On Xoodoo

Gilles Van Assche[1]

based on joint work with
Joan Daemen[2], Seth Hoffert and Ronny Van Keer[1]

[1]STMicroelectronics
[2]Radboud University

Advances in Permutation-Based Cryptography
Milano, Italy, October 2018

Outline

Outline

What is Xoodoo?

Xoodoo · *[noun, mythical]* · /zu: du:/ · Alpine mammal that lives in compact herds, can survive avalanches and is appreciated for the wide trails it creates in the landscape. Despite its fluffy appearance it is very robust and does not get distracted by side channels.

Xoodoo

Xoodoo cookbook:
https://eprint.iacr.org/2018/767
[Keccak team with Seth Hoffert]

- 384-bit permutation *Keccak philosophy ported to Gimli shape*
- Main purpose: usage in Farfalle: Xoofff
  - Achouffe configuration
  - Efficient on wide range of platforms

Xoodoo

Xoodoo cookbook:
https://eprint.iacr.org/2018/767
[Keccak team with Seth Hoffert]

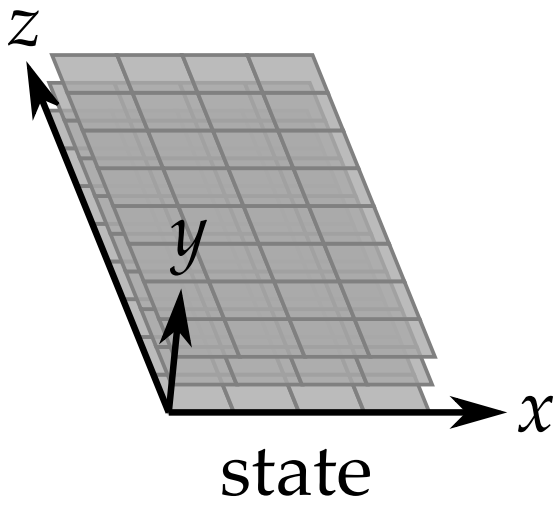
- 384-bit permutation *Keccak philosophy ported to Gimli shape*
- Main purpose: usage in Farfalle: Xoofff
  - Achouffe configuration
  - Efficient on wide range of platforms

Xoodoo

Xoodoo cookbook:
https://eprint.iacr.org/2018/767
[Keccak team with Seth Hoffert]

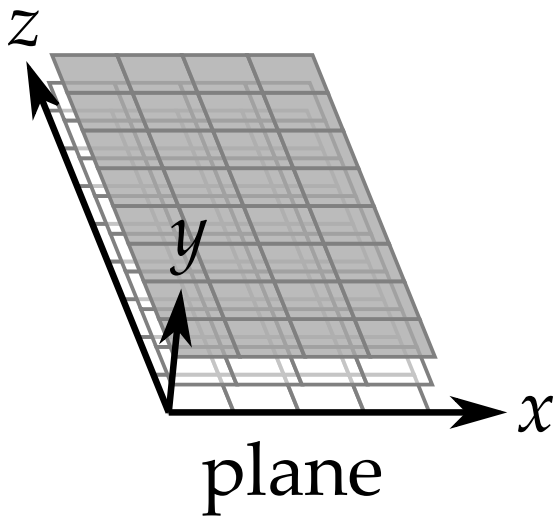
- 384-bit permutation *Keccak philosophy ported to Gimli shape*
- Main purpose: usage in Farfalle: Xoofff
  - Achouffe configuration
  - Efficient on wide range of platforms

Xoodoo state



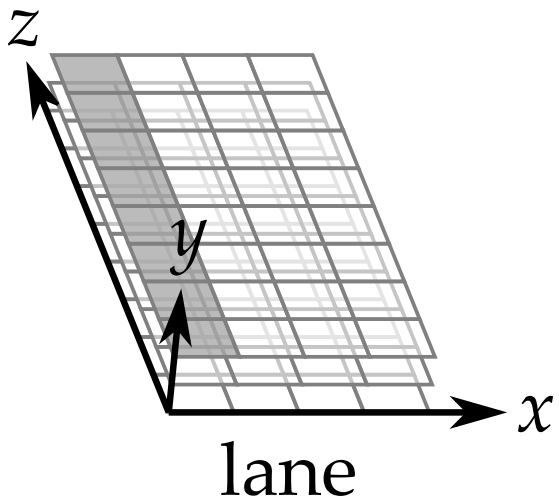
- State: 3 horizontal planes each consisting of 4 lanes

Xoodoo state



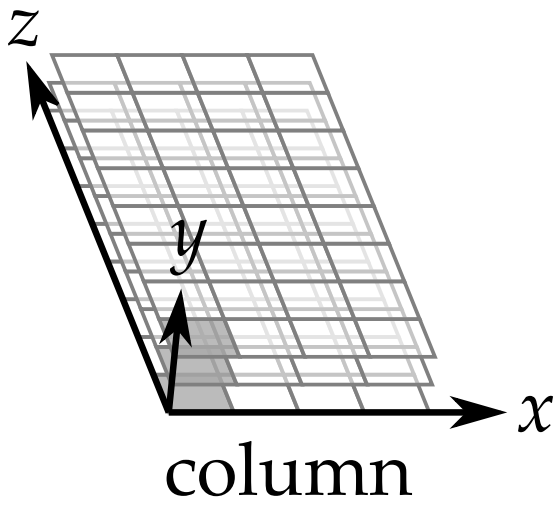
- State: 3 horizontal planes each consisting of 4 lanes

