ \mathcal{F}_{c'}^{l_r+1} &= \sum_{i=1}^{\lambda} \left[d^\circ \mathcal{F}_{c'}^{l_r} \right]_i \\ &= 1 + 1 \cdots \lambda \text{ times} \quad [\because d^\circ \mathcal{F}_{c'}^{l_r} = 1] \\ &= \lambda \end{aligned} \tag{5}$$

Next, `TYPE-II` monomials reach the highest degree when $(\lambda - 1)$ `TYPE-I` monomials from $(\lambda - 1)$ co-ordinate functions mix with one `TYPE-II` monomial. Thus for `TYPE-II` monomials we have

$$\begin{aligned} d^\circ \mathcal{F}_c^{l_r+1} &= \sum_{i=1}^{\lambda-1} \left[d^\circ \mathcal{F}_{c'}^{l_r} \right]_i + d^\circ \mathcal{F}_c^{l_r} \\ &= \sum_{i=1}^{\lambda-1} 1 + 0 \;\; [\because d^\circ \mathcal{F}_c^{l_r} = 0 \; (\texttt{TYPE-III}) \quad d^\circ \mathcal{F}_{c'}^{l_r} = 1] \\ &= \lambda - 1 \end{aligned} \tag{6}$$

Hence, by Equation (5) and (6) theorem holds for base case.

Inductive hypothesis : Let us assume the theorem holds for $n_r = m$ rounds i.e.,

$$d^\circ \mathcal{F}_c^m \leq d^\circ \mathcal{F}_{c'}^m - 1$$

Inductive step: Let $n_r = m + 1$ then by lemma 3 we have $d^\circ \mathcal{F}_{c'}^m \leq \lambda^{m-l_r}$ and $d^\circ \mathcal{F}_c^m \leq \lambda^{m-l_r} - 1$. Then by arguments similar to the base-case, we have degree of `TYPE-I` monomials as:

$$\begin{aligned} d^\circ \mathcal{F}_{c'}^{m+1} &= \sum_{i=1}^{\lambda} \left[d^\circ \mathcal{F}_{c'}^m\right]_i \\ &\leq \sum_{i=1}^{\lambda} \lambda^{m-l_r} = \lambda^{m-l_r+1} \end{aligned} \tag{7}$$

Similarly for `TYPE-II` monomials

$$\begin{aligned} d^\circ \mathcal{F}_c^{m+1} &= \sum_{i=1}^{\lambda-1} \left[d^\circ \mathcal{F}_{c'}^m\right]_i + d^\circ \mathcal{F}_c^m \\ &\leq \sum_{i=1}^{\lambda-1} \lambda^{m-l_r} + \lambda^{m-l_r} - 1 \\ &= (\lambda - 1)\lambda^{m-l_r} + \lambda^{m-l_r} - 1 = \lambda^{m-l_r+1} - 1 \\ &\leq d^\circ \mathcal{F}_{c'}^{m+1} - 1 \ \text{[By Equation (7)]} \end{aligned} \tag{8}$$

Thus by principle of induction Theorem 2 holds $\forall n_r \in \mathbb{N}$. □

We now have the following corollary which forms the base of all distinguishers reported in this work. As one might realize this constitutes a deviation from the result reported in [17] as stated in lemma 2.

Corollary 1. *With l_r linearized rounds $\left(\frac{d^\circ \mathbb{G}}{\lambda^{l_r}}\right)$−fold vectorial derivative of $\mathbb{G}$ is a function which is independent of round constants.*

The corollary easily follows from lemma 3 and Theorem 2. Since linearized version $\mathcal{G}'$ of $\mathcal{G}$ has degree $\left(\frac{d^\circ \mathbb{G}}{\lambda^{l_r}}\right)$ and the maximum degree of `TYPE-II` monomials in $\mathcal{G}'$ is $\left(\frac{d^\circ \mathbb{G}}{\lambda^{l_r}} - 1\right)$, so the $\left(\frac{d^\circ \mathbb{G}}{\lambda^{l_r}}\right)$−fold vectorial derivative of $\mathcal{G}$ will result in a round-constant independent function. Consequently, such a function would preserve the `SymSum` property as introduced in [17]. In the next section, we show how the above results are used to mount highly efficient and practical `SymSum` distinguishers on `SHA3` variants.

