The design of XOODOO and XOOFFF

Joan DAEMEN[2] Seth HOFFERT
Gilles VAN ASSCHE[1] Ronny VAN KEER[1]

[1]STMicroelectronics
[2]Radboud University

Fast Software Encryption
Paris, France, March 25, 2019

Outline

Outline

Motivation

Fast software encryption

Motivation

Fast **and secure** software encryption

Motivation

Fast **and secure** software authenticated encryption

Motivation

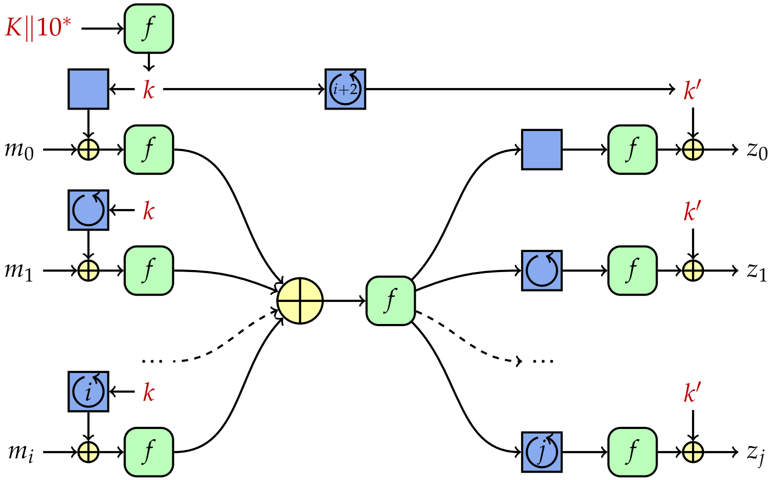
Fast **and secure** software and hardware authenticated encryption

Motivation

Fast **and secure** software and hardware authenticated encryption

Performance on a wide range of platforms

One year ago: Farfalle and parallelism



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

- 384-bit permutation: Gimli shape [Bernstein et al., CHES 2017]
  - ... but *Keccak's design philosophy*
- Main purpose: Xoofff
  - Farfalle in Achouffe configuration
  - Efficient on wide range of platforms
- Other purpose: Xoodyak
  - Duplex construction
  - See [Xoodoo Cookbook, ePrint 2018/767]

Xoodoo

- 384-bit permutation: Gimli shape [Bernstein et al., CHES 2017]
  - ... but *Keccak's design philosophy*
- Main purpose: Xoofff
  - Farfalle in Achouffe configuration
  - Efficient on wide range of platforms
- Other purpose: Xoodyak
  - Duplex construction
  - See [Xoodoo Cookbook, ePrint 2018/767]

Xoodoo

- 384-bit permutation: Gimli shape [Bernstein et al., CHES 2017]
  - ... but *Keccak's design philosophy*
- Main purpose: Xoofff
  - Farfalle in Achouffe configuration
  - Efficient on wide range of platforms
- Other purpose: Xoodyak
  - Duplex construction
  - See [Xoodoo Cookbook, ePrint 2018/767]

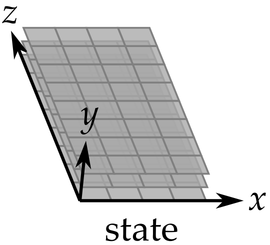
Xoodoo

- 384-bit permutation: Gimli shape [Bernstein et al., CHES 2017]
  - ... but *Keccak's design philosophy*
- Main purpose: Xoofff
  - Farfalle in Achouffe configuration
  - Efficient on wide range of platforms
- Other purpose: Xoodyak
  - Duplex construction
  - See [Xoodoo Cookbook, ePrint 2018/767]