Xoodoo state



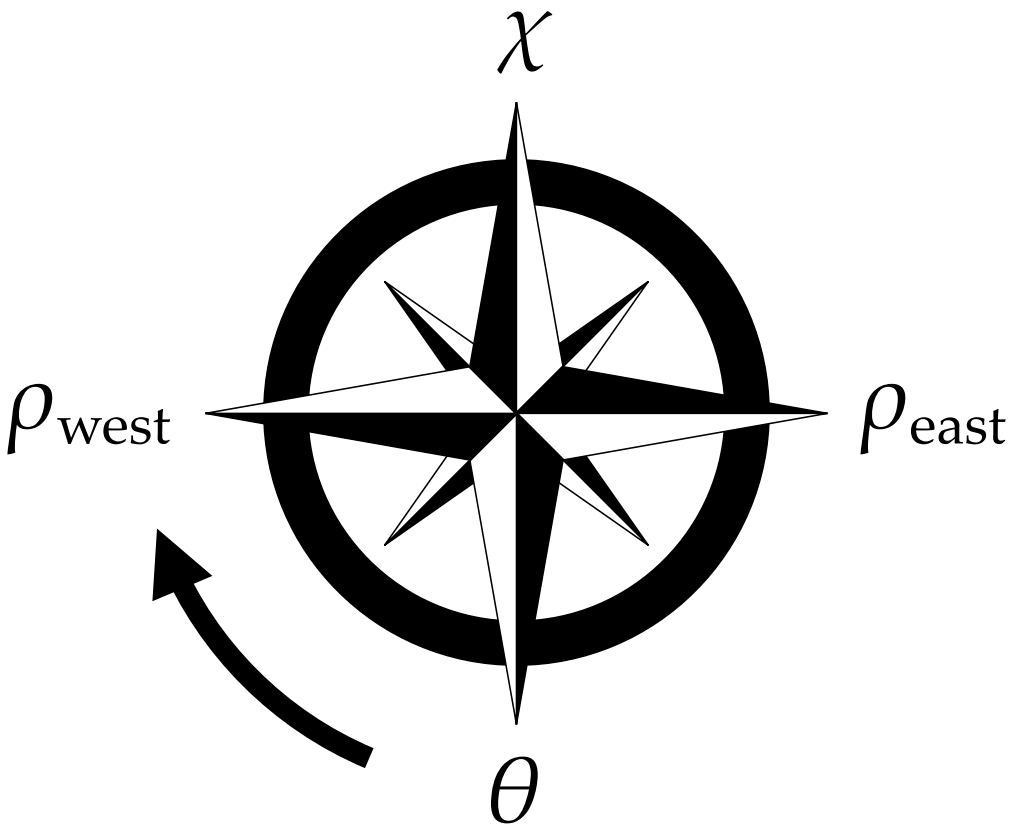
- State: 3 horizontal planes each consisting of 4 lanes