4 Augmenting the `SymSum` Distinguisher

The `SymSum` property can be extended at varied number of rounds based on the augmentation strategies like prepending linear structures and appending the hash-inversion trick wherever applicable. This is captured by Fig. 5. In the subsequent sub-sections we explore these strategies that help us to reach highest number of rounds for some `SHA3` variants.

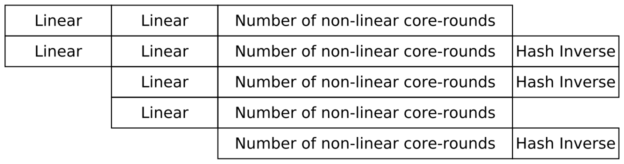

Linear | Linear | Number of non-linear core-rounds
Linear | Linear | Number of non-linear core-rounds | Hash Inverse
Linear | Number of non-linear core-rounds | Hash Inverse
Linear | Number of non-linear core-rounds
Number of non-linear core-rounds | Hash Inverse

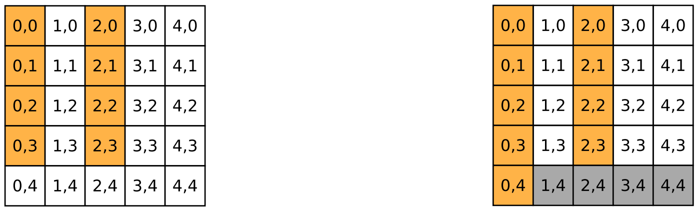
Fig. 5: Various extension strategies to verify the `SymSum` property by augmenting 1-round, 2-round linear structures and the hash-inversion trick

4.1 Extending `SymSum` using 1-round Linearization and χ^{-1} trick

To gain an advantage of 2 rounds for the `SymSum` distinguisher, we linearize the first round and perform $\chi^{-1} \circ \iota^{-1}$ on the output digest when applicable. The input set should satisfy the following conditions so that it linearizes 1 round and also satisfies the condition for `SymSum` distinguisher which constitutes giving self-symmetric inputs:

1. The input set is a set of inputs such that the first 32 slices of the state are the same as the last 32 slices
2. For linearization, input state has the restriction that $\forall A[i,j]$ where $i = 0, 2,\ j = 0, 1, 2, 3,$

$$A[i,3] = \bigoplus_{j=0}^{2} A[i,j] \oplus \alpha_i \text{ for any constant } \alpha$$



(a) Keccak state for 1-round linearization of `SHAKE128` and `SHA3-224`

(b) Input state for 1-round linearization of `SHAKE128`

Fig. 6: Different slice configurations for `SHA3`

The χ^{-1} trick applies only to those variants of `SHA3` which give at least one plane ofKeccak state in the output hash value. Therefore, it is not applicable to `SHA3-224` and `SHA3-256` because they give 224 and 256 bits of hash value respectively which is less that 320 bits required for a full plane. The degree of freedom of this state will be 192 if we take the input state equivalent to the state shown in Fig. 6a. Therefore, after 1-round linearization and applying χ^{-1} strategy, SymSum on SHAKE128 can distinguish up to 9 rounds. For the other variants of SHA3, the input state is different because of the difference in size of capacity part. For instance, after computing the output sum for 4 rounds on SHA3, we get SymSum for 2^4 invocations. For the classical SymSum distinguisher, it is obtained at 2^{15} and 2^{14}. Therefore, the extended SymSum distinguisher has an advantage of 2 rounds, although the effectiveness reduces by the factor of 2.

