# First practical results on reduced-round KECCAK

# Unaligned rebound attack

**María Naya-Plasencia**
INRIA Paris-Rocquencourt, France

# Outline

- Introduction

- First Practical Results [NP-Röck-Meier11]
  - CP-Kernel: Differential paths.
  - (Near) collisions and distinguishers.
  - 2-rounds 2nd preimage.

- Unaligned rebound attack [Duc-Guo-Peyrin-Wei12]
  - Rebound attack.
  - Distinguisher on 8 rounds of Keccak-f.

# Introduction

# Security requirements of hash functions

- Collision resistance
  Finding two messages $\mathcal{M}$ and $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard".
- Second preimage resistance
  Given a message $\mathcal{M}$ and $\mathcal{H}(\mathcal{M})$, finding another message $\mathcal{M}'$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}(\mathcal{M}')$ must be "hard".
- Preimage resistance
  Given a hash $\mathcal{H}$, finding a message $\mathcal{M}$ so that $\mathcal{H}(\mathcal{M}) = \mathcal{H}$ must be "hard".

# Security requirements of hash functions?

A strict definition of "hard":

- Collision resistance
  - Generic attack needs $2^{\ell_h/2}$ hash function calls $\Rightarrow$ any attack requires at least as many hash function calls as the generic attack.
- Second preimage resistance and preimage resistance
  - Generic attack needs $2^{\ell_h}$ hash function calls $\Rightarrow$ any attack requires at least as many hash function calls as the generic attack.

# Security requirements of hash functions

- Collision, (Second) Preimage resistance...

Is that all we ask of a hash function? NO.

- Other types of attacks: near-collisions, multicollisions, length extension attacks, distinguishers...

# What Is a Distinguisher?

Good question...

In general, it is used for describing non-random properties:

- For example, finding an output or a family of outputs of the studied function with higher probability than for a random function.

# Analysis of Building Blocks

Attacks on the hash functions not always possible, but we still value information about the security margin of a hash function. We can analyse **reduced versions** AND/OR the **building blocks**.

- Proofs based on ideal properties of compression functions or internal permutations $\Rightarrow$ Study these components to check if the assumptions hold.

# Differential cryptanalysis [Biham, Shamir90]

- Differential path = configuration of differences in the internal state of the compression function through time.

- Each differential path has a probability of being verified.

# First Practical Results on Reduced-round Keccak

[NP-Röck-Meier, Indocrypt 2011]

# Previous Analysis on KECCAK

- On building blocks.
  - Zero sums up to 24 permutation rounds ⇒ Anne Canteaut's talk.
  - Lathrop, Aumasson and Khovratovich: triangulation and cube attack results on 4 rounds.

- Unmodified Reduced-round Hash Function Setting.
  - Bernstein: 2nd preimages on 6,7,8 rounds, complexities $2^{506}, 2^{507}, 2^{511.5}$ in time and $2^{176}, 2^{320}, 2^{508}$ in memory.

# First Practical Results

- 4-round hash function distinguisher.
- 3-round near-collision.
- 2-round collision.

- 2-round (second) preimages.