Xoodoo

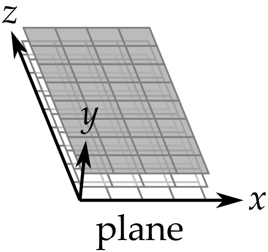
- 384-bit permutation: Gimli shape [Bernstein et al., CHES 2017]
  - ... but *Keccak's design philosophy*
- Main purpose: Xoofff
  - Farfalle in Achouffe configuration
  - Efficient on wide range of platforms
- Other purpose: Xoodyak
  - Duplex construction
  - See [Xoodoo Cookbook, ePrint 2018/767]

Xoodoo state



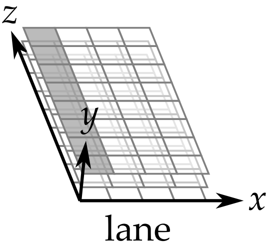
- State: 3 horizontal planes each consisting of 4 lanes

Xoodoo state



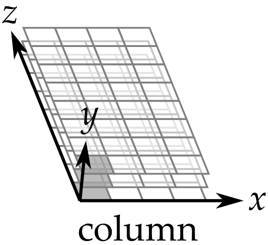
- State: 3 horizontal planes each consisting of 4 lanes

Xoodoo state



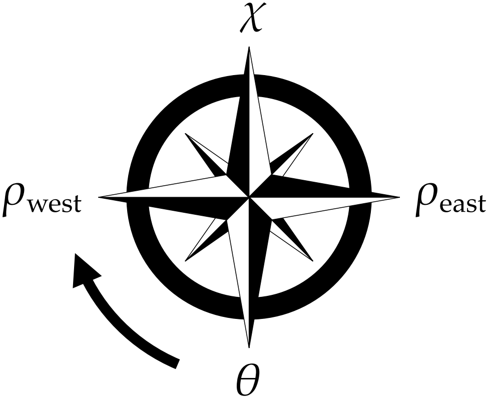
- State: 3 horizontal planes each consisting of 4 lanes

Xoodoo state



- State: 3 horizontal planes each consisting of 4 lanes

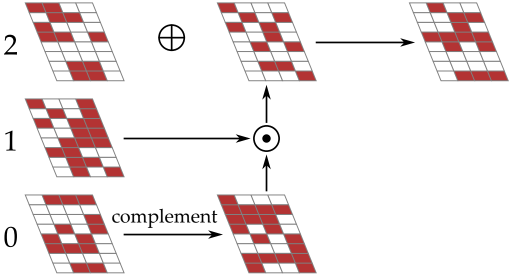
Xoodoo round function



Iterated: n_r rounds that differ only by round constant

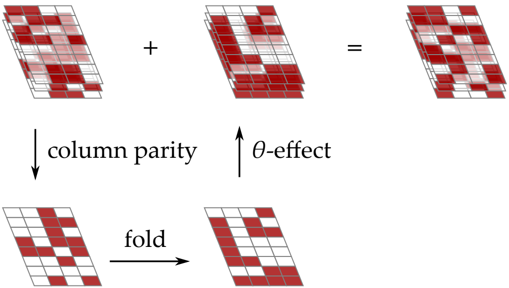
Nonlinear mapping χ

Effect on one plane:



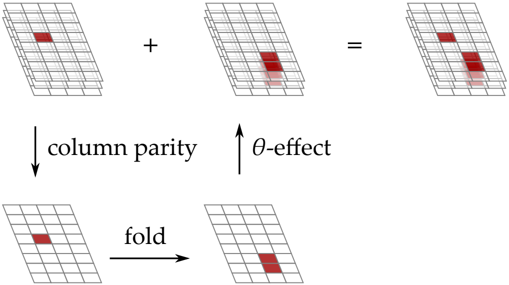
- χ as in Keccak-p, operating on 3-bit columns
- Involution and same propagation differentially and linearly