4.2 Extension of SymSum distinguisher up to 3 rounds:

We now show the use of 2-round linear structures in conjunction with inverting the hash for the last round. For 2 round linearization we use the linear structure, for which we need to handle the θ, ρ, π, χ mappings of KECCAK. To handle the first round χ we take variables in 2 alternative columns so that no two variables come adjacent in χ operation, thus maintaining the linearity after the first round. We restrict other columns to 0 and/or 1, as shown in the Fig. 7 so that before χ in the second round no two adjacent lanes become variable. Additionally, because of the columns that have variables, constant values may change because of θ. To handle θ the following conditions need to be imposed:

$$A[0,0] \oplus A[0,1] \oplus A[0,2] \oplus A[0,3] = A[0,4] \oplus 0xff\ldots f$$
$$A[2,0] \oplus A[2,1] \oplus A[2,2] \oplus A[2,3] = 0xff\ldots f$$

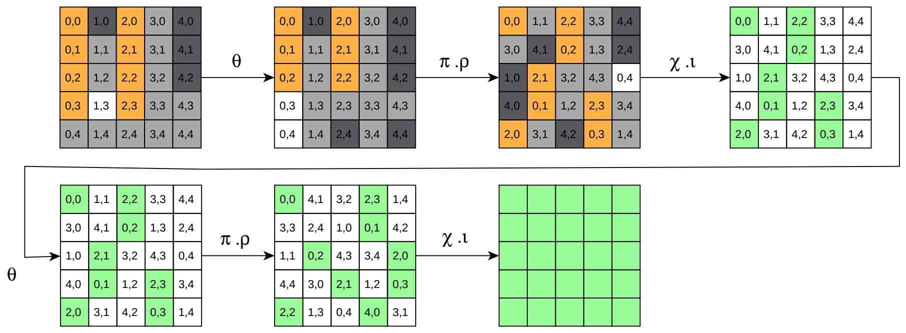


Fig. 7: KECCAK state for 2-round linearization with degree of freedom 64 [8]

Now, to linearize the second round we need to handle θ of second round. But for this the positions of the variables after first round χ need to be closely handles as would change due to ρ and π in the first round. Therefore to make two rounds linear the variables should satisfy the conditions as below:

$$A[2,0]_{\lll 62} = A[0,0] \oplus A[2,2]_{\lll 43}$$

$$A[2,1]_{\lll 6} = S[0,1]_{\lll 36} \oplus A[2,3]_{\lll 15}$$
$$A[2,2]_{\lll 43} = A[0,2]_{\lll 3}$$
$$A[2,3]_{\lll 15} = A[0,3]_{\lll 41} \oplus A[2,0]_{\lll 62}$$

It has been shown in [8] that the above system of equations has 128 degrees of freedom. However, for the SymSum property, we have the additional restriction of self-symmetric inputs which lead to revisiting the system of equations as below. The main idea is to rewrite the equations considering half of the state and then extend the solutions to the other half thereby always keeping the over-all solution self-symmetric.

$$\begin{aligned} A[0,0,k] \oplus A[0,1,k] \oplus A[0,2,k] \oplus A[0,3,k] &= A[0,4,k] \oplus 0xff\dots f \\ A[2,0,k] \oplus A[2,1,k] \oplus A[2,2,k] \oplus A[2,3,k] &= 0xff\dots f \end{aligned} \tag{9}$$

Here $k \in \{0,1,\dots,31\}$. Similarly, to make the second round linear the relations are rephrased w.r.t a 32-lane state as follows:

$$\begin{aligned} A[2,0,k]_{\lll 30} &= A[0,0,k] \oplus A[2,2,k]_{\lll 11} \\ A[2,1,k]_{\lll 6} &= S[0,1,k]_{\lll 4} \oplus A[2,3,k]_{\lll 15} \\ A[2,2,k]_{\lll 11} &= A[0,2,k]_{\lll 3} \\ A[2,3,k]_{\lll 15} &= A[0,3,k]_{\lll 9} \oplus A[2,0,k]_{\lll 30} \end{aligned} \tag{10}$$