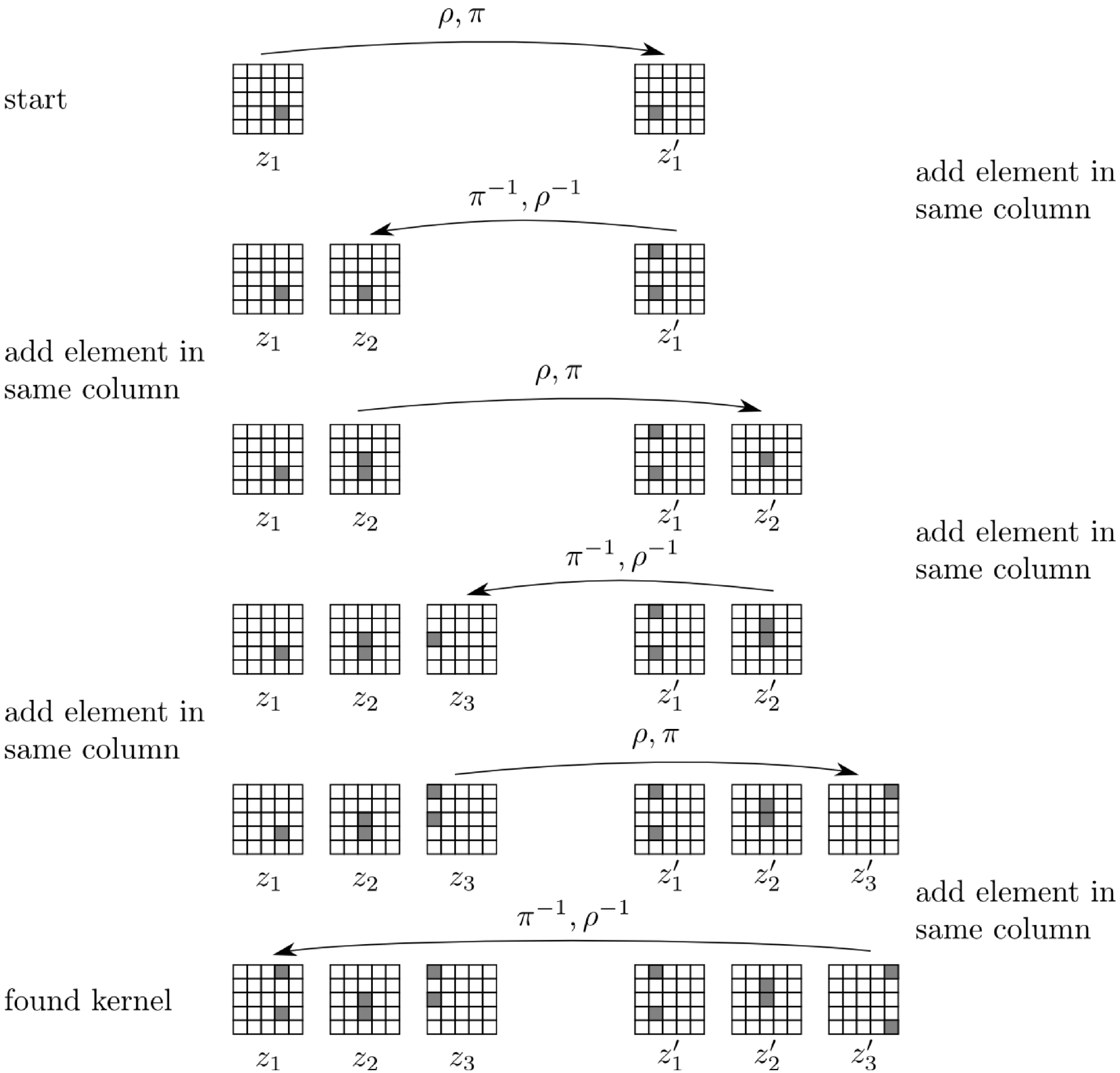
We have implemented all of them.

# Column Parity Kernel[Keccak team]

Transformation $\theta$ sums to each state-bit the parity of the weight of two columns $\rightarrow$ Property of $\theta$: when the weight of all the columns of a state is even, the transformation $\theta$ becomes the identity.

- For values and differences.
- Kernel: differences that are invariant through $\theta$.
- We searched Double Kernels: verified for two rounds.

# Building a double Kernel



# Building a double Kernel

- $\chi$: 1 difference stays the same with proba $2^{-2}$.
- Hash function setting: initial difference on message.
- Low weight differential paths for 3 rounds (6-6-6).

$$\Delta_1 \Rightarrow \Delta_2 \Rightarrow \Delta_3$$

Probability of $2^{-2(6+6)} = 2^{-24}$.

# Collision on 2 rounds (256)

- Best differential paths do not work as they impose a difference in hash value.

- Not possible with 3-slices in the kernel: we use 4-slice paths.

- With a probability of $2^{-32}$, the paths final differences are not on the hash part.

# Near-collision on 3 rounds (256)

- We can use the 3-slice kernel: 2 rounds with cost $2^{24}$, 1 more free round: 227 bits still without difference (generic $2^{64}$).

- We can control some bits in the last round, and then with cost $2^{44}$ we obtain collision on 247 bits (generic $2^{101}$).

# Distinguisher on 4 rounds (256)

- Consider the best path (6-6-6), and the neutral bits: bits of the message that won't affect the path if they are modified.

- There are 81 neutral bits out of the 1088 bits of the message block.

- Once we find a message that verifies the 2-round path, we can find $2^{81}$ more.

# Distinguisher on 4 rounds (256)

- Off-line complexity: $2^{25}$.