Xoodoo state



- State: 3 horizontal planes each consisting of 4 lanes

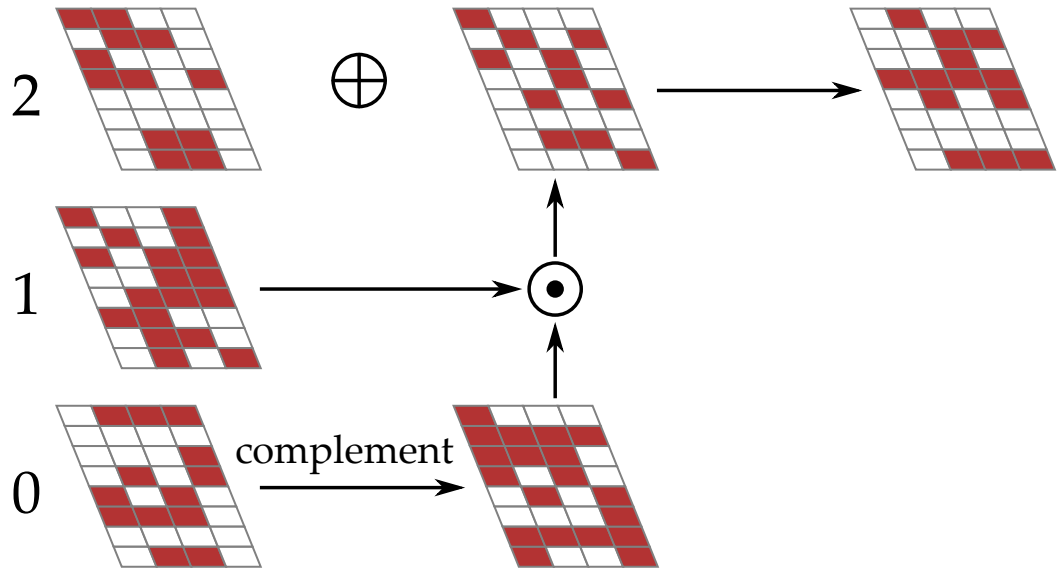
Xoodoo round function



Iterated: n_r rounds that differ only by round constant

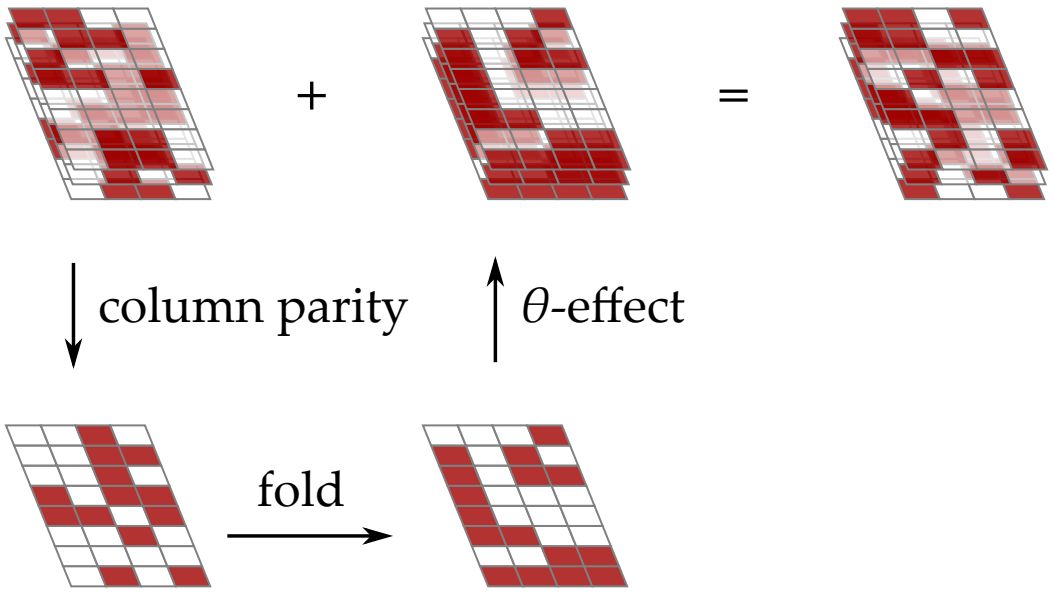
Nonlinear mapping χ

Effect on one plane:



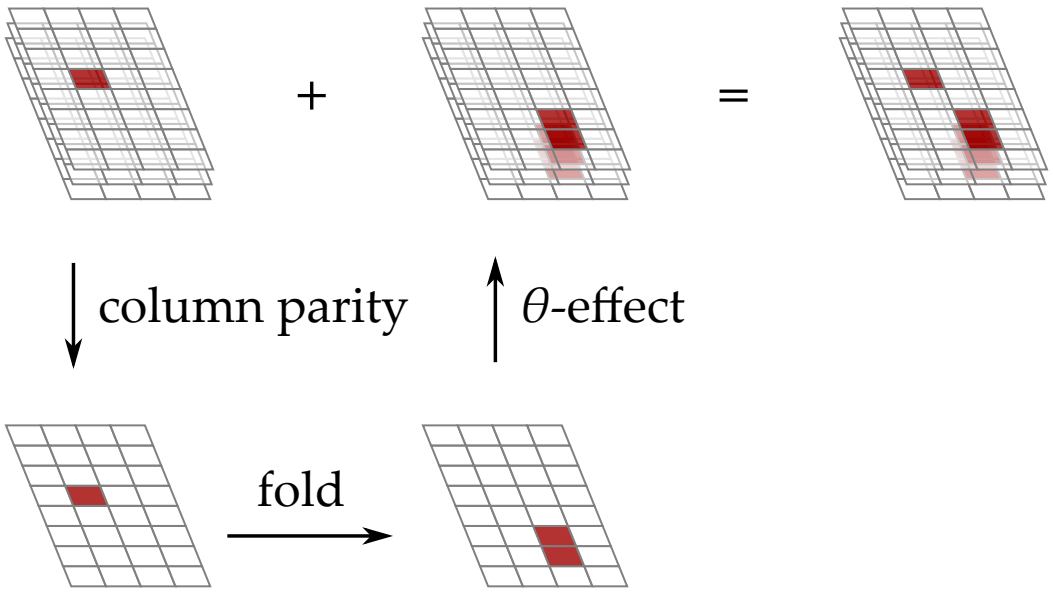
- χ as in Keccak-p, operating on 3-bit columns
- Involution and same propagation differentially and linearly

Mixing layer θ



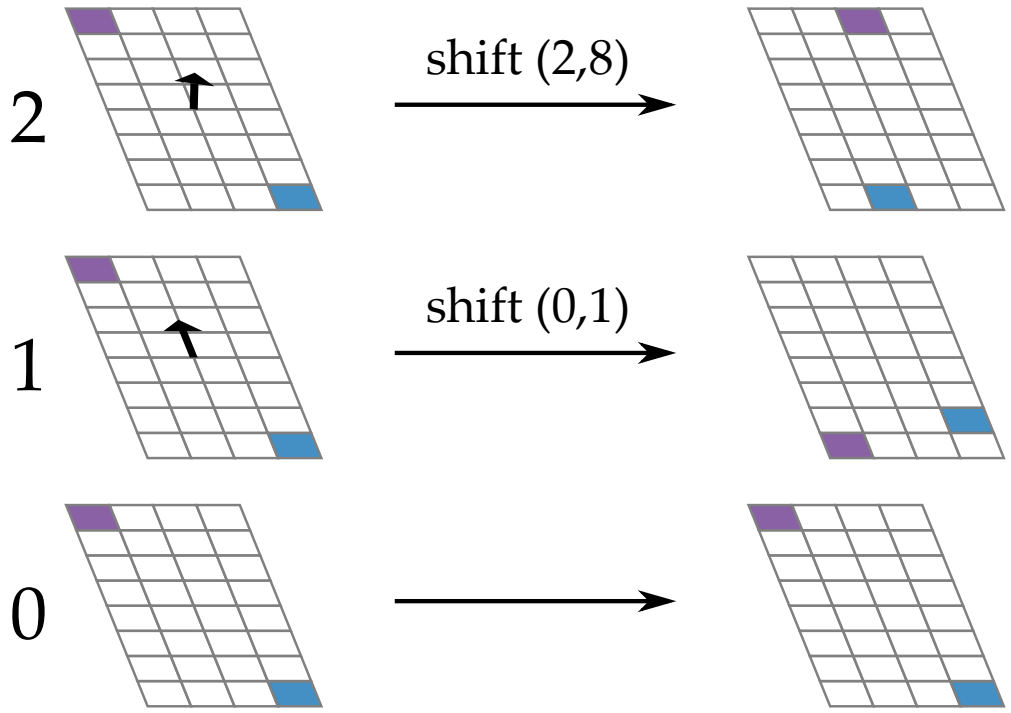
- Column parity mixer: compute parity, fold and add to state
- Good average diffusion, identity for states in *kernel*