From the above equations, we get the first 32 slices that are as per our requirement, therefore, we take a copy of this state and make them the last 32 slices. By doing this the degree of freedom will be 64 because we have 8×32 variables and 6×32 equations. Accordingly, the degree of freedom is $8 \times 32 - 6 \times 32$. Hence for SHAKE128, we get the SymSum for 9 rounds with complexity 2^{64}.

5 Experimental Validation

In this section, we present experimental validation of some of the claims furnished above. In particular, we choose SHAKE128 as it has the smallest capacity part allowing for more control over the input. However, the attacks can easily be extended onto other SHA3 variants with proper adjustments. In the following we demonstrate an attack on 6-rounds of SHAKE128 using the 1-round linearization and hash-inverse strategy. Due to 2-round extension, the degree of 6-rounds reduces to $2^{6-2} = 16$ and by Corollary 1, the 16^{th} order vectorial derivative will exhibit SymSum property.

Fig. 6b shows input state for SHAKE128. Here orange, white, light gray lanes are variable, constant and 0's (that is also capacity part of SHAKE128) respectively. Here, we have taken first and third column as variables which satisfies the conditions as per Equation (9). The input base message for our experiment is shown below:

Table 2 shows the full Keccak State, where **** is the variable nibble that generates individual messages by altering their values. To maintain the Self-Symmetry **** and **** should be the equivalent. To make one round linear

8bd9162e	8bd9162e	1245c1c7	1245c1c7	0a3f3940	0a3f3940	eb6e955a	eb6e955a	61d62226	61d62226
64cf1036	64cf1036	36da615c	36da615c	3d3b488a	3d3b488a	e86d0018	e86d0018	1b16874d	1b16874d
64cf1036	64cf1036	44bbe571	44bbe571	0d0b9c27	0d0b9c27	72f3c98c	72f3c98c	53598e96	53598e96
ebc29253	ebc29253	75f22314	75f22314	92d8c5f9	92d8c5f9	372772f3	372772f3	3839af6d	3839af6d
b185e09f	b185e0								

each message generated by changing **** should satisfy the condition described above thus the value of † † †† and † † †† will modify accordingly.

Table 2: Representing Keccak State

****9162e	****9162e	1245c1c7	1245c1c7	0a3f3940	0a3f3940	eb6e955a	eb6e955a	61d62226	61d62226
64cf1036	64cf1036	36da615c	36da615c	3d3b488a	3d3b488a	e86d0018	e86d0018	1b16874d	1b16874d
64cf1036	64cf1036	44bbe571	44bbe571	0d0b9c27	0d0b9c27	72f3c98c	72f3c98c	53598e96	53598e96
† † ††9253	† † ††9253	75f22314	75f22314	92d8c5f9	92d8c5f9	372772f3	372772f3	3839af6d	3839af6d
b185e09f	b185e09f	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000

By changing **** values of the input base message, 2^{16} individual messages were produced and inputted to 6-round SHAKE128 with output hash-size of 320. For each of the hash-values, apply $\chi^{-1} \circ \iota^{-1}$ and compute the output-sum. We witnessed ZeroSum with complexity 2^{17}. and SymSum with 2^{16} that confirms the expected outcome predicted by theoretical arguments.

6 Discussion

In this work, we have extended the classical SymSum distinguisher up to 3 rounds by applying linear structures and the χ^{-1} trick. Together, we have an advantage of 3 rounds on almost all previously reported derivative based distinguishers.

