Higher Order Differential Analysis of NORX

Sourav Das, Subhamoy Maitra, and Willi Meier

[1] Infinera Corporation, Bangalore, India, Email: sourav10101976@gmail.com
[2] Indian Statistical Institute, Kolkata, India, Email: subho@isical.ac.in
[3] FHNW, Switzerland, Email: willi.meier@fhnw.ch

Abstract. In this paper, we analyse the higher order differential properties of NORX, an AEAD scheme submitted to CAESAR competition. NORX is a sponge based construction. Previous efforts, by the designers themselves, have focused on the first order differentials and rotational properties for a small number of steps of the NORX core permutation, which turn out to have quite low biases when extended to the full permutation. In our work, the higher order differential properties are identified that allow to come up with practical distinguishers of the 4-round full permutation for NORX64 and half round less than the full permutation (i.e., 3.5-round) for NORX32. These distinguishers are similar to zero-sum distinguishers but are probabilistic in nature rather than deterministic, and are of order as low as four. The distinguishers have very low complexities, and are significantly more efficient than the generic generalized birthday attack for the same configurations of zero-sums. While these distinguishers identify sharper non-randomness than what the designers identified, our results do not lend themselves for cryptanalysis of full-round NORX encryption or authentication.

Keywords: NORX, Authenticated Encryption, CAESAR, ARX, Higher-order Differential, Bias.

1 Introduction

NORX [2] is a candidate of the CAESAR competition that attempts to select some good schemes for authenticated encryption with associated data (AEAD). NORX is based on the sponge construction [6] in the duplex mode. The core permutation $\mathcal{F}$ of NORX is derived from the basic ideas of ChaCha and BLAKE. However, ChaCha and BLAKE use ARX primitives, i.e., they are based on integer Addition (A) modulo 2^n, bit Rotations (R) and XOR (X). However, the permutation $\mathcal{F}$ of NORX does not exploit the modulo addition. Here, the integer addition, which can be seen as $a + b = (a \oplus b) + ((a \wedge b) \ll 1)$, is replaced by the approximation $(a \oplus b) \oplus ((a \wedge b) \ll 1)$, a purely logic-based operation to increase hardware efficiency and simplify cryptanalysis. One can thus call this an LRX construction, where the modulo Addition (A) is replaced by Logical operator (L). Although, ChaCha and BLAKE have already resisted extensive analysis and are considered to be secure, this new LRX based permutation $\mathcal{F}$ still lacks in-depth study and its security level is yet to be explored.

The security proofs of the sponge construction [6] make the assumption that the underlying permutation is random which is so called "hermetic sponge strategy". With this, it is known to achieve a security bound of $\min(2^{c/2}, 2^{\kappa})$, where c is the capacity and κ is the key-size, for a keyed sponge variant. In Asiacrypt 2014 [11], this bound was improved to $\min(2^{b/2}, 2^c, 2^{\kappa})$ b is the permutation size. It was proven that NORX achieves this bound with the ideal permutation model. Although, it is known that there are some non-ideal phenomenon in the NORX permutation [3], they are actually "harmless" in the sense they occur very rarely and the authors show that they are not applicable for the cipher.

In [3], designers of NORX use an automated framework to find the best differential for the underlying permutations in its two versions NORX32 and NORX64. The first order differentials of the full permutations are described, which have quite low probability. With the full

permutation, the designers could obtain differentials of weight 12 after 1.5 rounds and no high probability (truncated) differentials were found after 2 rounds. In [3], distinguishers for the permutations were also described, for example, a rotational distinguisher of weight 106 for the full permutation which would require a complexity of 2^{106}.

Contribution and Outline of this Paper In this paper we obtain several interesting results related to higher order differential distinguishers for both NORX32 and NORX64, that were not known earlier. In Section 3, we present a broad framework for presenting our results. Our results lead us to a rotational distinguisher for two full rounds of NORX64 with initial states as proposed by the designers. This we present in Section 3.3. We continue with further results on NORX64 and NORX32, but we do not consider the intialization constants for these cases as proposed by the designers. That is we look into the round functions to identify several kinds of non-randomness. In Section 4 we show how to obtain a distinguisher with a 4-th order differential in full permutation of NORX64 with four rounds with a complexity of 2^9 only that should have required a complexity of 2^{44} in case of generalized birthday attack. Our distinguisher considers zero-sum of 100 bits at the input and 100 bits at the output. In Section 5, we obtain such distinguishers of NORX32 with 3.5 rounds with a complexity less than 2^{12}. Here, our distinguisher considers zero-sum of 70 bits each at the input and the output. The generalized birthday attack would have required 2^{32} complexity in this case. Further, in Section 5.1, we provide results for NORX32 that shows the existence of first order differential bias after two rounds. This bias becomes more prominent with a few conditions on the states. Note that the previous best results with NORX32 was only for 1.5 rounds [3].

