Unaligned Rebound Attack for KECCAK

Thomas Peyrin - NTU

joint work with Alexandre Duc, Jian Guo and Lei Wei

ASK 2011

Singapore
Outline

Introduction

Building differential paths for KECCAK

The rebound attack

The unaligned rebound attack for KECCAK

Results and future works
Outline

Introduction

Building differential paths for KECCAK

The rebound attack

The unaligned rebound attack for KECCAK

Results and future works
Current status of the SHA–3 competition

In december 2010, the NIST announced the five SHA–3 finalists: Blake, Grøstl, JH, KECCAK, Skein.

So far, none of them broken. It is very unlikely that this happens before the selection of the winner. So in order to compare their security, the cryptanalysts look for

* “easier” attack models:
  - near collisions
  - distinguishers (zero-sums, subspace, limited-birthday)
  - etc ...

* reduced variants:
  - lower number of rounds
  - only some internal function of the whole hash
  - etc ...

Here we will be analyzing the reduced-round KECCAK internal permutations in regards to differential distinguishers.
A sponge function has been proven to be indifferentiable from a random oracle up to $2^{c/2}$ calls to the internal permutation $P$. However, the best known generic attacks have the following complexity:

- **Collision:** $\min\{2^{n/2}, 2^{c/2}\}$
- **Second-preimage:** $\min\{2^n, 2^{c/2}\}$
- **Preimage:** $\min\{2^n, 2^c, \max\{2^{n-r}, 2^{c/2}\}\}$
Previous cryptanalysis results on KECCAK

So far, the results on KECCAK [B+08]:

- **J.-P. Aumasson et al. (2009):**
  zero-sum distinguishers up to 16 rounds of KECCAK-1600 internal permutation with complexity $2^{1024}$.

- **P. Morawiecki and M. Srebrny (2010):**
  small messages preimage attack using SAT solvers, up to 3 rounds.

- **D. Bernstein (2010):**
  a (second)-preimage attack on 8 rounds with complexity $2^{511.5}$ and $2^{508}$ bits of memory.

- **C. Boura et al. (2010-2011):**
  zero-sum partitions distinguishers to the full 24-round version of KECCAK-1600 internal permutation with complexity $2^{1590}$. 

Previous cryptanalysis results on KECCAK

Motivation:

- the zero-sum distinguishers proposed can attack more rounds (or the same number of rounds with better complexity) than the distinguishers we will present here. However:
  - their advantage to the generic complexity is very small (always a factor about 2), while in our case the gap will be huge
  - zero-sums are difficult to exploit in order to get collisions for example, while in our case we use differential properties
  - zero-sums partitions descriptions are in fact huge without using full KECCAK rounds in the descriptions

- because it is difficult to apply on KECCAK, there is no “differential analysis” provided by a third party yet.

- we focus on attacks with a complexity lower than $2^{b/2}$
The KECCAK internal state

The $b$-bit internal state of KECCAK can be viewed as a rectangular cuboid of $5 \times 5 \times w$ bits.
The KECCAK internal permutation

The $b$-bit KECCAK internal permutation $P$ applies $R$ rounds (for $b = 1600$ we have $R = 24$ rounds), each composed of the five following layers:
The KECCAK internal permutation

The $b$-bit KECCAK internal permutation $P$ applies $R$ rounds (for $b = 1600$ we have $R = 24$ rounds), each composed of the five following layers:

- $\theta$: linear mapping that provides diffusion for the state (the xor of the two columns $a[x - 1][.]z$ and $a[x + 1][.]z - 1$ is xored to the bit $a[x][y][z]$)
The KECCAK internal permutation

The $b$-bit KECCAK **internal permutation** $P$ applies $R$ rounds (for $b = 1600$ we have $R = 24$ rounds), each composed of the five following layers:

- **$\theta$:** linear mapping that provides diffusion for the state (the xor of the two columns $a[x - 1][\cdot][z]$ and $a[x + 1][\cdot][z - 1]$ is xored to the bit $a[x][y][z]$)

- **$\rho$:** linear mapping that provides diffusion between the slices of the state through intra-lane bit translations
The KECCAK internal permutation

The $b$-bit KECCAK internal permutation $P$ applies $R$ rounds (for $b = 1600$ we have $R = 24$ rounds), each composed of the five following layers:

- $\theta$: linear mapping that provides diffusion for the state (the xor of the two columns $a[x - 1][.][]z$ and $a[x + 1][.][]z - 1$ is xored to the bit $a[x][y][z]$)