One of the most important observations was the shift in the highest degree reachable by TYPE-II monomials which are fundamental to achieving a round-constant independent function thereby being the basis of the SymSum distinguisher. As dictated by Theorem 2, irrespective of the number (≥ 1) of rounds linearized SymSum loses its 4 factor advantage over ZeroSum. However, that is a little price to pay against the increase in the number of rounds penetrated. A comparison among the various approaches that extend the SymSum distinguisher is furnished in Fig. 8. The comparisons are provided for 7, 8, 9 and 10 rounds for each variant of SHA3. As one can observe for SHA3-224 and SHA3-256, the best distinguisher in terms if #Rounds is still the classical SymSum. This is due to the fact that χ^{-1} is not applicable for SHA3-224 and SHA3-256 as the output hash value length is < 320 bits which is minimum requirement for applying χ^{-1} on the hash-digest. Another observation is that for SHA3-384/512 and SHAKE128/256, the maximum rounds are reached using χ^{-1} technique over classical SymSum. This is attributed to the degrees of freedom that is available when we just augment classical SymSum with χ^{-1} technique. On the other hand, linear structures lead to drastic reduction in degrees of freedom. Also, it can be

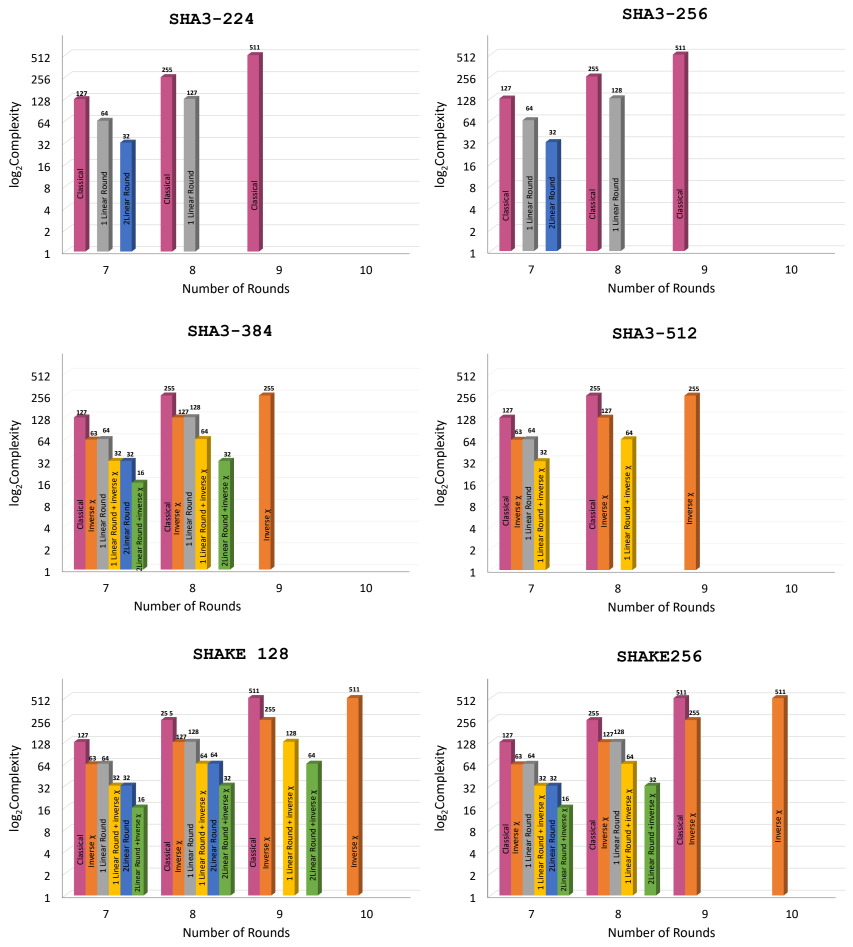


Fig. 8: Comparison of SHA3 variants for different approaches as applying χ^{-1}, 1-round linearization, 1-round linearization + χ^{-1}, 2-round linearization and 2-round linearization + χ^{-1} with classical SymSum distinguisher

noted that SymSum always enjoys a degree 2 advantage as predicted by the results discussed earlier. However, for the same number of round ZeroSum always has a better degree of freedom for well-understood reason of not having to conform to the self-symmetry constraint.

