# Distinguishing Attack on NORX Permutation

Tao Huang, Hongjun Wu

School of Physical and Mathematical Sciences
Nanyang Technological University
Singapore

5 March, 2018

FSE 2018, Belgium

# Outline

# NORX

- Authenticated encryption algorithm
- Designed by Aumasson, Jovanovic and Neves
- One of the 15 third-round CAESAR candidates
- Efficient in both software and hardware
- Current version is NORX v3.0
- Five instances (priority from highest to lowest):
  - NORX64-4-1
  - NORX32-4-1
  - NORX64-6-1
  - NORX32-6-1
  - NORX64-4-4 (parallel mode)

# Overview of NORX

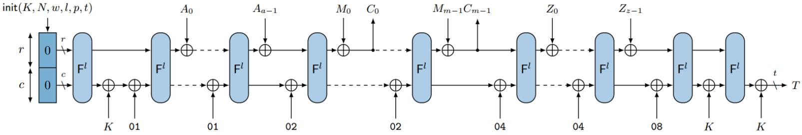



Figure: The layout of NORX construction in version 3.0 (from [AJN16])

# Overview of NORX

- NORX state words (red indicates the capacity words):

$$\begin{bmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{bmatrix}$$

- NORX state size:
  - NORX64: 1024-bit
  - NORX32: 512-bit

# The NORX Permutation

- $\mathtt{F}^l$: round function, $l = 4$ or $6$.
- $\mathtt{F}$ processes state by
  1. Column step ($\mathtt{F}_{col}$):

     $$\mathtt{G}(s_0, s_4, s_8, s_{12}), \mathtt{G}(s_1, s_5, s_9, s_{13}), \mathtt{G}(s_2, s_6, s_{10}, s_{14}), \mathtt{G}(s_3, s_7, s_{11}, s_{15}),$$

  2. Diagonal step ($\mathtt{F}_{diag}$):

     $$\mathtt{G}(s_0, s_5, s_{10}, s_{15}), \mathtt{G}(s_1, s_6, s_{11}, s_{12}), \mathtt{G}(s_2, s_7, s_8, s_{13}), \mathtt{G}(s_3, s_4, s_9, s_{14}).$$

# The NORX Round Function

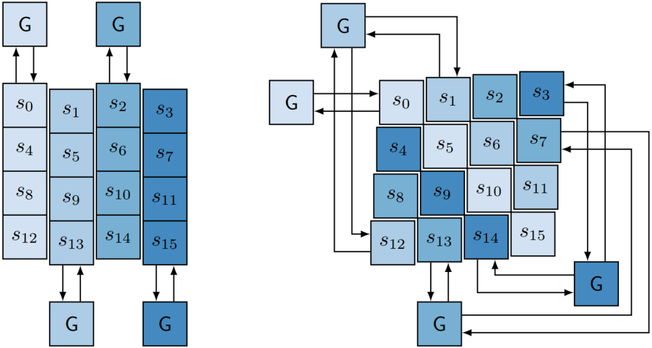



Figure: Column step and diagonal step of F (from [AJN16])

# The $\mathtt{G}$ Function

- The function $\mathtt{G}(a, b, c, d)$ computes the following 8 operations:

$$\begin{array}{ll} 1.\ a \leftarrow \mathtt{H}(a, b), & 2.\ d \leftarrow (a \oplus d) \ggg r_0, \\ 3.\ c \leftarrow \mathtt{H}(c, d), & 4.\ b \leftarrow (b \oplus c) \ggg r_1, \\ 5.\ a \leftarrow \mathtt{H}(a, b), & 6.\ d \leftarrow (a \oplus d) \ggg r_2, \\ 7.\ c \leftarrow \mathtt{H}(c, d), & 8.\ b \leftarrow (b \oplus c) \ggg r_3, \end{array}$$

  where $\mathtt{H}(x, y) = (x \oplus y) \oplus ((x \wedge y) \ll 1)$
- Notation: quarter round
  - $\mathrm{F}_{col} \to \mathrm{F}_{colH} + \mathrm{F}_{colL}$
  - $\mathrm{F}_{diag} \to \mathrm{F}_{diagH} + \mathrm{F}_{diagL}$

# Security Bounds for NORX

- NORX is based on monkeyDuplex mode.
- With security proof, the NORX mode of operation achieves security levels of $\min\{2^{b/2}, 2^{c}, 2^{k}\}$ assuming **an ideal underlying permutation**.