- $\rho$: linear mapping that provides diffusion between the slices of the state through intra-lane bit translations

- $\pi$: linear mapping that provides diffusion in the state through transposition of the lanes.
The KECCAK internal permutation

The $b$-bit KECCAK internal permutation $P$ applies $R$ rounds (for $b = 1600$ we have $R = 24$ rounds), each composed of the five following layers:

- **$\theta$:** linear mapping that provides diffusion for the state (the xor of the two columns $a[x - 1][.]z$ and $a[x + 1][.]z - 1$ is xored to the bit $a[x][y]z$)

- **$\rho$:** linear mapping that provides diffusion between the slices of the state through intra-lane bit translations

- **$\pi$:** linear mapping that provides diffusion in the state through transposition of the lanes.

- **$\chi$:** non-linear mapping similar to $s = 5w$ Sboxes applied independently to each 5-bit row of the state
The KECCAK internal permutation

The $b$-bit KECCAK internal permutation $P$ applies $R$ rounds (for $b = 1600$ we have $R = 24$ rounds), each composed of the five following layers:

- **$\theta$**: linear mapping that provides diffusion for the state (the xor of the two columns $a[x - 1][.][]z$ and $a[x + 1][.][]z - 1$ is xored to the bit $a[x][y][z]$)

- **$\rho$**: linear mapping that provides diffusion between the slices of the state through intra-lane bit translations

- **$\pi$**: linear mapping that provides diffusion in the state through transposition of the lanes.

- **$\chi$**: non-linear mapping similar to $s = 5w$ Sboxes applied independently to each 5-bit row of the state

- **$\iota$**: adds round-dependant constants to the lane $a[0][0][.]$. We can forget about this layer since completely transparent in terms of differential paths.
The KECCAK internal permutation

The $b$-bit KECCAK internal permutation $P$ applies $R$ rounds (for $b = 1600$ we have $R = 24$ rounds), each composed of the five following layers:

- $\theta$: linear mapping that provides diffusion for the state (the xor of the two columns $a[x - 1][.]\[z]$ and $a[x + 1][.]\[z - 1]$ is xored to the bit $a[x][y][z]$)
- $\rho$: linear mapping that provides diffusion between the slices of the state through intra-lane bit translations
- $\pi$: linear mapping that provides diffusion in the state through transposition of the lanes.
- $\chi$: non-linear mapping similar to $s = 5w$ Sboxes applied independently to each 5-bit row of the state
- $\iota$: adds round-dependant constants to the lane $a[0][0][.]$. We can forget about this layer since completely transparent in terms of differential paths.

One round is now composed of:

- a linear layer $\lambda = \pi \circ \rho \circ \theta$
- a non-linear Sbox layer $\chi$
Outline

Introduction

Building differential paths for KECCAK

The rebound attack

The unaligned rebound attack for KECCAK

Results and future works
The diffusion in KECCAK

Diffusion in KECCAK mostly provided by $\theta$, since:

- $\pi$ and $\rho$ layers only change bit positions
- diffusion of the Sboxes in $\chi$ layer is very small.

Good diffusion of $\theta$: 

![Diagram showing diffusion process in KECCAK](image-url)
The diffusion in KECCAK

Diffusion in KECCAK mostly provided by $\theta$, since:

- $\pi$ and $\rho$ layers only change bit positions
- diffusion of the Sboxes in $\chi$ layer is very small.

Excellent diffusion of $\theta^{-1}$:

![Diagram showing diffusion process in KECCAK](image)
The column parity kernel for $\theta$

An even number of active bits gives no diffusion through $\theta$ (column parity kernel, CPK):

![Diagram showing no diffusion through $\theta$]
The differential path search for KECCAK

Our goal is of course to minimize as much as possible the effect of the diffusion. When looking for a bitwise differential path, the branching in the search only comes from $\chi$ (for a given input, all valid transitions have the same success probability through the Sbox).

The core algorithm is simple:

- **Precomputation:** for every possible slice input difference, we precompute and store the best differential transitions through $\chi$, i.e. the ones that will minimize the diffusion through the next $\theta$ (favor CPK, low Hamming weight).

- **Keep repeating:**
  - start with a difference in $a_1$ composed of only $k$ CPK, with $k$ small
  - compute forward by choosing random candidates among the best slice transitions
  - if the current path tested is good, compute one round backward (about $2k$ active sboxes)

\[
\begin{aligned}
a_0 & \xleftarrow{\lambda^{-1}} b_0 \xleftarrow{\chi^{-1}} a_1 \xrightarrow{\lambda} b_1 \xrightarrow{\chi} a_2 \xrightarrow{\lambda} b_2 \xrightarrow{\chi} a_3 \xrightarrow{\lambda} b_3 \cdots
\end{aligned}
\]
### Differential paths results on KECCAK

**Table:** Best differential path results for each version of **KECCAK** internal permutations, for 1 up to 5 rounds (red = new results).