The maximum degree of freedom for different variants and approaches is depicted in Table 3. The table also shows the corresponding slice/state configuration for achieving that degree of freedom. Moreover, the constraints to be applied on the slice variables to fulfill the condition for 1-round linearization is also exhibited in the table. Similar data is furnished in Table 4 for 2-round linearization. It is worth mentioning that for SHA3-512 2-round linearization is

Table 3: Slice configuration, conditions and maximum degree of freedom for 1-round linearization of SHA3 variants. Orange, white and gray represent variable, constant and 0. Here we give one of the possible slice configurations and corresponding conditions and maximum degree of freedom.

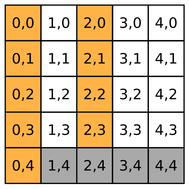
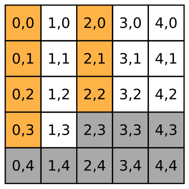
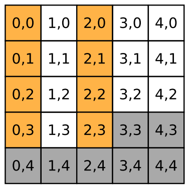
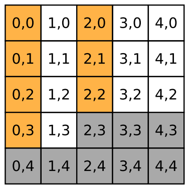
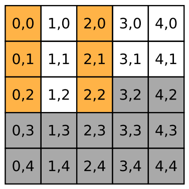
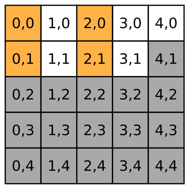
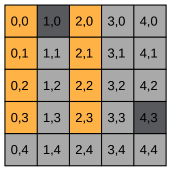
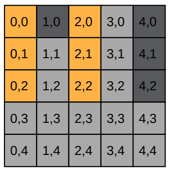
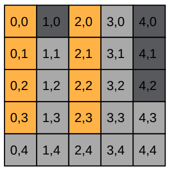
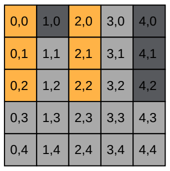
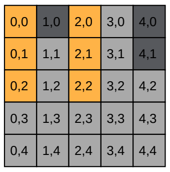
Variant	Slice Configuration	Restrictions on variables	Degree of Freedom
SHAKE128		$A[0,4] = \alpha_1 \oplus \sum_{i=0}^{3} A[0,i]$ $A[j,3] = \alpha_2 \oplus \sum_{i=0}^{2} A[j,i] \quad (j \in \{2,3\})$	2^{224}
SHAKE256		$A[0,3] = \alpha_1 \oplus \sum_{i=0}^{2} A[0,i]$ $A[j,2] = \alpha_2 \oplus \sum_{i=0}^{1} A[j,i] \quad (j \in \{2,3\})$	2^{160}
SHA3-224		$A[0,3] = \alpha_1 \oplus \sum_{i=0}^{2} A[0,i]$ $A[2,3] = \alpha_2 \oplus \sum_{i=0}^{2} A[2,i]$	2^{192}
SHA3-256		$A[0,3] = \alpha_1 \oplus \sum_{i=0}^{2} A[0,i]$ $A[j,2] = \alpha_2 \oplus \sum_{i=0}^{1} A[j,i] \quad (j \in \{2,3\})$	2^{160}
SHA3-384		$A[0,2] = \alpha_1 \oplus \sum_{i=0}^{1} A[0,i]$ $A[2,2] = \alpha_2 \oplus \sum_{i=0}^{1} A[2,i]$	2^{128}
SHA3-512		$A[0,1] = \alpha_1 \oplus A[0,0]$ $A[j,1] = \alpha_2 \oplus A[j,0] \quad (j \in \{2,3\})$	2^{64}