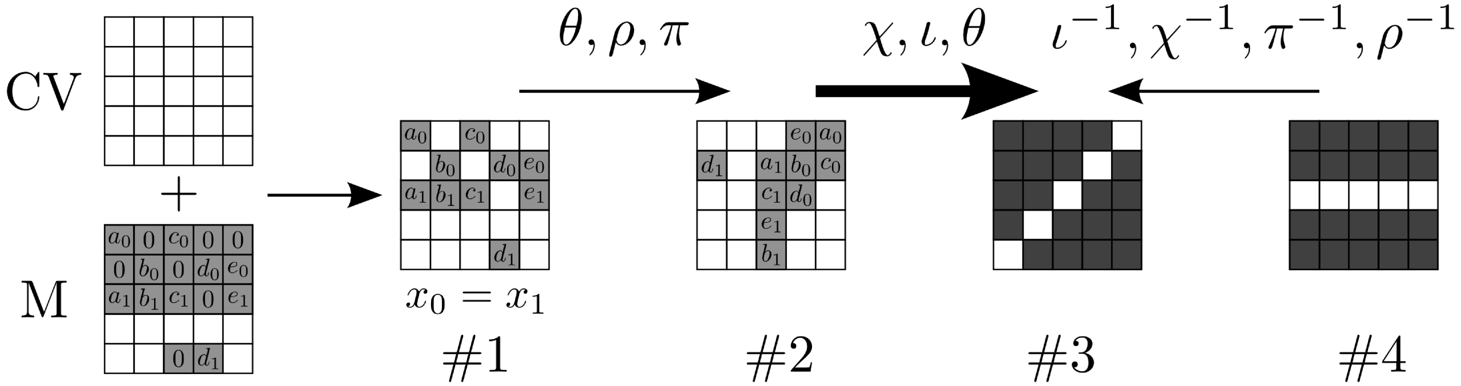
- There are 18 positions in the hash that will stay constant for any value of the 81 bits.

- On-line complexity: $2N$, for a false alarm probability of $2^{-18*N}$.

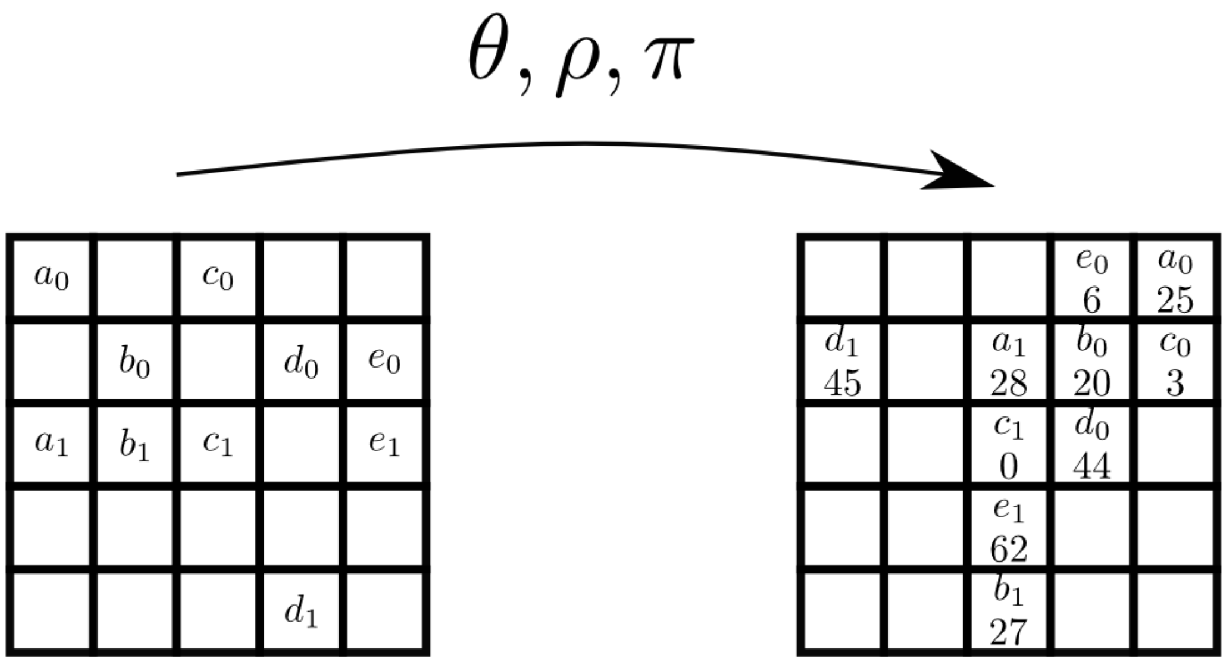
- Complexity $2^{25} + 2N \approx 2^{25}$.