<table>
<thead>
<tr>
<th>$b$</th>
<th>best differential path probability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 rd</td>
</tr>
<tr>
<td>100</td>
<td>$2^{-2}$ (2)</td>
</tr>
<tr>
<td>200</td>
<td>$2^{-2}$ (2)</td>
</tr>
<tr>
<td>400</td>
<td>$2^{-2}$ (2)</td>
</tr>
<tr>
<td>800</td>
<td>$2^{-2}$ (2)</td>
</tr>
<tr>
<td>1600</td>
<td>$2^{-2}$ (2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$b$</th>
<th>best differential path probability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4 rds</td>
</tr>
<tr>
<td>100</td>
<td>$2^{-30}$ (4 - 8 - 10 - 8)</td>
</tr>
<tr>
<td>200</td>
<td>$2^{-46}$ (11 - 9 - 8 - 8)</td>
</tr>
<tr>
<td>400</td>
<td>$2^{-84}$ (16 - 14 - 12 - 42)</td>
</tr>
<tr>
<td>800</td>
<td>$2^{-109}$ (12 - 12 - 12 - 73)</td>
</tr>
<tr>
<td>1600</td>
<td>$2^{-142}$ (12 - 12 - 12 - 106)</td>
</tr>
</tbody>
</table>
Simple distinguishers

**Obvious distinguisher:**
for a differential path $\Delta_{in} \leftrightarrow \Delta_{out}$ with success probability $P > 2^{-b}$ (the generic algorithm finds such a pair with complexity $2^b$)

**Use the freedom degrees (+1 round):**
add an extra round for free to the left (or to the right) by fixing the Sboxes values for this round. Same overall complexity (same generic complexity)

**Add an extra round to the left and to the right (+2 rounds):**
without controlling the new differential transitions (i.e. same complexity). This will increase the amount of reachable input and output differences (from 1 to $IN$ and 1 to $OUT$) and therefore reduce the generic complexity (limited-birthday distinguishers [GP10]): $\max\{\sqrt{2^b/IN}, \sqrt{2^b/OUT}, 2^b/(IN \cdot OUT)\}$
Simple distinguishers

Obvious distinguisher:
for a differential path $\Delta_{in} \leftrightarrow \Delta_{out}$ with success probability $P > 2^{-b}$ (the generic algorithm finds such a pair with complexity $2^b$)

Use the freedom degrees (+1 round):
add an extra round for free to the left (or to the right) by fixing the Sboxes values for this round. Same overall complexity (same generic complexity)

Add an extra round to the left and to the right (+2 rounds):
without controlling the new differential transitions (i.e. same complexity). This will increase the amount of reachable input and output differences (from 1 to $IN$ and 1 to $OUT$) and therefore reduce the generic complexity (limited-birthday distinguishers [GP10]): $\max\{\sqrt{2^b/IN}, \sqrt{2^b/OUT}, 2^b/(IN \cdot OUT)\}$
Simple distinguishers

Obvious distinguisher:
for a differential path $\Delta_{in} \leftrightarrow \Delta_{out}$ with success probability $P > 2^{-b}$ (the generic algorithm finds such a pair with complexity $2^b$)

Use the freedom degrees (+1 round):
add an extra round for free to the left (or to the right) by fixing the Sboxes values for this round. Same overall complexity (same generic complexity)

Add an extra round to the left and to the right (+2 rounds):
without controlling the new differential transitions (i.e. same complexity). This will increase the amount of reachable input and output differences (from 1 to $IN$ and 1 to $OUT$) and therefore reduce the generic complexity (limited-birthday distinguishers [GP10]): $\max\{\sqrt{2^b/IN}, \sqrt{2^b/OUT}, 2^b/(IN \cdot OUT)\}$
Outline

Introduction

Building differential paths for KECCAK

The rebound attack

The unaligned rebound attack for KECCAK

Results and future works
The rebound attack [M+09] (example with AES-like permutation):
The **rebound attack** [M+09] (example with AES-like permutation):

- **step 1:** choose input difference $\Delta_{\text{in}}$ and output difference $\Delta_{\text{out}}$ of the inbound phase ...
Rebound attack and improvements