Mixing layer θ



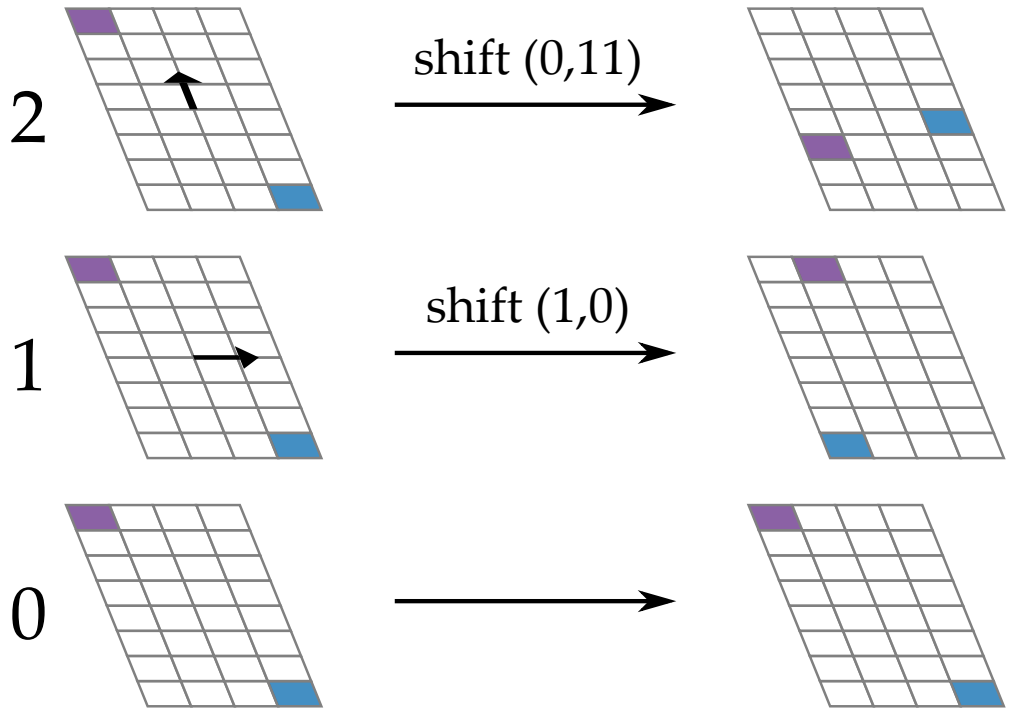
- Column parity mixer: compute parity, fold and add to state
- Good average diffusion, identity for states in *kernel*

Plane shift ρ_{east}



- After χ and before θ
- Shifts planes $y = 1$ and $y = 2$ over different directions

Plane shift ρ_{west}



- After θ and before χ
- Shifts planes $y = 1$ and $y = 2$ over different directions

Xoodoo pseudocode

n_r rounds from $i = 1 - n_r$ to 0, with a 5-step round function:

θ :

$$P \leftarrow A_0 + A_1 + A_2$$
$$E \leftarrow P \lll (1, 5) + P \lll (1, 14)$$
$$A_y \leftarrow A_y + E \text{ for } y \in \{0, 1, 2\}$$

ρ_{west} :

$$A_1 \leftarrow A_1 \lll (1, 0)$$
$$A_2 \leftarrow A_2 \lll (0, 11)$$

ι :

$$A_{0,0} \leftarrow A_{0,0} + C_i$$

χ :

$$B_0 \leftarrow \overline{A_1} \cdot A_2$$
$$B_1 \leftarrow \overline{A_2} \cdot A_0$$
$$B_2 \leftarrow \overline{A_0} \cdot A_1$$
$$A_y \leftarrow A_y + B_y \text{ for } y \in \{0, 1, 2\}$$

ρ_{east} :

$$A_1 \leftarrow A_1 \lll (0, 1)$$
$$A_2 \leftarrow A_2 \lll (2, 8)$$

Xoodoo software performance

	width	cycles/byte per round	
	bytes	ARM Cortex M3	Intel Skylake
Keccak-$p[1600, n_r]$	200	2.44	0.080
ChaCha	64	0.69	0.059
Gimli	48	0.91	0.074*
Xoodoo	48	1.10	0.083

$*$ on Intel Haswell

- Xoodoo has slower rounds than Gimli but ...
- ... requires less rounds for equal security objectives!

Xoodoo software performance

	width	cycles/byte per round	
	bytes	ARM Cortex M3	Intel Skylake
Keccak-$p[1600, n_r]$	200	2.44	0.080
ChaCha	64	0.69	0.059
Gimli	48	0.91	0.074*
Xoodoo	48	1.10	0.083

$*$ on Intel Haswell

- Xoodoo has slower rounds than Gimli but ...
- ... requires less rounds for equal security objectives!

Xoodoo software performance

	width	cycles/byte per round	
	bytes	ARM Cortex M3	Intel Skylake
Keccak-$p[1600, n_r]$	200	2.44	0.080
ChaCha	64	0.69	0.059
Gimli	48	0.91	0.074*
Xoodoo	48	1.10	0.083

$*$ on Intel Haswell

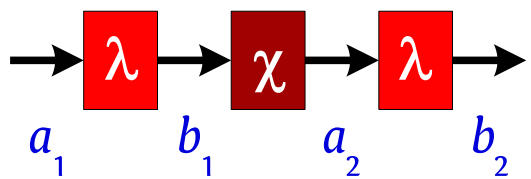
- Xoodoo has slower rounds than Gimli but ...
- ... requires less rounds for equal security objectives!