# Preimage attack on 2 rounds

$2^{33}$ in time and $2^{29}$ in memory.



# Preimage attack on 2 rounds



# Preimage attack on 2 rounds

Treating first 48 slices (16 groups of 3):

- We consider three consecutive slices: $10 * 3 - 2 = 28$ unknown variables.

- We can compute from $\#2$ the output of $\theta$ on two slices: 10 known bits from the backward computation ($\#3$).

- $2^{28-10} = 2^{18}$ remain.

# Preimage attack on 2 rounds

- $8 \times 6$-slice groups: $2^{18+18-7-5} = 2^{24}$.
- $4 \times 12$-slice groups: $2^{24+24-16-5} = 2^{27}$.
- $2 \times 24$-slice groups: $2^{27+27-22-5} = 2^{27}$.
- $1 \times 48$-slice group: $2^{27+27-22-5} = 2^{27}$

(with 44 non-repeated variables).

- 16 remaining: 12-slice ($2^{27}$) and 4-slice ($2^{20}$) group.
- 12-s and 4-s: 15 common bits: $2^{27+20-15-5} = 2^{27}$.
- 16-s and 48-s: 44 common bits: $2^{27+27-44-5*2} = 1$.

# Preimage attack on 2 rounds

- Time complexity: $10 \times 2^{27} \times 2^{2} \approx 2^{33}$.
- Memory complexity: $4 * 2^{27} = 2^{29}$.

# Unaligned rebound attack

[Duc-Guo-Peyrin-Lei FSE 2012]

# Unaligned Rebound Attack FSE 2012

- Simultaneous and independent work from ours.
- Also look for low weight differential paths using Kernels (similar found).
- Distinguishers by inverting one round.
- Unaligned rebound attack: 8 rounds permutation distinguisher.

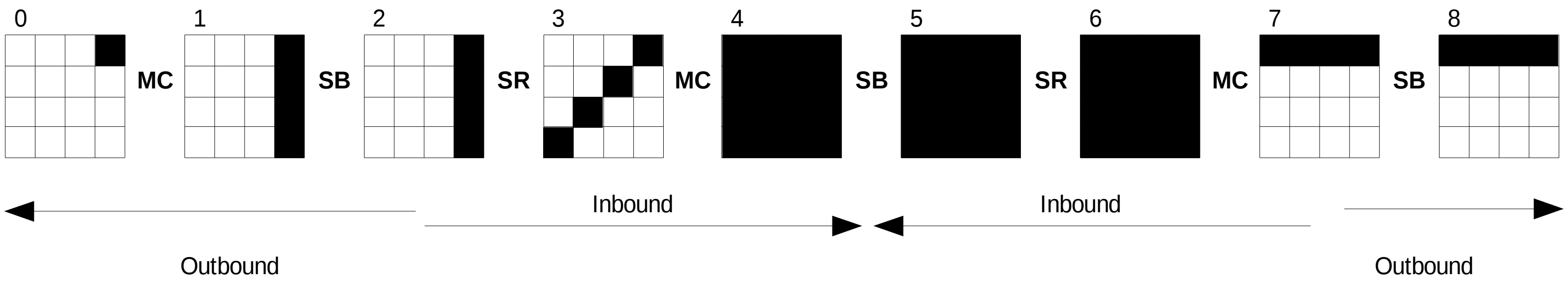
# Rebound attack [Mendel et al.09]

- Used for efficiently finding solutions of a differential path.

- Find solutions for an expensive part of the path in a cheap way, fill in the rest probabilistically.

- Largely used for building distinguishers on compression functions (mostly AES-based).

# Rebound attack [Mendel et al.09]




We choose the differential path.

Inbound phase:

1. we find differences for the black bytes that verify the path with a meet-in-the-middle (probability$=2^{-16}$ ).
2. then, for each difference match, $2^{16}$ values make the inbound possible.

Outbound phase: we need $2^{24}$ inbound solutions.