The **rebound attack** [M+09] (example with AES-like permutation):

- **step 1**: choose input difference $\Delta_{in}$ and output difference $\Delta_{out}$ of the inbound phase ...
- **step 2**: ...and propagate those **differences** forward and backward up to the middle layer of Sboxes, until reaching a differential match (with probability $p_{\text{match}}$)

![Diagram of the rebound attack process](image-url)
Rebound attack and improvements

The **rebound attack** [M+09] (example with AES-like permutation):

- **step 1:** choose input difference $\Delta_{in}$ and output difference $\Delta_{out}$ of the inbound phase ...
- **step 2:** ...and propagate those **differences** forward and backward up to the middle layer of Sboxes, until reaching a differential match (with probability $p_{\text{match}}$)
- **step 3:** once a differential match obtained, deduce and generate all the $N_{\text{match}}$ valid Sbox values $V$
Rebound attack and improvements

The **rebound attack** \([M+09]\) (example with AES-like permutation):

- **step 1:** choose input difference \(\Delta_{in}\) and output difference \(\Delta_{out}\) of the inbound phase ...
- **step 2:** ...and propagate those **differences** forward and backward up to the middle layer of Sboxes, until reaching a differential match (with probability \(p_{\text{match}}\))
- **step 3:** once a differential match obtained, deduce and generate all the \(N_{\text{match}}\) valid Sbox **values** \(V\)
- **step 4:** propagate the **values and differences** forward and backward and check if the differential path is entirely verified (with probability \(p_F\) and \(p_B\))

![Diagram of the rebound attack on KECCAK]

\(\Delta_{in} \rightarrow V \rightarrow \Delta_{out} \)
The overall complexity is \( \frac{1}{p_{\text{match}}} \cdot \left\lceil \frac{1}{p_F \cdot p_B \cdot N_{\text{match}}} \right\rceil + \frac{1}{p_B \cdot p_F} \), since:

- we need to start with at least \( p_{\text{match}}^{-1} \) pairs of differences for the inbound before finding a differential match in the middle
- we need to generate at least \( p_B^{-1} \cdot p_F^{-1} \) valid inbound values in order to find a solution for the entire path

Some improvements exist:

- **Super-Sbox [L+09,GP10]**: merge two rounds in the middle in order to build a layer of bigger Sboxes (gain of one round)
- **Non-full active [S+10]**: do not necessarily use a full active state in the middle (lower complexity)
Why rebound is hard on KECCAK?

Our goal: take the best differential path on \( x \) rounds of KECCAK, and merge it using the rebound to create a \((2x + 1)\)-round one (we hope for 9 rounds at max for a complexity < \(2^{512}\)).

But there are many problems for KECCAK:

- there is (by far !) **not enough differential paths** with good probability
- the differential match probability of the KECCAK Sbox depends on the input and output difference mask (see its DDT) ...
- ... but fortunately the distribution of output difference probabilities is the same when the input difference hamming weight is fixed

Moreover, the improvements will not apply:

- **alignement in KECCAK is bad** (see designers recent article at ECRYPT HASH3), thus the Super-Sbox improvement cannot be used
- we will see later that it is very hard to build non-full active differential paths using rebound technique
# The KECCAK Sbox DDT

| \(\Delta_{in}\) | \(\Delta_{out}\) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 1A | 1B | 1C | 1D | 1E | 1F |
| 00 | 32 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 01 | - | 8 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 02 | - | - | 8 | 8 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 03 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 04 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 05 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 06 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 07 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 08 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 09 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 0A | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 0B | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 0C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 0D | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 0E | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 0F | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 10 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 11 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 12 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 13 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 14 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 15 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 16 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 17 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 18 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 19 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 1A | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 1B | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 1C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 1D | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 1E | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 1F | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
Outline

Introduction

Building differential paths for KECCAK

The rebound attack

The unaligned rebound attack for KECCAK

Results and future works
We consider an inbound composed of one KECCAK round.

Due to the very good diffusion of $\theta^{-1}$, the amount of forward paths will be small. In order to have a chance to find at least one match for the inbound, we will need a lot of backward paths.

In the following, we will focus on the case KECCAK-1600 but our framework allows to apply the unaligned rebound attack on any version.
We consider an inbound composed of one KECCAK round.

Due to the very good diffusion of $\theta^{-1}$, the amount of forward paths will be small. In order to have a chance to find at least one match for the inbound, we will need a lot of backward paths.