Table 4: Slice configuration, conditions and degree of freedom for 2-round linearization of `SHA3` variants. Orange, white, light gray and dark gray represent variable, constant, 0 and 1 respectively. Here we give one of the possible slice configurations and corresponding conditions and maximum degree of freedom. Note that this strategy is not applicable for `SHA3` 512

Variant	Slice Configuration	Restrictions on variables	Degree of Freedom
`SHAKE128`		$A[0,0] \oplus A[0,1] \oplus A[0,2] \oplus A[0,3] = \texttt{0xff...f}$ $A[2,0] \oplus A[2,1] \oplus A[2,2] \oplus A[2,3] = \texttt{0xff...f}$ $A[2,0]_{\lll 30} = A[0,0] \oplus A[2,2]_{\lll 11}$ $A[2,1]_{\lll 6} = A[0,1]_{\lll 4} \oplus A[2,3]_{\lll 15}$ $A[2,2]_{\lll 11} = A[0,2]_{\lll 3}$ $A[2,3]_{\lll 15} = A[0,3]_{\lll 9} \oplus A[2,0]_{\lll 30}$	2^{64}
`SHAKE256`		$A[i,0] \oplus A[i,1] \oplus A[i,2] = \texttt{0}, \quad i = 0,2$ $A[2,0]_{\lll 30} = A[0,0] \oplus A[2,2]_{\lll 11}$ $A[2,1]_{\lll 6} = A[0,1]_{\lll 4}$ $A[2,2]_{\lll 11} = A[0,2]_{\lll 3}$	2^{32}
`SHA3-224`		$A[0,0] \oplus A[0,1] \oplus A[0,2] \oplus A[0,3] = A[0,4] \oplus \texttt{0}$ $A[2,0] \oplus A[2,1] \oplus A[2,2] \oplus A[2,3] = \texttt{0}$ $A[2,0]_{\lll 30} = A[0,0] \oplus A[2,2]_{\lll 11}$ $A[2,1]_{\lll 6} = A[0,1]_{\lll 4} \oplus A[2,3]_{\lll 15}$ $A[2,2]_{\lll 11} = A[0,2]_{\lll 3}$ $A[2,3]_{\lll 15} = A[0,3]_{\lll 9} \oplus A[2,0]_{\lll 30}$	2^{64}
`SHA3-256`		$A[i,0] \oplus A[i,1] \oplus A[i,2] = \texttt{0}, \quad i = 0,2$ $A[2,0]_{\lll 30} = A[0,0] \oplus A[2,2]_{\lll 11}$ $A[2,1]_{\lll 6} = A[0,1]_{\lll 4}$ $A[2,2]_{\lll 11} = A[0,2]_{\lll 3}$	2^{32}
`SHA3-384`		$A[i,0] \oplus A[i,1] \oplus A[i,2] = \texttt{0}, \quad i = 0,2$ $A[2,0]_{\lll 30} = A[0,0] \oplus A[2,2]_{\lll 11}$ $A[2,1]_{\lll 6} = A[0,1]_{\lll 4}$ $A[2,2]_{\lll 11} = A[0,2]_{\lll 3}$	2^{32}

not applicable as the rate part is substantially lower leaving very less room to formulate the necessary constraints. It is easy to appreciate that the results reported here are better than ZeroSum and classical SymSum. Interestingly, even the simple χ^{-1} trick helps classical SymSum to breach the 10-round barrier (as stated in [17]) which is now possible to be distinguished with 2^{511} calls to SHA3.

7 Conclusion

This work aims to combine two very interesting results on SHA3 namely the SymSum property and the idea of linear structures to devise the best distinguishers on the SHA3 standard in terms of complexity and number of rounds penetrated. The main contribution lies in studying the effect of linearization on the core SymSum property. The results show that due to the effect of linear structures the factor of four advantage that SymSum enjoys over ZeroSum is reduced to two. Theoretical arguments are provided to explain this reduction. A simple χ inversion trick is also devised on applicable variants to penetrate one round further. With the combined power of all strategies, this work reaches up to 9 rounds of certain SHA3 variants with a practically feasible complexity of 2^{64}.