The algebraic cryptanalysis of the function G by the designers (in Sect. 6.2 in the NORX document [2]) shows varying degrees of the ANFs of the component functions. A majority of component functions have degree 5 or 8 (for a half round). In view of these findings, one wouldn't expect biased higher order differentials of order as low as 4 for 2 rounds in forward direction (say). In backwards direction, its expected degrees of component functions spread more. For NORX64, we find strong bias with a 4-th-order differential after 1.75 rounds forward (where 0.25 round means a half of a G function) and 2.25 rounds backwards, that is for a total of 4 rounds. We acknowledge that our results do not threaten the security of NORX cipher itself and one may also note that there are 4-round and 6-round versions for a full NORX permutation. In this paper, we restrict ourselves to the 4-round version only. However, our results and (heuristic) explanations provide insights in the area and to the best of our knowledge, these results were not known earlier.

2 Background

In this section, we present some necessary background material that will be needed to follow this paper. We first describe NORX briefly. The NORX family of AEAD schemes is based on the monkeyDuplex construction [6] and parameterized by a word size $W \in 32, 64$, a round number $1 \leq R \leq 63$, a parallelism degree $0 \leq D \leq 255$ and a tag size $\mid A \mid \leq 10W$. The state S of NORX consists of sixteen words $s_0, \cdots, s_{15}$ each of size W bits, which are interpreted as a 4×4 matrix. Thus, the state has a size of 512 bits for $W = 32$ and a size of 1024 bits for $W = 64$. The state S is initialized by loading the nonce n_0, n_1, the key $k_0, \cdots, k_3$ and pre-defined constants $u_0, \cdots, u_9$ in the following way:

$$\begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} = \begin{pmatrix} u_0 & n_0 & n_1 & u_1 \\ k_0 & k_1 & k_2 & k_3 \\ u_2 & u_3 & u_4 & u_5 \\ u_6 & u_7 & u_8 & u_9 \end{pmatrix}$$

The round function $\mathcal{F}$ of NORX is composed of one column round followed by one diagonal round as described in Table 1.

Table 1. Description of One round NORX Permutation. For NORX32 and NORX64, the rotational constants are (8, 11, 16, 31) and (8, 19, 40, 63) respectively. Thus each of the column or diagonal round can be seen as a half-round. In case only first four steps (i)-(iv) or the last four (v)-(viii) of G function in either column or diagonal round is considered, we call that a quarter or 0.25 round.

Column round	Diagonal round	G function	
(i) $G(s_0, s_4, s_8, s_{12})$	(i) $G(s_0, s_5, s_{10}, s_{15})$	(i) $a \leftarrow (a \oplus b) \oplus ((a \wedge b) \ll 1)$	(v) $a \leftarrow (a \oplus b) \oplus ((a \wedge b) \ll 1)$
(ii) $G(s_1, s_5, s_9, s_{13})$	(ii) $G(s_1, s_6, s_{11}, s_{12})$	(ii) $d \leftarrow (d \oplus a) \lll r_0$	(vi) $d \leftarrow (d \oplus a) \lll r_2$
(iii) $G(s_2, s_6, s_{10}, s_{14})$	(iii) $G(s_2, s_7, s_8, s_{13})$	(iii) $c \leftarrow (c \oplus d) \oplus ((c \wedge d) \ll 1)$	(vii) $c \leftarrow (c \oplus d) \oplus ((c \wedge d) \ll 1)$
(iv) $G(s_3, s_7, s_{11}, s_{15})$	(iv) $G(s_3, s_4, s_9, s_{14})$	(iv) $b \leftarrow (b \oplus c) \lll r_1$	(viii) $b \leftarrow (b \oplus c) \lll r_3$

In the general set-up, we define a full permutation of NORX when all the state words are initially random (in actual cipher there are several constants and other constraints) and the permutation $\mathcal{F}$ is iterated over R rounds. This is the case when the cipher-texts are produced after the initialization is complete. There are versions of NORX with 4 or 6 rounds for the full permutation, but we focus on the 4-round version only throughout this paper. We denote the full permutation with 4 rounds as $\mathcal{F}^4$. For further information, the reader may refer to [2].