Mixing layer θ



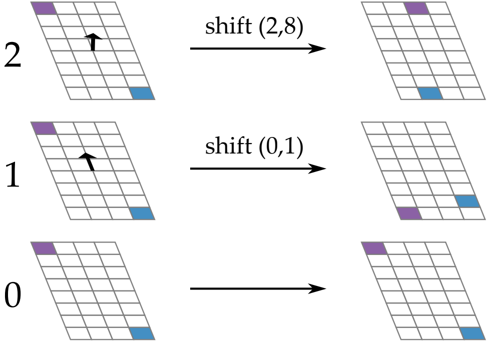
- Column parity mixer: compute parity, fold and add to state
- Good average diffusion, identity for states in *kernel*

Mixing layer θ



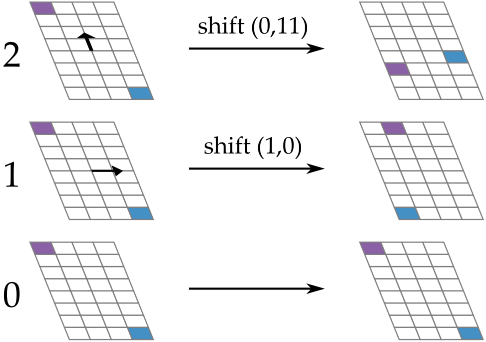
- Column parity mixer: compute parity, fold and add to state
- Good average diffusion, identity for states in *kernel*

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:

$$
\begin{aligned}
\theta: & \\
& P \leftarrow A_0 + A_1 + A_2 \\
& E \leftarrow P \lll (1,5) + P \lll (1,14) \\
& A_y \leftarrow A_y + E \text{ for } y \in \{0,1,2\} \\
\rho_{\text{west}}: & \\
& A_1 \leftarrow A_1 \lll (1,0) \\
& A_2 \leftarrow A_2 \lll (0,11) \\
\iota: & \\
& A_{0,0} \leftarrow A_{0,0} + C_i \\
\chi: & \\
& B_0 \leftarrow \overline{A_1} \cdot A_2 \\
& B_1 \leftarrow \overline{A_2} \cdot A_0 \\
& B_2 \leftarrow \overline{A_0} \cdot A_1 \\
& A_y \leftarrow A_y + B_y \text{ for } y \in \{0,1,2\} \\
\rho_{\text{east}}: & \\
& A_1 \leftarrow A_1 \lll (0,1) \\
& A_2 \leftarrow A_2 \lll (2,8)
\end{aligned}
$$

Xoodoo propagation properties

- Security limited by $\max \mathrm{DP}(\Delta_a, \Delta_b)$
  - $\max \mathrm{DP}(\Delta_a, \Delta_b)$ by itself hard to determine
- For Xoodoo we believe $\max \mathrm{DP}(\Delta_a, \Delta_b) \approx \max_Q \mathrm{DP}(Q)$
  - Q a differential trail: $\Delta_0, \Delta_1, \Delta_2, \ldots \Delta_r$
  - Trail weight $\mathrm{w}(Q)$ defined by $2^{-\mathrm{w}(Q)} = \mathrm{DP}(Q)$

Bounds on trail weights, using [Mella, Daemen, Van Assche, ToSC 2016]:

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

Satisfying Strict Avalanche Criterion (SAC) takes:

- 3.5 rounds in forward direction
- 2 rounds in backward direction

Xoodoo propagation properties

- Security limited by $\max \mathrm{DP}(\Delta_a, \Delta_b)$
  - $\max \mathrm{DP}(\Delta_a, \Delta_b)$ by itself hard to determine
- For Xoodoo we believe $\max \mathrm{DP}(\Delta_a, \Delta_b) \approx \max_Q \mathrm{DP}(Q)$
  - Q a differential trail: $\Delta_0, \Delta_1, \Delta_2, \ldots \Delta_r$
  - Trail weight $\mathrm{w}(Q)$ defined by $2^{-\mathrm{w}(Q)} = \mathrm{DP}(Q)$

Bounds on trail weights, using [Mella, Daemen, Van Assche, ToSC 2016]:

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

Satisfying Strict Avalanche Criterion (SAC) takes:

- 3.5 rounds in forward direction
- 2 rounds in backward direction

Xoodoo propagation properties

- Security limited by $\max \mathrm{DP}(\Delta_a, \Delta_b)$
  - $\max \mathrm{DP}(\Delta_a, \Delta_b)$ by itself hard to determine
- For Xoodoo we believe $\max \mathrm{DP}(\Delta_a, \Delta_b) \approx \max_Q \mathrm{DP}(Q)$
  - Q a differential trail: $\Delta_0, \Delta_1, \Delta_2, \ldots \Delta_r$
  - Trail weight $\mathrm{w}(Q)$ defined by $2^{-\mathrm{w}(Q)} = \mathrm{DP}(Q)$

Bounds on trail weights, using [Mella, Daemen, Van Assche, ToSC 2016]:

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

Satisfying Strict Avalanche Criterion (SAC) takes:

- 3.5 rounds in forward direction
- 2 rounds in backward direction

Xoodoo propagation properties

- Security limited by $\max \mathrm{DP}(\Delta_a, \Delta_b)$
  - $\max \mathrm{DP}(\Delta_a, \Delta_b)$ by itself hard to determine
- For Xoodoo we believe $\max \mathrm{DP}(\Delta_a, \Delta_b) \approx \max_Q \mathrm{DP}(Q)$
  - Q a differential trail: $\Delta_0, \Delta_1, \Delta_2, \ldots \Delta_r$
  - Trail weight $\mathrm{w}(Q)$ defined by $2^{-\mathrm{w}(Q)} = \mathrm{DP}(Q)$

Bounds on trail weights, using [Mella, Daemen, Van Assche, ToSC 2016]:

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

Satisfying Strict Avalanche Criterion (SAC) takes:

- 3.5 rounds in forward direction
- 2 rounds in backward direction

Xoodoo propagation properties

- Security limited by $\max \mathrm{DP}(\Delta_a, \Delta_b)$
  - $\max \mathrm{DP}(\Delta_a, \Delta_b)$ by itself hard to determine
- For Xoodoo we believe $\max \mathrm{DP}(\Delta_a, \Delta_b) \approx \max_Q \mathrm{DP}(Q)$
  - Q a differential trail: $\Delta_0, \Delta_1, \Delta_2, \ldots \Delta_r$
  - Trail weight $\mathrm{w}(Q)$ defined by $2^{-\mathrm{w}(Q)} = \mathrm{DP}(Q)$

Bounds on trail weights, using [Mella, Daemen, Van Assche, ToSC 2016]:

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

Satisfying Strict Avalanche Criterion (SAC) takes:

- 3.5 rounds in forward direction
- 2 rounds in backward direction

Xoodoo propagation properties

- Security limited by $\max \mathrm{DP}(\Delta_a, \Delta_b)$
  - $\max \mathrm{DP}(\Delta_a, \Delta_b)$ by itself hard to determine
- For Xoodoo we believe $\max \mathrm{DP}(\Delta_a, \Delta_b) \approx \max_Q \mathrm{DP}(Q)$
  - Q a differential trail: $\Delta_0, \Delta_1, \Delta_2, \ldots \Delta_r$
  - Trail weight $w(Q)$ defined by $2^{-w(Q)} = \mathrm{DP}(Q)$

Bounds on trail weights, using [Mella, Daemen, Van Assche, ToSC 2016]:

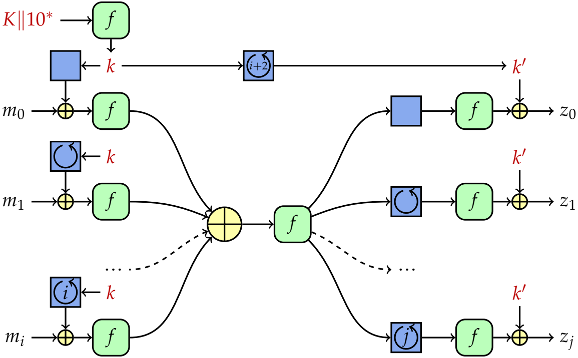
# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