Outline

Trail bounds in Xoodoo

# rounds:	1	2	3	4	5	6
differential:	2	8	36	≥ 54	≥ 56	≥ 104
linear:	2	8	36	≥ 54	≥ 56	≥ 104

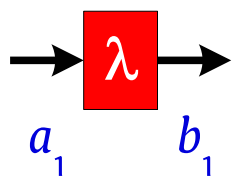
$w(Q) = w_{rev}(a_1) + w(b_1) + w(b_2)$



Trail bounds in Xoodoo

# rounds:	1	2	3	4	5	6
differential:	2	8	36	≥ 54	≥ 56	≥ 104
linear:	2	8	36	≥ 54	≥ 56	≥ 104

$w(Q) = w_{rev}(a_1) + w(b_1)$

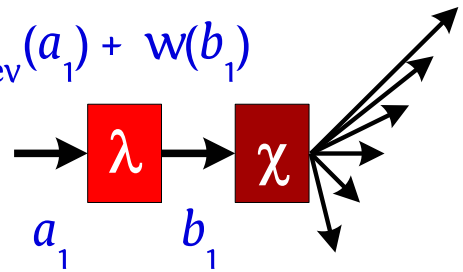


- Generating (a_1, b_1)

Trail bounds in Xoodoo

# rounds:	1	2	3	4	5	6
differential:	2	8	36	≥ 54	≥ 56	≥ 104
linear:	2	8	36	≥ 54	≥ 56	≥ 104

$w(Q) = w_{rev}(a_1) + w(b_1)$

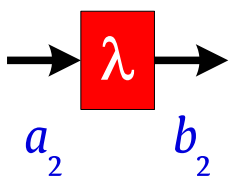


- Generating (a_1, b_1)
- Extending forward by one round till weight 50

Trail bounds in Xoodoo

# rounds:	1	2	3	4	5	6
differential:	2	8	36	≥ 54	≥ 56	≥ 104
linear:	2	8	36	≥ 54	≥ 56	≥ 104

$w(Q) = w_{rev}(a_2) + w(b_2)$

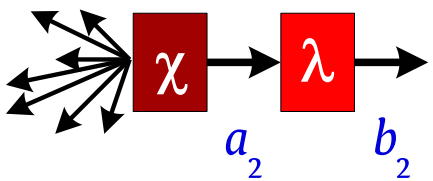


- Generating (a_2, b_2)

Trail bounds in Xoodoo

# rounds:	1	2	3	4	5	6
differential:	2	8	36	≥ 54	≥ 56	≥ 104
linear:	2	8	36	≥ 54	≥ 56	≥ 104

$w(Q) = w_{rev}(a_2) + w(b_2)$

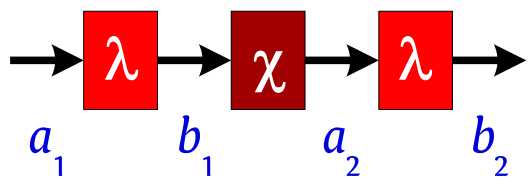


- Generating (a_2, b_2)
- Extending backward by one round till weight 50

Trail bounds in Xoodoo

# rounds:	1	2	3	4	5	6
differential:	2	8	36	≥ 54	≥ 56	≥ 104
linear:	2	8	36	≥ 54	≥ 56	≥ 104

$w(Q) = w_{rev}(a_1) + w(b_1) + w(b_2)$



- Extending all 3-round trail cores to 6 rounds till weight 102

Using the tree-search approach

Set U of *units* with a total order relation $\prec$

Tree

- Node: subset of U, represented as a *unit list*

$$a = (u_i)_{i=1,\ldots,n} \quad u_1 \prec u_2 \prec \cdots \prec u_n$$

- Children of a node a:

$$a \cup \{u_{n+1}\} \quad \forall\, u_{n+1} \,:\, u_n \prec u_{n+1}$$

- Root: the empty set $a = \emptyset$

[Mella, Daemen, Van Assche, FSE 2017]