2.1 High-order Differential Distinguishers

Higher order differentials consider iterative derivatives [10]. Let us present the following definition in this regard.

Definition 1. *The derivative of a Boolean function f on n input variables with respect to a linear subspace V of F_2^n is defined as $\Delta_V f(x) = \bigoplus_{v \in V} f(x + v)$.*

If V has dimension d, $\Delta_V f$ is called a d-th order derivative. This concept easily generalizes to derivatives (or differentials) of functions from F_2^n to F_2^m, $m > 1$. First order derivatives δ_a with respect to a difference vector a correspond to ordinary differential cryptanalysis. One may note that [10] if $deg(f)$ denotes the non-linear degree of a Boolean function f, then, $deg(\delta_a f(y)) \leq deg(f(y)) - 1$. We also need the following definition.

Definition 2. ***[k-sums problem]*** *Given k lists of random n-bit values (for example, k distinct instances of a compression function $f_1, \cdots, f_k$), to find one value from each list such that the sum of the k values is zero. When $k = 2$, it is essentially the collision problem of hash functions.*

The k-sums problem can be solved in polynomial time (using the XHASH attack [5]) when $k \geq n$. However, the problem is believed to be hard for small k. One well known method for solving the k-sums problem with small k is Wagner's "generalized birthday" technique, which requires time and space $O(k \cdot 2^{\frac{n}{1+\log_2 k}})$ [13]. As an application, k-sums can be used to forge message authentication codes (MACs) [1]. The zero-sum distinguishers, as a special case of k-sums, were defined for the first time in [4] where if a set of inputs to a function sums to zero the corresponding output set also sums to zero.

Definition 3. *Let F be a function from F_2^n into F_2^m. A zero-sum for F of size K is a subset $x_1, \cdots, x_K \subseteq F_2^n$ of elements which sum to zero and for which the corresponding images by F also sum to zero, i.e., $\sum_{i=1}^{K} x_i = \sum_{i=1}^{K} F(x_i) = 0$.*

Assume the algebraic degree of F is n. Then a zero-sum is necessarily obtained by taking an n-th order derivative of F. From now on let $\mathcal{F}^i$ be an iterated permutation over i rounds. It can be noted that one can start in some intermediate state and calculate an n-th order differential backwards to get 2^n input states that may or may not sum to zero. This resembles the "known-key-model" [8]. Then one can go forward to obtain 2^n output states of $\mathcal{F}^i$. For example, to see how this is performed on Keccak, see [4], that lead Keccak authors to increase the number of rounds of Keccak to 24 (see [6]). The zero sum distinguishers for Keccak were improved later in [9, 7]. However, in the all the cases, the complexity has been well above the security claims of Keccak. An application of k-sums problem on the hash function Hamsi can be found in [1].

Instead of deterministic zero-sums, towards obtaining distinguishers, we shall often consider zero-sums that hold with some probability. In case the basis vectors for the subspace V in a higher order derivative are unit vectors, probabilistic zero-sums are called cube distinguishers.

3 Broad Frameworks of our Distinguishers

In this section we describe our directions to obtain the new distinguishers. We here discuss our ideas in a generic manner and later we use them specifically in case of NORX32 and NORX64.

3.1 Zero-sum Distinguishers for NORX

Here we describe a technique to generate zero-sum distinguishers for NORX. First, let us observe a few properties of the G function. Since the G function constitutes a half round of NORX, we define a quarter round of NORX as the first four state update steps of G (i.e. either column update or diagonal update of NORX) as described in Table 1. In the G function (or half round) such steps are repeated twice, whereas as per our description, the steps are not repeated in a quarter round.

Property 1. If the algebraic degree of the four inputs (a, b, c, d) of the G function are $deg(a) = p, deg(b) = q, deg(c) = r, deg(d) = s$, respectively, then after a quarter round of NORX the algebraic degrees of the output are at most: $deg(a) = p + q$, $deg(d) = \max(p + q, s)$, $deg(c) = \max(p + q, s) + r$, $deg(b) = \max(p + q, s) + r$.