References

1. Aumasson, J.P., Meier, W.: Zero-sum distinguishers for reduced keccak-f and for the core functions of luffa and hamsi. rump session of Cryptographic Hardware and Embedded Systems-CHES **2009**, 67 (2009)
2. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: Sponge functions. Ecrypt Hash Workshop 2007 (May 2007)
3. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: The Keccak SHA-3 submission. Submission to NIST (Round 3) (2011), http://keccak.noekeon.org/Keccak-submission-3.pdf
4. Boura, C., Canteaut, A., Cannière, C.D.: Higher-order differential properties of keccak and *Luffa*. In: FSE. Lecture Notes in Computer Science, vol. 6733, pp. 252–269. Springer (2011)
5. Dinur, I., Morawiecki, P., Pieprzyk, J., Srebrny, M., Straus, M.: Cube attacks and cube-attack-like cryptanalysis on the round-reduced keccak sponge function. In: EUROCRYPT (1). Lecture Notes in Computer Science, vol. 9056, pp. 733–761. Springer (2015)
6. Duan, M., Lai, X.: Improved zero-sum distinguisher for full round keccak-f permutation. IACR Cryptology ePrint Archive **2011**, 23 (2011)
7. Duc, A., Guo, J., Peyrin, T., Wei, L.: Unaligned rebound attack: Application to keccak. In: FSE. Lecture Notes in Computer Science, vol. 7549, pp. 402–421. Springer (2012)
8. Guo, J., Liu, M., Song, L.: Linear structures: Applications to cryptanalysis of round-reduced keccak. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 10031, pp. 249–274 (2016)
9. Huang, S., Wang, X., Xu, G., Wang, M., Zhao, J.: Conditional cube attack on reduced-round keccak sponge function. In: EUROCRYPT (2). Lecture Notes in Computer Science, vol. 10211, pp. 259–288 (2017)

10. Jean, J., Nikolic, I.: Internal differential boomerangs: Practical analysis of the round-reduced keccak-f permutation. In: FSE. Lecture Notes in Computer Science, vol. 9054, pp. 537–556. Springer (2015)
11. Kuila, S., Saha, D., Pal, M., Chowdhury, D.R.: 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. pp. 88–108 (2014)
12. Li, T., Sun, Y., Liao, M., Wang, D.: Preimage attacks on the round-reduced keccak with cross-linear structures. IACR Trans. Symmetric Cryptol. **2017**(4), 39–57 (2017)
13. Liu, T., Sun, Y.: Preimage attacks on round-reduced keccak-224/256 via an allocating approach. IACR Cryptology ePrint Archive **2019**, 248 (2019)
14. Morawiecki, P., Pieprzyk, J., Srebrny, M.: Rotational cryptanalysis of round-reduced keccak. In: FSE. Lecture Notes in Computer Science, vol. 8424, pp. 241–262. Springer (2013)
15. Naya-Plasencia, M., Röck, A., Meier, W.: Practical analysis of reduced-round keccak. In: INDOCRYPT. Lecture Notes in Computer Science, vol. 7107, pp. 236–254. Springer (2011)
16. Qiao, K., Song, L., Liu, M., Guo, J.: New collision attacks on round-reduced keccak. IACR Cryptology ePrint Archive **2017**, 128 (2017)
17. Saha, D., Kuila, S., Chowdhury, D.R.: Symsum: Symmetric-sum distinguishers against round reduced SHA3. IACR Trans. Symmetric Cryptol. **2017**(1), 240–258 (2017)
18. of Standards, N.I., Technology.: SHA-3 : Cryptographic hash algorithm competition, http://csrc.nist.gov/groups/ST/hash/sha-3/index.html.