# Previous Cryptanalysis on NORX Permutation

- Aumasson et al. [AJN15] analysed the differential property of the NORX core permutation when differences can **only be introduced in the nonce**.
  - 4 round permutation: $2^{-836}$ for NORX64 and $2^{-584}$ for NORX32.
- Das et al. [DMM15] analysed the higher order differential properties of the NORX core permutation.
  - Zero-sum distinguishers for 4-round NORX64 permutation and 3.5-round NORX32 permutation
  - Require **chosen intermediate states**, computing 4-th order differential backward for 2.25 rounds and forward by 1.75 rounds.

# Previous Cryptanalysis on NORX Permutation

- Chaigneau et al. [CFG+17] proposed an attack on the full primitive of NORX v2.0.
  - The attack exploited a structural property that the 4 columns are rotationally identical in NORX permutation.
- Biryukov et al. [BUV17] analysed the NORX core permutation using symmetric truncated differentials.
  - 2.125-round distinguishers for both NORX32 and NORX64

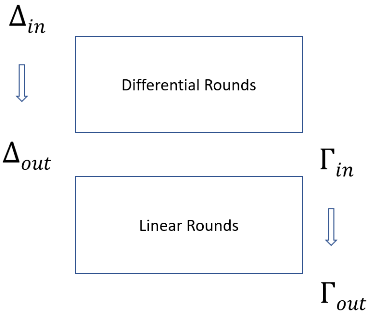
# Summary of Our Results

- A new distinguishing attack on the 4-round NORX permutation with low complexity.
- NORX64 permutation
  - Time complexity: $2^{48.5}$
  - Memory complexity: negligible
- NORX32 permutation
  - Time complexity: $2^{64.7}$
  - Memory complexity: negligible

# Distinguishing Attack on NORX64 Permutation

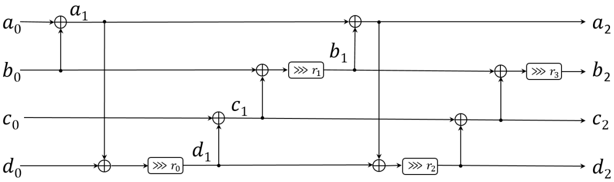
# Differential-Linear Attacks

- Proposed by Langford and Hellman in 1994
- Query messages with $\Delta_{in}$ and analyse the statistics of the XORed differences of $\Gamma_{out}$



# Constructing Linear Characteristic

- Linear approximation of the G Function
  - Remove the non-linear operation AND
  - Derive the expressions of the input $a_0$, $b_0$, $c_0$ and $d_0$ of the G function, in terms of the output $a_2$, $b_2$, $c_2$ and $d_2$



# Linear Approximation of the G

Table: Biases of the linear approximation for $i$-th bit of G function.

| | $i = 0$ | $i = 1$ | $i > 1$ |
|---|---|---|---|
| Bias of $a[i]$ | $2^{-1}$ | 0 | $2^{-5}$ |
| Bias of $b[i]$ | $2^{-1}$ | $2^{-2}$ | $2^{-2}$ |
| Bias of $c[i]$ | $2^{-1}$ | 0 | $2^{-4}$ |
| Bias of $d[i]$ | $2^{-1}$ | $2^{-2}$ | $2^{-2}$ |

# Searching for Linear Characteristic

- Consider $\Gamma_{in}$ has only 1 active bit.
- Position 0 has the best bias.
  - Only $a[0], b[0], c[0], d[0]$ need to be considered.
- When $c[0]$ is active, the largest biased can be obtained for 1.25-round NORX64 permutation, which is $2^{-8}$.

# An Example of the 1.25-round Linear Characteristic

- $\Gamma_{in}$:

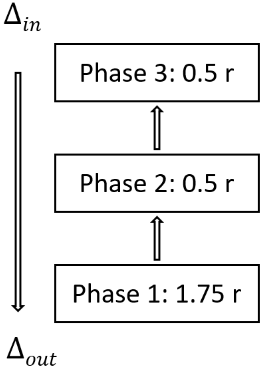
| | | | |
|---|---|---|---|
| 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 |
| 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 |
| 0x0000000000000000 | 0x0000000000000001 | 0x0000000000000000 | 0x0000000000000000 |
| 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 |