For calculation of zero-sums in NORX, we choose an intermediate state where $s_0 = s_{12} = x$ and rest of the state words are zero. That is, we choose $a = d$ in the G function input of the first column and select the rest of the state words as 0. It can be observed that after the column transformation,

- the degree of each state variable in the first column is 1, i.e., linear and
- the rest of the state variables are still constant, i.e., zero.

Since, the algebraic degree is 1 for the first column, after the second quarter round, i.e., after the half round, two of the states (i.e., $a = s_0$ and $d = s_{12}$), will have a maximum algebraic degree 2 and rest two (i.e., $c = s_4$ and $b = s_8$) will have the maximum algebraic degree as 3. We calculated experimentally the zero-sum for the first column round. Since most of the bits in the column variable are still zero, we could get complete state sum to zero only after a 2nd order differential, i.e., with 2^4 inputs. The rest of the state words in the other 3 columns continue to be zeros.

With this initial state, we continue to find higher order differentials for two full rounds (i.e., four half rounds or four instances of G function). We could see visible non-randomness on state words $s_0, \cdots, s_3$ and $s_{12}, \cdots, s_{15}$ after 2^{10} iterations (10-th order differential). Those state words

summed to zero after 2^{26} iterations, i.e., for 26-th order differential. That is, if one takes all the 2^{26} states S_i with $x = i$, for $0 \leq i < 2^{26}$, then $\oplus_{i=0}^{2^{26}-1} \mathcal{F}^2(S_i)$ will have zero at the state words $(\mathcal{F}^2(S_i))_j$ for $j = 0, 1, 2, 3, 12, 13, 14, 15$. Further if one starts with a random 64-bit value ρ and considers all the 2^{26} states S_i with $x = \rho \oplus i$, for $0 \leq i < 2^{26}$, then also $\oplus_{i=0}^{2^{26}-1} \mathcal{F}^2(S_i)$ provides a significant number of zeros in the above state words.

Zero-sum Backwards We start with this same forward initial state and move backwards to find a zero sum. As mentioned before, the idea is to use this initial state as intermediate state and find the initial state words that sum to zero. The greatest difficulty in moving backwards is that the G function has an algebraic degree of 64 in each non-linear layer. However, if one state is linear and the other state is zero then the backward direction also remains linear for the nonlinear operation. As our intermediate state has many zeros, in going backwards two of the diagonals are completely zero and other two have only one linear variable in the state block. Thus, we could obtain certain non-randomness results while going backwards. Although, we could not see zero-sum on the entire state, we could see a lot of non-randomness where a large part of the states are zero after 1 and 1.5 rounds backwards.

Although, the above presents a new result of non-randomness for NORX64, the probability of occurrence is very low. This is because we set most of the intermediate state (or the starting state for moving forward and backwards) to zero to make the first forward round linear. This result adds to the list of non-randomness results but does not affect the security of the NORX AEAD scheme in practice.

3.2 Cube Distinguishers

In this case, we take the initial state as random and place n unit vectors on the nonce word to calculate the n-th order differential with the output sum of 2^n possible linear combination of those unit vectors. Let us go each half round to check how the cube distinguishers propagate.

0.5 Round After 2-nd order differential, from experiments, we observe a zero sum across the states. After 1-st order differential, all the state variables in the nonce column is non-zero and all the remaining columns are zero. We analyse the situation as follows. Since the states are random, the algebraic degree is 2 across the states with respect to nonce variable in the nonce column. Rest all states have algebraic degree 0, i.e., constant. This is expected, as the first half round only transforms the column-wise states.