Satisfying Strict Avalanche Criterion (SAC) takes:

- 3.5 rounds in forward direction
- 2 rounds in backward direction

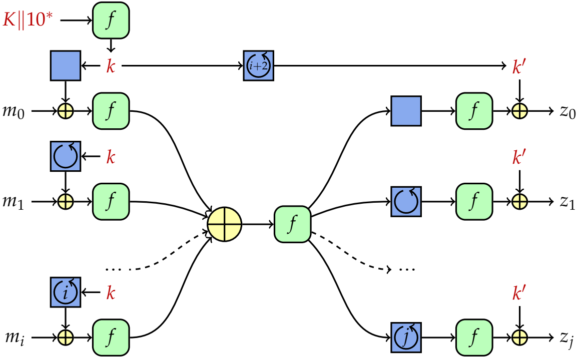
Outline

Xoofff = Farfalle + Xoodoo



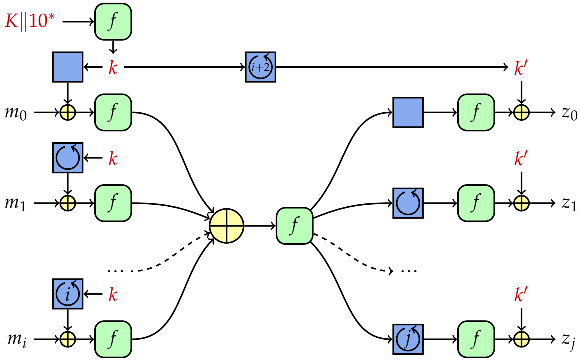
- $f = \text{Xoodoo}[6]$
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: $\geq$ 128 bits (96 bits post-quantum)

Xoofff = Farfalle + Xoodoo



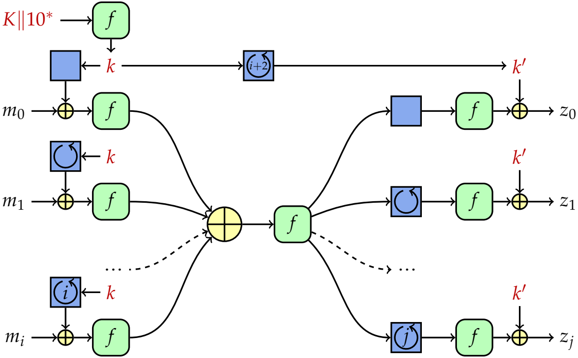
- $f = \text{Xoodoo}[6]$
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: $\geq$ 128 bits (96 bits post-quantum)

XOOFFF = Farfalle + XOODOO



- $f = $ XOODOO[6]
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: $\geq$ 128 bits (96 bits post-quantum)

Xoofff = Farfalle + Xoodoo



- $f = $ Xoodoo[6]
- Input mask rolling with LFSR, state rolling with NLFSR
- Target security: $\geq$ 128 bits (96 bits post-quantum)

Xoofff performance

Xoofff		
mask derivation	1985	cycles
less than 48 bytes	5658	cycles
MAC computation use case:		
long inputs	26.0	cycles/byte
Stream encryption use case:		
long outputs	25.1	cycles/byte
AES-128 counter mode	121.4	cycles/byte

ARM® Cortex-M0

Xoofff performance

Xoofff		
mask derivation	781	cycles
less than 48 bytes	2568	cycles
MAC computation use case:		
long inputs	8.8	cycles/byte
Stream encryption use case:		
long outputs	8.1	cycles/byte
AES-128 counter mode	33.2	cycles/byte

ARM® Cortex-M3

Xoofff performance

Xoofff		
mask derivation	168	cycles
less than 48 bytes	504	cycles
MAC computation use case:		
long inputs	0.90	cycles/byte
Stream encryption use case:		
long outputs	0.94	cycles/byte
AES-128 counter mode	0.65	cycles/byte