- $\Gamma_{out}$:

| | | | |
|---|---|---|---|
| 0x0000000000000001 | 0x0000000000000001 | 0x0000000001010000 | 0x0000000000010101 |
| 0x0000000000020000 | 0x0000000000000000 | 0x0000c00000000002 | 0x0200404002000000 |
| 0x0002020001000001 | 0x0000000000010001 | 0x0000000000000001 | 0x0000600000000002 |
| 0x0000000000000003 | 0x0100010000010001 | 0x0000000101000001 | 0x0000000001000001 |

# Constructing Differential Characteristic

- Target to 2.75-round differential characteristic
- Overview of the differential characteristic



# Differential Characteristic in Phase 1

- Input: 1-bit arbitrary input difference
- Output: difference on bit $s_9[0]$ with largest bias
- Rounds: 1.75-round
- Result:

$$s_{10}[17] \xrightarrow{\mathrm{F}_{diagH} \circ \mathrm{F}_{col} \circ \mathrm{F}_{diag} \circ \mathrm{F}_{col}} s_9[0]$$

- Bias: $-2^{-3.9}$

# Differential Characteristic in Phase 2

- Propagate the 1-bit difference $s_{10}[17]$ backwards.
- Linear approximation of the H function is used
- Rounds: 0.5-round
- Probability: $2^{-5}$

Table: Input difference in Phase 2.

| | | | |
|---|---|---|---|
| 0x0000001000000000 | 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 |
| 0x0000000000000000 | 0x0000001000020000 | 0x0000000000000000 | 0x0000000000000000 |
| 0x0000000000000000 | 0x0000000000000000 | 0x0000000000020000 | 0x0000000000000000 |
| 0x0000000000000000 | 0x0000000000000000 | 0x0000000000000000 | 0x0000000000020000 |

# Differential Characteristic in Phase 3

- Propagate input difference in Phase 2 backward.
- Linear approximation of the `H` function is used
- Rounds: 0.5-round
- Probability: 1

Table: Input difference in Phase 3 $(\Delta_{in})$.

| | | | |
|---|---|---|---|
| 0x0000001000000000 | 0x0040000000010000 | 0x0000001000000000 | 0x0000000000020000 |
| 0x0000000000000000 | 0x0040000800000000 | 0x0000001000020000 | 0x0000000000020000 |
| 0x0000001000000000 | 0x0000000000000000 | 0x0000000000020000 | 0x0000000000020000 |
| 0x0000101000000000 | 0x0000000800010000 | 0x0000000000020000 | 0x0000000000000002 |

- **Derive initial conditions**

# Derive Initial Conditions in Phase 3

- By setting conditions on the initial state, the probability that the linear approximations of H function hold can be 1.
- Example 1: $\mathbf{a_1} = \mathtt{H}(\mathbf{a_0}, \mathbf{b_0})$
  - $\Delta a_0 = 0x0000001000000000$.
  - $\Delta b_0 = 0x0000000000000000$.
  - $\mathtt{H}(a_0, b_0) = a_0 \oplus b_0$ holds with probability 1 when $b_0[36] = 0$.
- Example 2: $\mathbf{c_1} = \mathtt{H}(\mathbf{c_0}, \mathbf{d_1})$
  - $\Delta c_0 = 0x0000001000000000$.
  - $\Delta d_1 = 0x0000001000000000$.
  - $\mathtt{H}(c_0, d_1) = c_0 \oplus d_1$ holds with probability 1 if:

$$
\begin{aligned}
1 =& c_0[36] \oplus d_1[36] \\
=& c_0[36] \oplus d_0[44] \oplus a_1[44] \\
=& c_0[36] \oplus d_0[44] \oplus a_0[44] \oplus b_0[44] \oplus (a_0[43] \wedge b_0[43]).
\end{aligned}
$$

  - Conditions: $a_0[43, 44] = 0,\ b_0[44] = 0,\ c_0[36] = 0,\ d_0[44] = 1$

# Derive Initial Conditions in Phase 3

Table: Conditions on the initial state for NORX64.