1 Round At the second half round, the G transformation occurs on the diagonal. In each diagonal, one state variable has an algebraic degree 2 and rest of the state variables are constant, as in (i). After the first half of the G transformation i.e. after quarter rounds, the algebraic degree becomes as in (ii).

$$\text{(i)} Deg \begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} = \begin{pmatrix} 0\,2\,0\,0 \\ 0\,2\,0\,0 \\ 0\,2\,0\,0 \\ 0\,2\,0\,0 \end{pmatrix} \quad \text{(ii)} Deg \begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} = \begin{pmatrix} 2\,2\,0\,0 \\ 2\,2\,2\,2 \\ 2\,2\,2\,2 \\ 2\,2\,0\,2 \end{pmatrix}$$

After the second quarter round of the second half round, the degree matrix becomes as in (iii). Thus, after one round, we still have state variables with degree 2. Our experimental results conform with this finding, i.e., s_2, s_3, s_{13}, s_{14} sum to zero after second order differential. However, on 3-rd order differential, all state words except s_5, s_{10}, sum-up to zero implying these state

variables have maximum degree 3, with respect to nonce variables. On 4-th order differential, all the state words sum-up to zero, implying s_5, s_{10}, have algebraic degree 4 with respect to nonce bits. Thus, based on experiment, the actual degree matrix can be explained as in (iv).

$$\text{(iii)}Deg\begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} = \begin{pmatrix} 4\,4\,2\,2 \\ 4\,4\,6\,4 \\ 4\,4\,4\,6 \\ 4\,2\,2\,4 \end{pmatrix} \quad \text{(iv)}Deg\begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} = \begin{pmatrix} 3\,3\,2\,2 \\ 3\,4\,3\,3 \\ 3\,3\,4\,3 \\ 3\,2\,2\,3 \end{pmatrix}$$

1.5 Rounds In this round, again the column transformation takes place. After 1.25 rounds, the degree evolution becomes:

$$Deg\begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix} = \begin{pmatrix} 6\ \ 7\ \ 5\,5 \\ 9\,10\,9\,8 \\ 9\,10\,9\,8 \\ 6\ \ 7\ \ 5\,5 \end{pmatrix}$$

The experimental results were different in different runs. On 4-th order differential, in many cases the entire first row (s_0, s_1, s_2, s_3) and the entire last row summed to zero. However, in a few runs, s_0 and s_{12} became non-zero with only one bit set and were rotations of each other. On 5th order differential, at certain runs the entire states became zero-sum, but at certain runs some of the state words in the second and the third row became non-zero with low weight and were rotations of each other.

This is inconclusive on the degree in random experiments. However, this variation can be explained due to varied diffusion with respect to different constant values in each experiment. In the nonlinear operation, if one of the variables is a constant and zero, then the nonlinear component does not add to diffusion of the bits. However, if the constant is all ones, then the nonlinear part also provides diffusion of the variable bits. After 2 rounds, we did not get any biased bits for the NORX64 cipher. However, we got a very special property after 2 rounds in NORX64 as described in the next section.

3.3 Rotational Properties in the Higher Order Differential

In our experiments, we found that after a certain order of differentials, the state words in the last row and the first row become rotations of each other. This can be explained as follows. We can see that after each quarter round, $deg(b) > (deg(d), deg(a))$. On the next quarter round, when the state a is updated, $deg(a)$ becomes $deg(a) + deg(b)$. Now, if we calculate a differential of order $(deg(d))$, we get a zero-sum. When we calculate the sum for the next quarter round with $deg(d)$-order differential, in the equation $(d = (a \oplus d) \ggg r_0)$, it (i.e., sum of d) simply becomes the rotation of a.

Let us now present the experimental results after 2 full rounds. we consider the initialization of NORX64 exactly. We take random values in all the 6 words (two nonce words and the four key words), the rest ten words are assigned to constants as explained in the design. We put the differentials in the 0-th nonce word, i.e., we consider s_1. Say we have a random 64-bit value ρ and we consider all the 2^d states S_i with $s_1 = \rho \oplus i$, for $0 \leq i < 2^d$. Then we obtain significant non-randomness in

$$\mathcal{R}^d_{0,15} = (\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_0 \oplus ((\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_{15} \lll 40),\ \mathcal{R}^d_{1,12} = (\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_1 \oplus ((\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_{12} \lll 40),$$
$$\mathcal{R}^d_{2,13} = (\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_2 \oplus ((\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_{13} \lll 40),\ \mathcal{R}^d_{3,14} = (\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_3 \oplus ((\oplus_{i=0}^{2^d-1}\mathcal{F}^2(S_i))_{14} \lll 40).$$