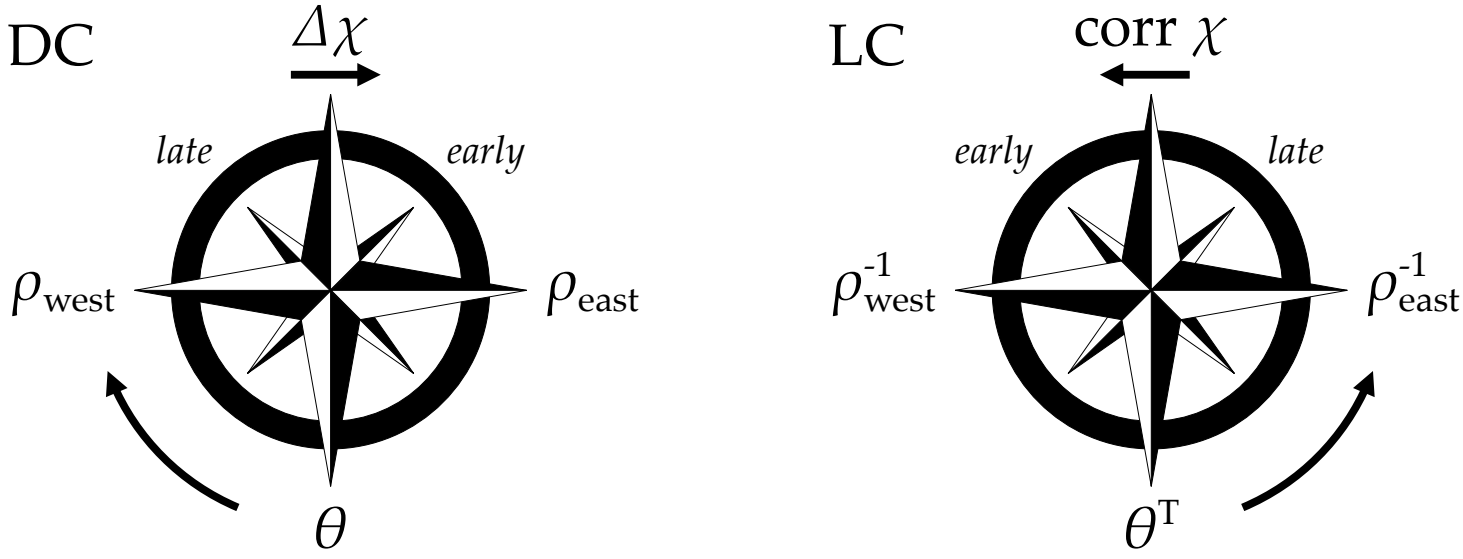
Definition of units

Units represent one bit at a time:

- Active bit in odd column (x, y, z)
- Bit in affected column $(x, y, z, \text{value } 0/1)$
- Active bit of an orbital (x, y, z)

$\Rightarrow$ allows for finer-grained bounding

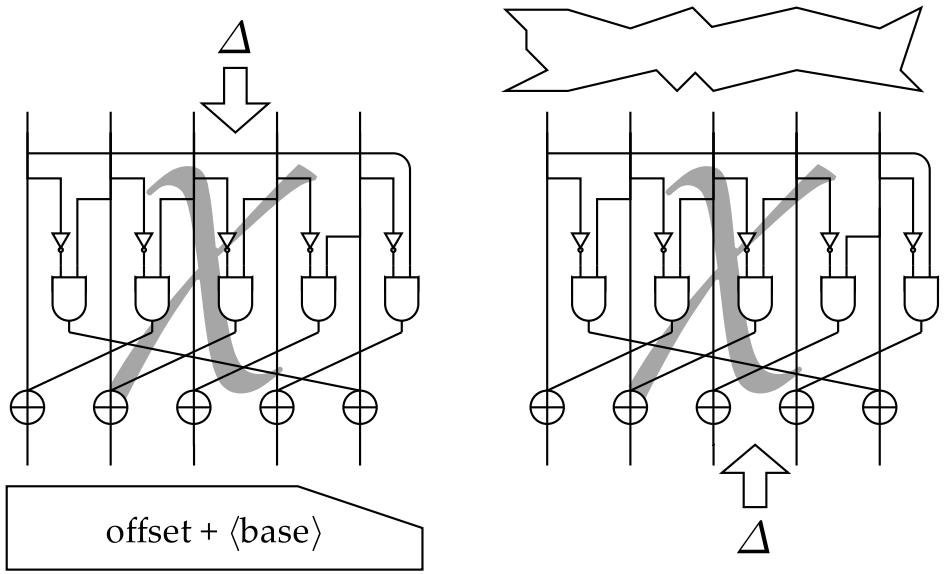
Properties of the trail search



Difference and mask propagation in χ follow the same rule
$\Rightarrow$ differential and linear trail search are almost identical

Properties of the trail search

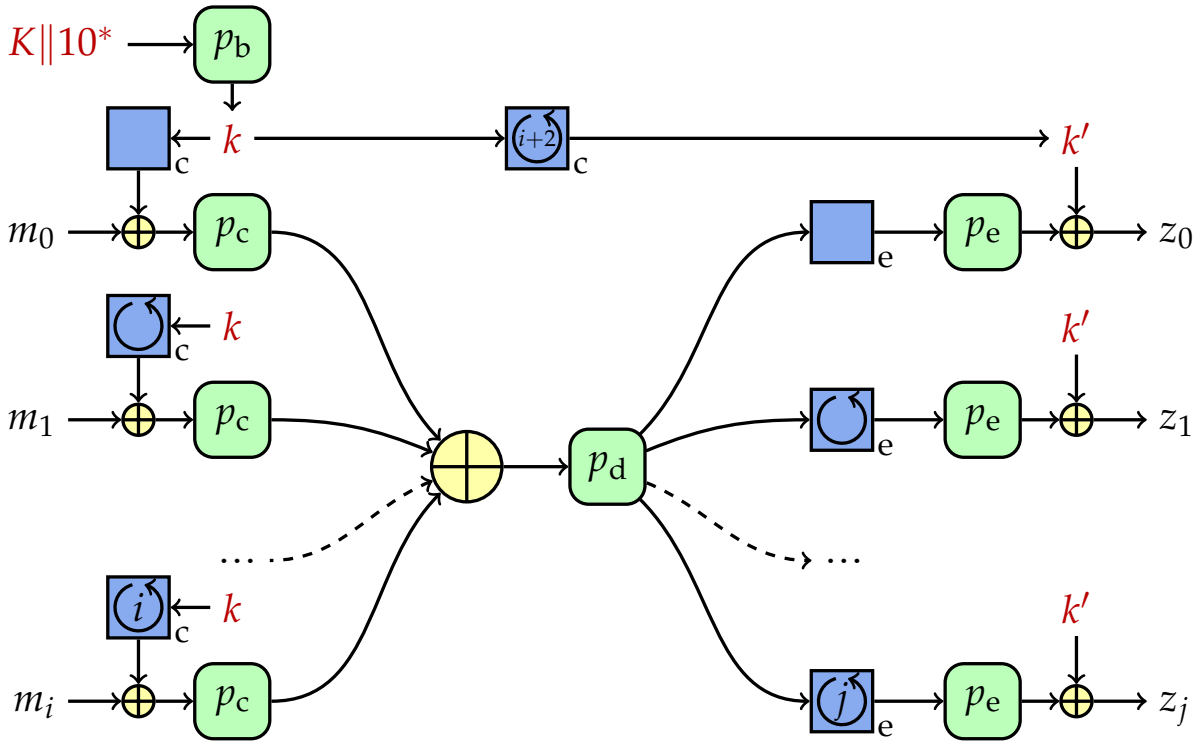
Compared to trail search in KECCAK-p:



In XOODOO, both χ and χ^{-1} have algebraic degree 2
$\Rightarrow$ affine-space extension in both directions

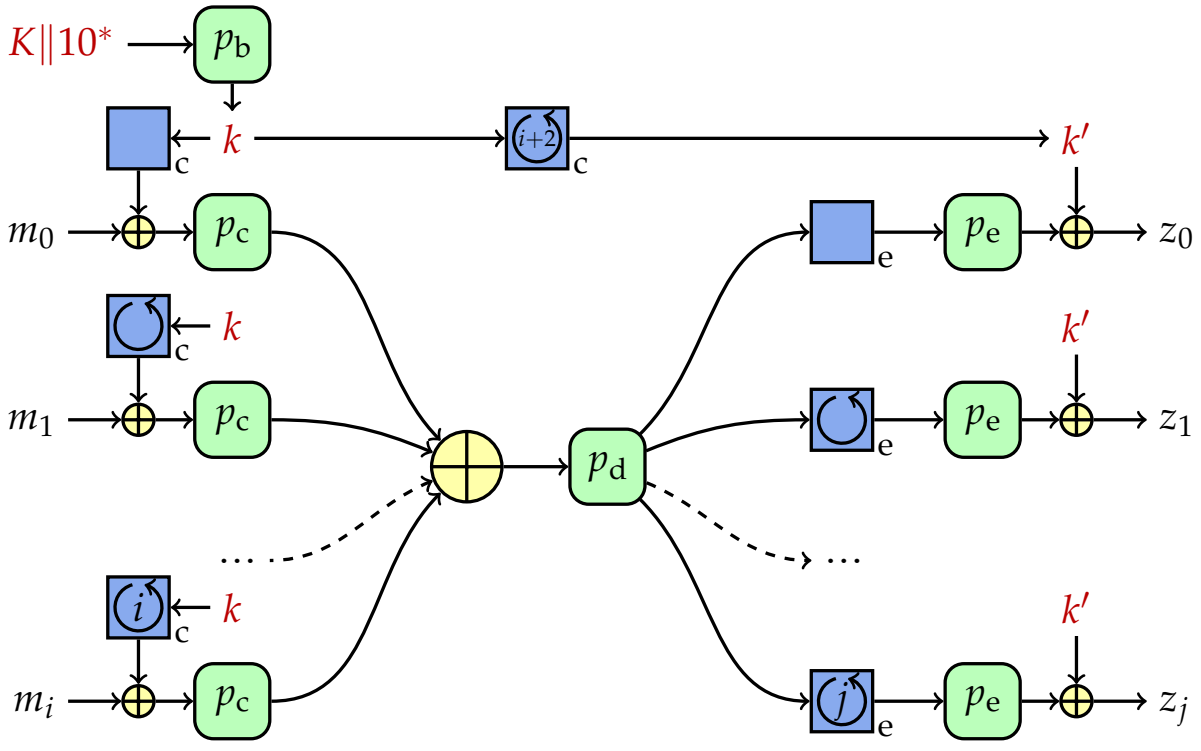
Outline

XOOFFF = Farfalle + XOODOO



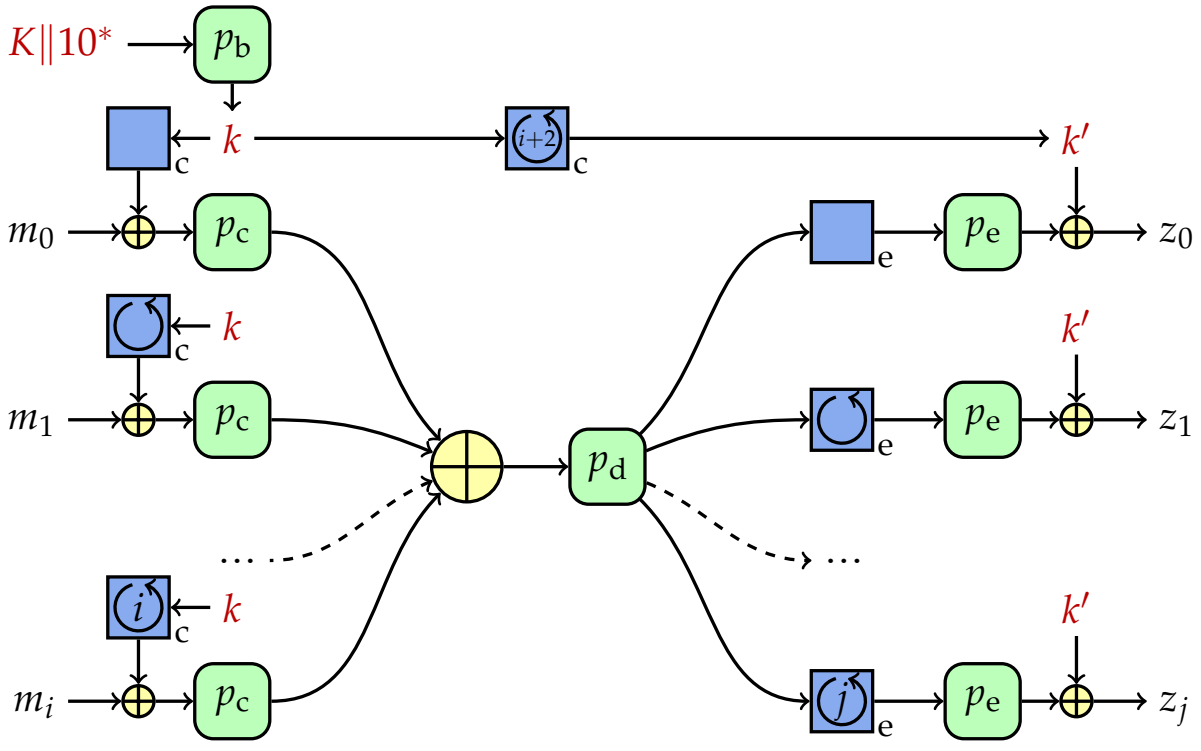
- $p_b = p_c = p_d = p_e =$ XOODOO[6]
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: 128 bits, incl. multi-target and quantum adv.