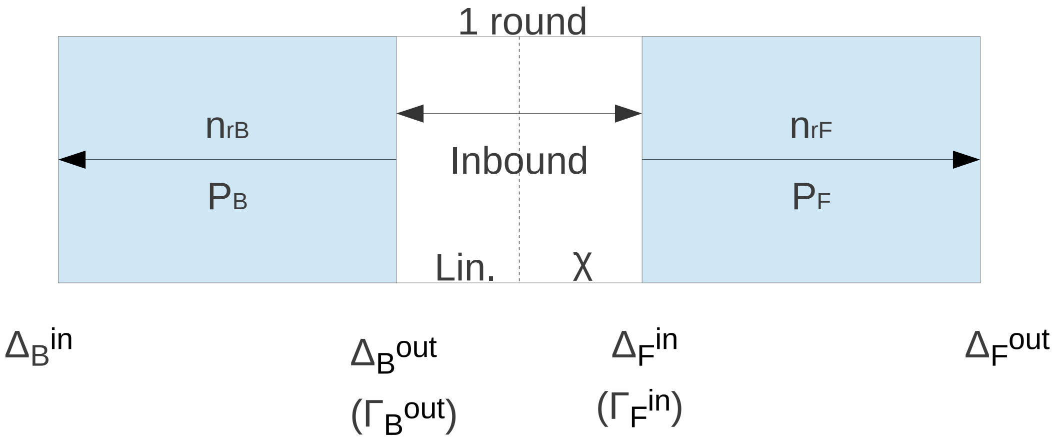
# Rebound attack [Mendel et al.09]

- Average cost of finding one solution for the inbound part $=1$, but minimal cost needs to be paid ($2^{16}$ in the example).

- As the remaining part of the path is verified with probability $2^{-24}$, we obtain a solution for the whole path with cost $2^{24}$.

- Generic cost in comparison: $2^{89}$.

# Unaligned Rebound attack [DGPW 12]




KECCAK has weak alignment: impossible to exploit truncated differentials or Super-Sboxes

$$C = n_F + n_B + \frac{1}{p_{match}} \frac{1}{\lceil p_F p_B N_{match} \rceil} + \frac{1}{p_B p_F}$$

$$\Gamma_B^{out} \Gamma_F^{in} = \frac{1}{p_{match}} \frac{1}{\lceil p_F p_B N_{match} \rceil}$$

# Buckets and Balls

- KECCAK has $64 * 5 = 320$ sboxes. Match through the inbound possible $\Rightarrow$ input active sboxes the same as the output active sboxes.
- Adapted buckets and balls problem $\Rightarrow$ all the sboxes need to be active.
- How are the bits distributed in the sboxes? DDT for a fixed input difference has all possible output differences with same probability, but the number of possible output differences depends strongly on the Hamming weight of the input.

# Forward Path

- Use one of the previous low weight differential paths (ex: 2 rounds, $2^{-24}$).
- Invert one round $\to$ are all sboxes in the middle active? (ex: $2^{-6*2}$, generates $2^{19-1.7}$ all-active-sbox inputs.)
- Add one or two rounds in the end.
- 64 equivalent paths by translation ($\Gamma_F^{in} = 2^{6+17.3} = 2^{23.3}$).

# Backward Path

- Same technique $\Rightarrow$ not enough paths.
- Second round: $X$ columns active, 2 bits per column, paths with 1 or 0 active bits per sbox.

- Half of the bits active for good probability of all sboxes active.
- Enough paths for the inbound, but more paths, less probability. We need: $p_B \geq \frac{1}{p_F N_{match}}$.
- First round: they spread.

# How do they compute complexities

- Incorrect to just take into account the average probability:

- $p_{match}$ increases with the hamming weight.

- $N_{match}$ decreases with the hamming weight.

- Computations for obtaining one solution take into account the hamming weight.

# Unaligned Rebound attack [DGPW 12]

- 8-round permutation distinguisher of Keccak-f[1600], $2^{491.47}$ compared to $2^{1057.6}$.

- Assumptions on some subparts of the distinguisher have been verified indepedently with implementations.

- Distinguisher implemented on the 100-bit version.

# Conclusions

- We presented the first practical results on the hash function reduced-round scenario of Keccak (4 out of 24).
- More rounds (Orr Dunkelman and Itai Dinur's talks).
- We briefly described the unaligned rebound attacks applied up to 8 rounds of Keccak permutation.
- Keccak (aka SHA-3) is a secure hash function with a (very) big security margin.