Intel® Core™ i5-6500 (Skylake), single core, Turbo Boost disabled (256-bit SIMD)

Xoofff performance

Xoofff		
mask derivation	74	cycles
less than 48 bytes	358	cycles
MAC computation use case:		
long inputs	0.40	cycles/byte
Stream encryption use case:		
long outputs	0.51	cycles/byte
AES-128 counter mode	0.65	cycles/byte

Intel® Core™ i7-7800X (SkylakeX), single core, Turbo Boost disabled
(512-bit SIMD)

Outline

Definition of a deck function

A deck function F_K

$$Z = 0^n + F_K\left(X^{(m)} \circ \cdots \circ X^{(1)}\right) \ll q$$

doubly extendable cryptographic keyed function

Definition of a deck function

A deck function F_K

$$Z = 0^n + F_K\left(X^{(m)} \circ \cdots \circ X^{(1)}\right) \ll q$$

- Input: sequence of strings $X^{(m)} \circ \cdots \circ X^{(1)}$

Definition of a deck function

A deck function F_K

$$Z = 0^n + F_K\left(X^{(m)} \circ \cdots \circ X^{(1)}\right) \ll q$$

- Input: sequence of strings $X^{(m)} \circ \cdots \circ X^{(1)}$
- Output: potentially infinite output
  - **pseudo-random function of the input**
  - taking n bits starting from offset q

Definition of a deck function

A deck function F_K

$$Z = 0^n + F_K\left(X^{(m)} \circ \cdots \circ X^{(1)}\right) \ll q$$

Efficient incrementality

- Extendable input
  1. Compute $F_K(X)$
  2. Compute $F_K(Y \circ X)$: cost independent of X

Definition of a deck function

A deck function F_K

$$Z = 0^n + F_K\left(X^{(m)} \circ \cdots \circ X^{(1)}\right) \ll q$$

Efficient incrementality

- Extendable input
  1. Compute $F_K(X)$
  2. Compute $F_K(Y \circ X)$: cost independent of X
- Extendable output
  1. Request n_1 bits from offset 0
  2. Request n_2 bits from offset n_1: cost independent of n_1

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$

$e \leftarrow 0^1$

history $\leftarrow N$

return optional setup tag $T = 0^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$

$C \leftarrow P + F_K(\text{history}) \ll t$

history $\leftarrow A||0||e \circ$ history

history $\leftarrow C||1||e \circ$ history

$T \leftarrow 0^t + F_K(\text{history})$

$e \leftarrow e + 1^1$

return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow 0^1$
history $\leftarrow N$
return optional setup tag $T = 0^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||0||e \circ$ history
history $\leftarrow C||1||e \circ$ history
$T \leftarrow 0^t + F_K(\text{history})$
$e \leftarrow e + 1^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ$ history
history $\leftarrow C||1||e \circ$ history
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + 1^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ$ history
history $\leftarrow C||\mathtt{1}||e \circ$ history
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + \mathtt{1}^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ$ history
history $\leftarrow C||\mathtt{1}||e \circ$ history
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + \mathtt{1}^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||0||e \circ \text{history}$
history $\leftarrow C||1||e \circ \text{history}$
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + 1^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ$ history
history $\leftarrow C||\mathtt{1}||e \circ$ history
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + \mathtt{1}^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ \text{history}$
history $\leftarrow C||\mathtt{1}||e \circ \text{history}$
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + \mathtt{1}^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ$ history
history $\leftarrow C||1||e \circ$ history
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + 1^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ \text{history}$
history $\leftarrow C||\mathtt{1}||e \circ \text{history}$
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + \mathtt{1}^1$
return ciphertext C and tag T