| | |
|---|---|
| Column 0 | $s_0[43, 44, 62, 63] = 0$, $s_4[36, 44, 55, 63] = 0$, $s_8[36, 54, 55] = 0$, $s_{12}[44] = 1$, $s_{12}[63] = 0$ |
| Column 1 | $s_1[15, 16, 34, 35, 42, 43, 54, 61, 62] = 0$, $s_5[16, 35, 43, 54, 62] = 0$, $s_9[35, 54] = 1$, $s_9[53] = 0$, $s_{13}[43, 62] = 0$ |
| Column 2 | $s_2[0, 1, 16, 17, 19, 20, 24, 25, 43, 44, 55, 56, 57] = 0$, $s_2[36] = 1$, $s_6[1, 12, 17, 20, 25, 36, 44, 56, 57] = 0$, $s_{10}[11, 12, 35] = 0$, $s_{10}[36] = 1$, $s_{14}[1, 20, 25, 44] = 0$ |
| Column 3 | $s_3[0, 1, 17, 19, 20, 24, 25, 55, 56, 57] = 0$, $s_7[1, 12, 20, 25, 56, 57] = 0$, $s_7[17] = 1$, $s_{11}[11, 12, 16] = 1$, $s_{11}[17, 57] = 1$, $s_{15}[1, 20, 25] = 0$ |

# The Differential-Linear Characteristic

- $\Delta_{in}$:

| | | | |
|---|---|---|---|
| 0x0000001000000000 | 0x0040000000010000 | 0x0000001000000000 | 0x0000000000020000 |
| 0x0000000000000000 | 0x0040000800000000 | 0x0000001000020000 | 0x0000000000020000 |
| 0x0000001000000000 | 0x0000000000000000 | 0x0000000000020000 | 0x0000000000020000 |
| 0x0000101000000000 | 0x0000000800010000 | 0x0000000000020000 | 0x0000000000000002 |

- $\Gamma_{out}$:

| | | | |
|---|---|---|---|
| 0x0000000000000001 | 0x0000000000000001 | 0x0000000001010000 | 0x0000000000010101 |
| 0x0000000000020000 | 0x0000000000000000 | 0x0000c00000000002 | 0x0200404002000000 |
| 0x0000202001000001 | 0x0000000000010001 | 0x0000000000000001 | 0x0000600000000002 |
| 0x0000000000000003 | 0x0100010000010001 | 0x0000000101000001 | 0x0000000001000001 |

- Differential-linear bias: $-2^{-22.9}$

# The Differential-Linear Distinguisher for NORX64 Permutation

- The distinguishing attack procedure:
  1. Query $2^{47.5}$ pairs of 1024-bit message with difference $\Delta_{in}$ and initial conditions.
  2. For each pair of output from the oracle, compute the XORed sum of bits in $\Gamma_{out}$.
  3. Count the number $X$ that is the number of pairs such that the XORed sum is 0.
  4. If $X < 2^{46.5} - 2^{23.6}$, the oracle is the NORX64 permutation. Otherwise, the oracle is a random permutation.
- Complexity:
  - Time: $2^{48.5}$
  - Memory: Negligible
- Probability of success: 96%

# Experimental Results for NORX64 Permutation

- Environment: GPU server with 4 Tesla K-40 GPUs
- Generate $2^{47.49}$ pairs of random input with the initial condition and difference specified by the differential-linear characteristic
- Time: 63.1 hours
- The bias on the output bits is $-2^{-22.88}$
- Very close to the estimated bias $-2^{22.9}$.

# Distinguishing Attack on NORX32 Permutation

# Distinguishing Attack on NORX32 Permutation

- Similar method can be applied to NORX32 permutation
- Differential-linear characteristic
  - $\Delta_{in}$:

| | | | |
|---|---|---|---|
| 0x00000020 | 0x00000010 | 0x0400a020 | 0x02014020 |
| 0x00000020 | 0x00010030 | 0x0400a020 | 0x02010020 |
| 0x00200020 | 0x00110030 | 0x2020a030 | 0x00000000 |
| 0x20000000 | 0x11010020 | 0x20801020 | 0x00006000 |

  - $\Gamma_{out}$:

| | | | |
|---|---|---|---|
| 0x00000001 | 0x00000001 | 0x00010100 | 0x00000100 |
| 0x00000200 | 0x00000000 | 0x00c00002 | 0x02424000 |
| 0x00212001 | 0x00000101 | 0x00000001 | 0x00600002 |
| 0x00000003 | 0x00000101 | 0x00000001 | 0x00010001 |

  - Differential-linear bias: $-2^{-31.2}$

# Initial Conditions

Table: Conditions on the initial state for NORX32.