XOOFFF = Farfalle + XOODOO



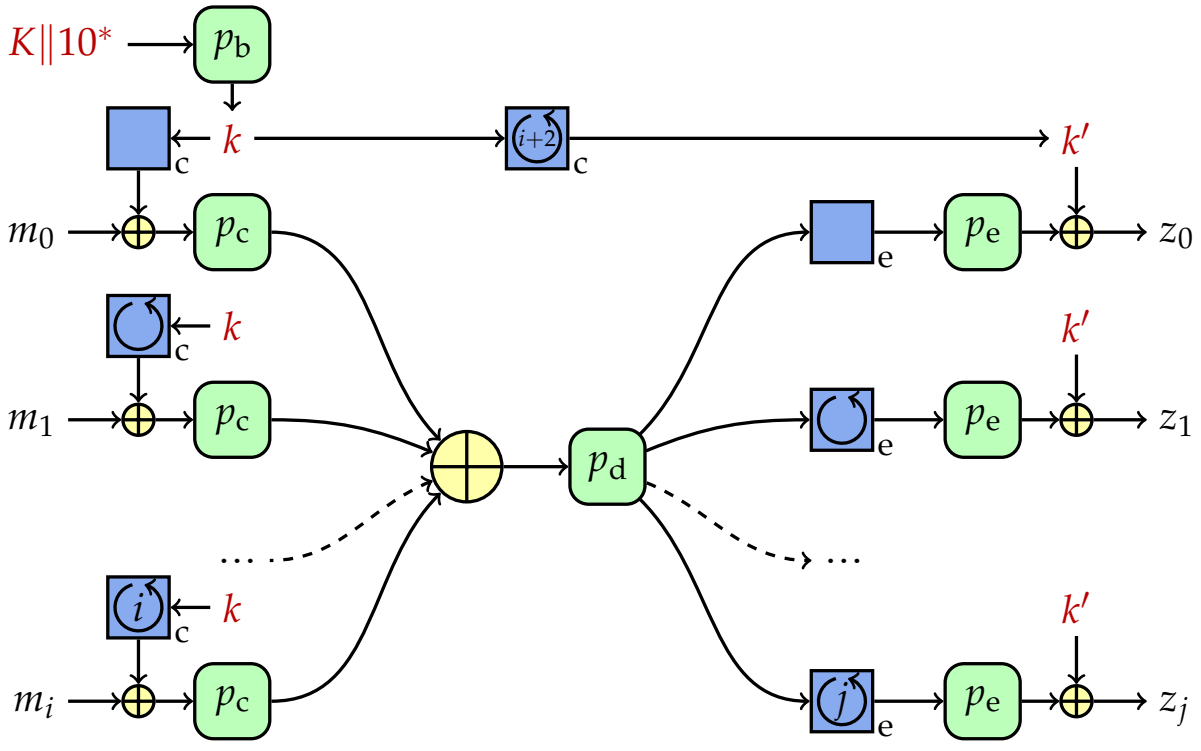
- $p_b = p_c = p_d = p_e =$ XOODOO[6]
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: 128 bits, incl. multi-target and quantum adv.

XOOFFF = Farfalle + XOODOO



- $p_b = p_c = p_d = p_e = \text{XOODOO}[6]$
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: 128 bits, incl. multi-target and quantum adv.

XOOFFF = Farfalle + XOODOO



- $p_b = p_c = p_d = p_e = \text{XOODOO}[6]$
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: 128 bits, incl. multi-target and quantum adv.

Xoofff applications and implementations

The Xoodoo Cookbook also specifies:

- Xoofff-SANE: session AE relying on user nonce
- Xoofff-SANSE: session AE using SIV technique
- Xoofff-WBC: tweakable wide block cipher

Keccak Code Package

⇓

eXtended Keccak Code Package

Xoofff applications and implementations

The Xoodoo Cookbook also specifies:

- Xoofff-SANE: session AE relying on user nonce
- Xoofff-SANSE: session AE using SIV technique
- Xoofff-WBC: tweakable wide block cipher

Keccak Code Package

⇓

eXtended Keccak Code Package

Any questions?

Thanks for your attention!

- More information
  `https://eprint.iacr.org/2018/767`
- Some implementations
  `https://github.com/XoodooTeam/Xoodoo/` (ref. code in C++ and Python)
  `https://github.com/XKCP/XKCP` (C, Assembler)
  `https://tinycrypt.wordpress.com/2018/02/06/...` (C, Assembler)