For perfectly random scenario, we should have values very close to 32 for $wt(\mathcal{R}^d_{0,15})$,$wt(\mathcal{R}^d_{1,12})$, $wt(\mathcal{R}^d_{2,13})$ and $wt(\mathcal{R}^d_{3,14})$. However, we can observe a bias from $d = 1$ itself for the event $s_2 = s_{13} \lll 40$. In fact, with the average of 2^{30} experiments, we obtain $wt(\mathcal{R}^1_{2,13}) = 31.968$, which can be clearly distinguished from 32. The bias starts increasing as d increases. We obtain an almost complete rotation between s_2 and s_{13}, i.e., $s_2 = s_{13} \lll 40$, with the 24-th order differential, i.e., with a complexity 2^{24}. Our experiment with 2^{14} trials provides $wt(\mathcal{R}^{24}_{2,13}) = 0.013$. Clear biases can also be observed for $wt(\mathcal{R}^d_{0,15})$ from $d = 3$, for $wt(\mathcal{R}^d_{1,12})$ from $d = 11$ and for $wt(\mathcal{R}^d_{3,14})$ from $d = 5$. As we see that the rotational constant r_2 in NORX specification document for NORX64 is 40, the experimental result directly matches the observation above.

Although, the above results do not directly lead to any immediate attack, it shows that the hidden capacity part of NORX64 is not really hidden and an attacker can extract some information about the hidden capacity part after two rounds of NORX. In particular, this experiment refers to initialization. The initial state consists of constants, key words, and two nonce words, at positions s_1, s_2 (see Section 2) which are the only words that are varied. The NORX initialization considers 8 rounds in total. The designers could obtain differentials of weight 12 after 1.5 rounds and no high probability (truncated) differentials were found after 2 rounds [3]. Our results clearly improve that analysis.

4 Distinguisher for Four rounds of the NORX64 Permutation

In this section we consider the results for NORX64. We do not consider the initialization restrictions described by the designers. However, we consider the round operations exactly as described for the scheme.

4.1 Forward and Reverse Biases

As noted by the designers [2], for full permutation of NORX, the diffusion becomes minimal in G function if one starts with an initial difference in the MSBs of the inputs a, b, c. As the G function operates on each column on the first half round, we can have four such combinations to make the diffusion minimal after the first half round. We take each of these sets (i.e., MSBs of a, b, c) as a basis vector for calculating higher order differentials. Denote the MSB of each state word s_i as μ_i. Thus we have four basis vectors as $\{(\mu_0, \mu_4, \mu_8), (\mu_1, \mu_5, \mu_9), (\mu_2, \mu_6, \mu_{10}), (\mu_3, \mu_7, \mu_{11})\}$. With the above four basis vectors, we run 4-th order differential for two rounds forward starting with a random state. That is we start from 16 states and run certain rounds to obtain states after those rounds and then xor the states bitwise. The j-th bit of the i-th state word ($0 \leq i \leq 15,\ 0 \leq j \leq 63$) is referred by $u = (64i + j)$-th bit. The probability that the u-th bit will be zero is denoted by ϕ_u in forward direction and β_u in backward direction. We observe significant biases in many of the output bits. Similarly, with the same initial random state and with the same basis vectors, we ran 4-th order differential with two rounds backwards for NORX64. Here also we observed quite a large number of biased bits.

Improvement of Biases We notice that the the number of biased bits in the forward 2 rounds is quite less than the 2 rounds backwards. Observing this fact, we made the intermediate state in the middle of the G function. We start at the middle of a row round in this manner. Then, we ran the experiments with 1.75 rounds forward (half row round, and then diagonal round, row round and diagonal round) and 2.25 rounds backward (half row round, diagonal round, row round, diagonal round and row round). We could see that the number of biased bits in the forward direction increases drastically and at the same time we still have enough number of biased bits in the backward direction to mount a distinguisher.

Let us now explain our experimental results. With an experiment of 2^{26} samples, we note that in the forward direction, after 1.75 round, there are 292 bits which are 0 with probability 1. Further there are quite significant biases in 100 bits for the backward direction with 2.25 rounds and we get 100 bits which are 0 with probability ≥ 0.75. Together, The probability that each of these 100 bits in the forward direction and each of these 100 bits in the backward direction is $2^{-8.95}$. Note that this is much higher than the case if one would consider the individual probabilities of the bits and multiplied them by considering independence. However, there is dependence among the bits and we obtain much higher joint probability of all those bits to be zero together. Thus, we can easily obtain 16 states such that XORing them we get 100 specific bits to be zero at the input and after four rounds of application of $\mathcal{F}$, XORing them we get 100 specific bits to be zero at the output with a probability of $2^{-8.95}$. That is, in expected $2^{8.95}$ attempts, one may find the zero-sum over those bits with certainty.