| | |
|---|---|
| Column 0 | $s_0[5] = 1,\ s_0[7, 8, 12, 13, 19, 20, 21, 28, 29] = 0,$<br>$s_4[0, 5, 8, 13, 20, 21, 29] = 0,$<br>$s_8[0, 4] = 0,\ s_8[5, 21] = 1,$<br>$s_{12}[8, 13, 29] = 0$ |
| Column 1 | $s_1[2, \ldots, 8, 11, 12, 13, 16, 18, 19, 20, 21, 23, 24, 27, 28, 29] = 0,$<br>$s_5[0, 3, 5, 7, 8, 12, 13, 16, 19, 20, 21, 24, 27, 28] = 0,\ s_5[4] = 1,$<br>$s_9[0, 3, 15, 26, 27, 30, 31] = 0,\ s_9[4, 16, 20] = 1,$<br>$s_{13}[3, 7, 8, 12, 13, 24, 28, 29] = 0$ |
| Column 2 | $s_2[3, 4, 5, 7, 8, 11, 12, 13, 15, 16, 19, \ldots, 23, 26, 28, 29] = 0,\ s_2[14] = 1,$<br>$s_6[0, 4, 8, 12, 16, 20, 21, 23, 28, 29] = 0,\ s_6[5, 13, 14, 26] = 1,$<br>$s_{10}[0, 5, 7, 8, 12, 13, 21, 29, 31] = 0,\ s_{10}[4, 15] = 1,$<br>$s_{14}[5, 8, 12, 16, 21, 23] = 0,\ s_{14}[13, 29] = 1$ |
| Column 3 | $s_3[0, 1, 5, 7, 8, 13, 14, 16, 19, 20, 21, 25, 28, 29] = 0,\ s_3[4] = 1,$<br>$s_7[0, 1, 8, 14, 20, 21, 29] = 0,\ s_7[4, 5, 16, 25] = 1,$<br>$s_{11}[0, 5, 24, 25, 31] = 0,$<br>$s_{15}[1, 8, 29] = 0$ |

# The Differential-Linear Distinguisher for NORX32 Permutation

- The distinguishing attack procedure
  1. Query $2^{63.7}$ pairs of 512-bit message with difference $\Delta_{in}$ and initial conditions in Table 5.
  2. For each pair of output from the oracle, compute the XORed sum of bits in $\Gamma_{out}$.
  3. Count the number $X$ that is the number of pairs such that the XORed sum is 0.
  4. If $X < 2^{62.7} - 2^{31.7}$, the oracle is the NORX32 permutation. Otherwise, the oracle is a random permutation.
- Complexity:
  - Time: $2^{64.7}$
  - Memory: Negligible
- Probability of success: 96%

# Conclusion

- The 4-round NORX permutations used in both NORX64 and NORX32 are not ideal.
  - NORX64 permutation can be distinguished with $2^{48.5}$ queries which has been experimentally verified.
  - NORX32 permutation can be distinguished with $2^{64.7}$ queries, which may be considered as semi-practical.
- The distinguishing attacks on the permutations do not directly lead to an attack on NORX authenticated encryption algorithm.
  - Restrictions on the positions where difference can be introduced
  - Output are not fully known
- The complexity of the attacks may be further improved by controlling the initial difference for more rounds.

*Thank you for your attention!*

# References

Jean-Philippe Aumasson, Philipp Jovanovic, and Samuel Neves.
Analysis of NORX: investigating differential and rotational properties.
In Diego F. Aranha and Alfred Menezes, editors, *LATINCRYPT 2014*, volume 8895 of *LNCS*, pages 306–324. Springer, 2015.

Jean-Philippe Aumasson, Philipp Jovanovic, and Samuel Neves.
NORX V3.0, 2016.
http://competitions.cr.yp.to/round3/norxv30.pdf.

Alex Biryukov, Aleksei Udovenko, and Vesselin Velichkov.
Analysis of the NORX core permutation.
*IACR Cryptology ePrint Archive*, 2017:34, 2017.

Colin Chaigneau, Thomas Fuhr, Henri Gilbert, Jérémy Jean, and Jean-René Reinhard.
Cryptanalysis of NORX v2.0.
*IACR Trans. Symmetric Cryptol.*, 2017(1):156–174, 2017.

Sourav Das, Subhamoy Maitra, and Willi Meier.
Higher order differential analysis of NORX.
*IACR Cryptology ePrint Archive*, 2015:186, 2015.