XOOFFF-SANE = Deck-SANE + XOOFFF

Deck-SANE: session-supporting and nonce-based

Initialization taking nonce $N \in \mathbb{Z}_2^*$
$e \leftarrow \mathtt{0}^1$
history $\leftarrow N$
return optional setup tag $T = \mathtt{0}^t + F_K(\text{history})$

Wrap taking metadata $A \in \mathbb{Z}_2^*$ and plaintext $P \in \mathbb{Z}_2^*$
$C \leftarrow P + F_K(\text{history}) \ll t$
history $\leftarrow A||\mathtt{0}||e \circ \text{history}$
history $\leftarrow C||\mathtt{1}||e \circ \text{history}$
$T \leftarrow \mathtt{0}^t + F_K(\text{history})$
$e \leftarrow e + \mathtt{1}^1$
return ciphertext C and tag T

Xoofff-SANE = Deck-SANE + Xoofff

Other applications

Using XOOFFF as a deck function:

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

[eXtended KECCAK Code Package]

Any questions?

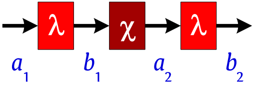
Thanks for your attention!

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

Trail bounds in XOODOO

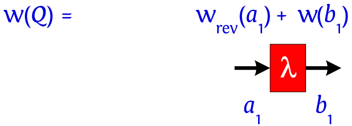
# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

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



Trail bounds in XOODOO

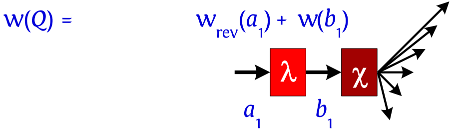
# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104



- Generating (a_1, b_1)

Trail bounds in XOODOO

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

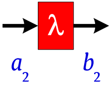


- Generating (a_1, b_1)
- Extending forward by one round till weight 50

Trail bounds in XOODOO

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

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

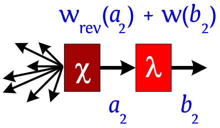


- Generating (a_2, b_2)

Trail bounds in Xoodoo

# rounds:	1	2	3	4	5	6
differential:	2	8	36	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104

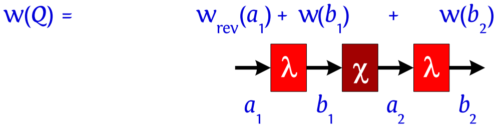
$w(Q) =$



- 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	$[74, 80]$	≥ 90	≥ 104
linear:	2	8	36	$[74, 80]$	≥ 90	≥ 104



- 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]

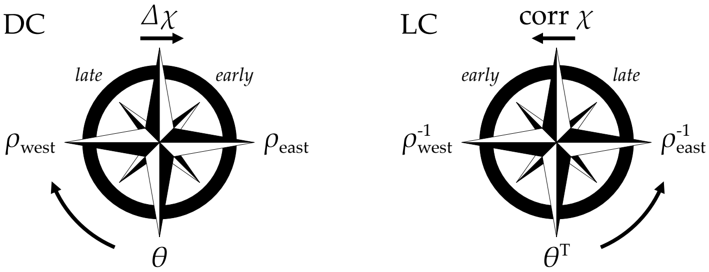
Definition of units

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

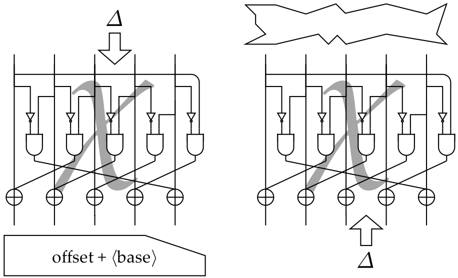
Properties of the trail search



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

Xoodoo software performance

	width	cycles/byte per round	
		ARM	
	bytes	Cortex M3	Cortex M0
Keccak-$p[1600, n_r]$	200	2.44	3.64
ChaCha	64	0.69	2.00
Gimli	48	0.91	2.04
Xoodoo	48	1.10	3.76