4.2 Method to mount the Distinguisher

We may exploit the biased bits in forward rounds and backward rounds to come up with a distinguisher of full $\mathcal{F}^4$. We do this in line with the known-key distinguishers as in [12, 8] where the attacker starts in the intermediate state of the permutation ("known-key") of the block cipher and performs encryption and decryption simultaneously to come-up with a distinguisher. We select a suitable set of bits (sums) in both input and output with large enough bias. An optimal event would be the one in which all selected bits follow the value (0 or 1) as suggested by the bias both in the forward and the backward direction. This is very similar to zero-sums analysis performed for Keccak hash function [4].

While zero-sum is deterministic, our approach is probabilistic. Our findings use 4-th order differentials starting from an intermediate state, and we compute backwards 2.25 rounds and forwards 1.75 rounds. Hence we select a subset of bit positions in input and output of $\mathcal{F}^4$ according to our finding with 4-th order differential. In zero-sum, we have to do only one computation forwards and backwards, as sums are deterministically 0. In our case, we need to do a trial and error kind of (partial) zero-sum. The step wise procedure is just iterating our experiment several times. To do this, (i) we first choose an intermediate state; (ii) with this state, we do 4-th order forwards computation for certain rounds (here 1.75); (iii) again, with this intermediate state, we do 4-th order backwards computation for certain rounds (here 2.25); until we have a success. "Success" here means that a suitable number of previously selected bit sums in the input and bit sums in the output are simultaneously "correct".

Now, let us look into the complexities of generic distinguishers. In all our subsequent analysis, we will select the number of bits more than the number of summands k (k is 16 for 4-th order differential). If $k < n$, then XHASH [5] algorithm is believed to be hard. Thus, we need to use generalized birthday attack [13] for our complexity comparison. As described in [7, Page 3] (following [13]), in this case the complexity of generalized birthday attack is $O(k \cdot 2^{\frac{2n}{1+\log_2 k}})$.

To compare with the k-sums problem, we note that, here $k = 16$ and $n = 100$. As $k < n$, with generalized birthday attack, the complexity is $O(k \cdot 2^{\frac{2n}{1+\log_2 k}})$ and here $k \cdot 2^{\frac{2n}{1+\log_2 k}}$ is $16 \cdot 2^{\frac{200}{1+\log_2 16}} = 2^{44}$, whereas our required complexity is $\frac{1}{2^{-8.95}} = 2^{8.95} \approx 2^9$ only. This complexity is much smaller than the generalized birthday attack. Thus, using the biased bits, we can formulate a distinguisher as outlined above.

5 Distinguishers for 3.5 Rounds of NORX32 Permutation

We performed the same experiment as above with NORX32. We selected the same basis vectors as in NORX64 and performed 4-th order differential with $k = 16$. We performed the experiment

with 1.75 rounds forward and 2.25 rounds backwards. However, in this case, the sum bits were better balanced than NORX64 and we are not successful in obtaining better results than the generic distinguishers.

Thus here we consider 3.5 rounds instead of 4 rounds for NORX32, i.e., 1.5 rounds forward (row round, diagonal round and then row round) and 2 rounds backward (diagonal round, row round, diagonal round and row round). With an experiment of 2^{26} samples, we note that in the forward direction, after 1.5 round, there are 304 bits which are 0 with probability 1. Further there are quite significant biases in the backward direction with 2 rounds and we get 70 bits which are 0 with probability ≥ 0.76. The joint probability of 70 bits each in the forward direction and the reverse direction to hold good becomes $p = 2^{-11.73}$ on an average with an experiment with 2^{26} samples. Thus our complexity requirement is $2^{11.73}$, which is less than 2^{12}.

Note that in this case, $k = 16$ and $n = 70$, i.e., $k < n$. Thus XHASH algorithm is likely to be hard to solve this. Using Wagner's generalized birthday attack, the complexity of finding such a distinguisher is $O(k \cdot 2^{\frac{2n}{1+\log_2 k}})$ and in this case $k \cdot 2^{\frac{2n}{1+\log_2 k}}$ is $16 \cdot 2^{\frac{140}{1+\log_2 16}} = 2^{32}$.

5.1 First-order Differential Analysis of NORX32 after Two Rounds

In NORX analysis paper [3], the best found differential characteristic propagates upto 1.5 rounds. In this section, we propose an improvement where we observed differential bias after two rounds by taking conditional differences. We start with a random state and introduce differences in the 0, 4, 8 and 12-th words as 0x00000c00, 0x80000400, 0x80000000, 0x80000000 respectively. We note that there are 20 bits (out of the 512 state bits), where the biases are ≥ 0.01, with the maximum bias having the value around 0.045. In fact, the bias becomes more prominent if we can set the bits 8, 9 of the 0-th word and bits 10, 11 of the 4-th word to zero. With this setting, we got 42 bits in the output after two rounds with a bias ≥ 0.01, with maximum bias of 0.18. These we note with an average of 2^{27} instances.

6 Conclusion

An extensive analysis of higher order differential properties of the basic permutation underlying the sponge based NORX AEAD scheme is provided. A well established type of distinguishers for key-less permutations based on high-order differentials are zero-sums. For the permutations, in both NORX64 as well as NORX32, deterministic zero-sum distinguishers are not applicable, but it is demonstrated that statistical variants of zero-sums allow to distinguish the full permutations with complexity lower than the known generic distinguisher, namely the generalized birthday attack. The results of this paper are not applicable to the full-round encryption or authentication mode of NORX, and thus present no threat to the security of the NORX AEAD scheme. However, several of our findings are newer results than what obtained by the designers and our results may serve for comparison of the characteristics as detected with those of other sponge based AEAD schemes. To our understanding, for the 4-round scenario, the security margin of NORX32 seems to be better than that of NORX64.

Acknowledgements: The authors like to acknowledge J. P. Aumasson, P. Jovanovic and S. Neves, the designers of NORX for their valuable feedback on this work.

References

1. J. P. Aumasson et al. Distinguishers for the Compression Function and Output Transformation of Hamsi-256. ACISP 2010. Available at: https://eprint.iacr.org/2010/091.pdf.

2. J. P. Aumasson, P. Jovanovic and S. Neves. NORX. https://norx.io
3. J. P. Aumasson, P. Jovanovic and S. Neves. Analysis of NORX: Investigating Differential and Rotational Properties. https://eprint.iacr.org/2014/317.
4. J. P. Aumasson and W. Meier. Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. NIST mailing list, 2009. Available at: http://aumasson.jp/data/papers/AM09.pdf.
5. M. Bellare, D. Micciancio. A new paradigm for collision-free hashing: Incrementality at reduced cost. In: Fumy, W. (ed.) EUROCRYPT. LNCS, vol. 1233, pp. 163-192. Springer. 1997.
6. G. Bertoni, J. Daemen, M. Peeters and G. V. Assche. Cryptographic sponge functions. Available at : http://sponge.noekeon.org/.
7. C. Boura and A. Canteaut. Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak-f and Hamsi-256. Selected Areas in Cryptography. Lecture Notes in Computer Science Volume 6544, 2011, pp 1-17.
8. H. Gilbert. A Simplified Representation of AES. Asiacrypt 2014.
9. M. Duan and X. Lai. Improved Zero-Sum Distinguisher for Full Round Keccak-f Permutation. Cryptology ePrint Archive, Report 2011/023, 2011.
10. X. Lai. Higher Order Derivatives and Differential Cryptanalysis. In Richard E. Blahut, Daniel J. Costello Jr., Ueli Maurer, and Thomas Mittelholzer, editors, Communications and Cryptography: Two Sides of One Tapestry, pp. 227 – 233. Kluwer Academic Publishers, 1994.
11. P. Jovanovic, A. Luykx and B. Mennink. Beyond $2^{c/2}$ Security in Sponge-Based Authenticated Encryption Modes. Asiacrypt, 2014. Available at: https://eprint.iacr.org/2014/373.
12. L. R. Knudsen and V. Rijmen. Known-Key Distinguishers for Some Block Ciphers. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 315–324. Springer, Heidelberg (2007).
13. D. Wagner. A generalized birthday problem. In: Yung, M. (ed.) CRYPTO. LNCS, vol. 2442, pp. 288–303. Springer. 2002.