In the following, we will focus on the case KECCAK-1600 but our framework allows to apply the unaligned rebound attack on any version.
Balls and bucket problem

In order for a differential match to happen during the inbound, we first need the exact same set of Sboxes to be active forward and backward. We modeled this with a limited capacity balls and buckets problem:

**Theorem**

Given a set $B$ of $s$ buckets of capacity 5 in which we throw $x_B$ balls and a set $F$ of $s$ buckets of capacity 5 in which we throw $x_F$ balls, the probability that $B$ and $F$ have the same pattern of empty buckets is given by

$$p_{\text{pattern}}(s, x_B, x_F) = \frac{1}{\binom{5s}{x_B} \binom{5s}{x_F}} \sum_{i=0}^{s} b_{\text{bucket}}(x_B, s - i) b_{\text{bucket}}(x_F, s - i) \binom{s}{i},$$

where $b_{\text{bucket}}(x, s) = \sum_{i=}^{\lceil n/5 \rceil} (-1)^i \binom{s}{i} \binom{5i}{n}$ if $s \leq n \leq 5s$ and 0 otherwise. The average number $n_{\text{pattern}}$ of non-empty buckets if both experiments results follow the same pattern is given by

$$n_{\text{pattern}}(s, x_B, x_F) = \frac{\sum_{i=0}^{s} b_{\text{bucket}}(x_B, s - i) b_{\text{bucket}}(x_F, s - i) \binom{s}{i} (s - i)}{\sum_{i=0}^{s} b_{\text{bucket}}(x_B, s - i) b_{\text{bucket}}(x_F, s - i) \binom{s}{i}}.$$
Balls and bucket problem

In order for a differential match to happen during the inbound, we first need the exact same set of Sboxes to be active forward and backward.

We modeled this with a limited capacity balls and buckets problem:

**Theorem**

**Conclusion:** for our range of difference bit Hamming weights (not too small) on the input and output of the inbound:

- it is very likely that a match on the active Sboxes pattern happens ($p_{\text{pattern}}$ is high)
- when it happens, it is very likely that all sboxes are active ($n_{\text{pattern}} = s$).
**The forward paths**

<table>
<thead>
<tr>
<th>INBOUND</th>
<th>1st round</th>
<th>2nd round</th>
<th>3rd round</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \lambda )</td>
<td>( \chi )</td>
<td>( \lambda )</td>
</tr>
</tbody>
</table>

**Active bits**

<table>
<thead>
<tr>
<th>( \log_2 ) proba</th>
</tr>
</thead>
</table>

| Number of paths |
The forward paths

<table>
<thead>
<tr>
<th>INBOUND</th>
<th>1st round</th>
<th>2nd round</th>
<th>3rd round</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\lambda)</td>
<td>(\chi)</td>
<td>(\lambda)</td>
<td>(\chi)</td>
</tr>
</tbody>
</table>

Active bits:
- 1st round: 6 \(\leftarrow\) 6
- 2nd round: 6 \(\leftarrow\) 6
- 3rd round: 6 \(\rightarrow\) 6

\(\log_2\) proba:
- 1st round: -12
- 2nd round: -12
- 3rd round: -12

Number of paths:
- 1st round: \(2^6\)
- 2nd round: \(2^6\)
- 3rd round: \(2^6\)
The forward paths

1st round | 2nd round | 3rd round

| INBOUND | λ | χ | λ | χ | λ | χ |

Active bits

6 ← 6 | 6 ← 6 | 6 → *

log2 proba

-12 | -12 | 0

Number of paths

2^6 | 2^6 | 2^6 | 2^18
The forward paths

INBOUND

1st round: \( \lambda \) → \( \chi \)
2nd round: \( \lambda \) → \( \chi \)
3rd round: \( \lambda \) → \( \chi \)

Active bits:
- 1st round: \(* \leftarrow 6\)
- 2nd round: \(6 \leftarrow 6\)
- 3rd round: \(6 \rightarrow *\)

log2 proba:
- 1st round: \([-24,-12]\)
- 2nd round: \(-12\)
- 3rd round: \(0\)

Number of paths:
- 1st round: \(2^{25}\)
- 2nd round: \(2^6\)
- 3rd round: \(2^6\)
- Total: \(2^{18}\)
The forward paths

1st round 2nd round 3rd round

\[ \lambda \quad \chi \quad \lambda \quad \chi \quad \lambda \quad \chi \]

**Active bits**
- INBOUND: 320 act.
- 1st round: \( \lambda \) active
- 2nd round: \( \chi \) active
- 3rd round: \( \lambda \) active

**sboxes**
- INBOUND: 320 active
- 1st round: 6 active
- 2nd round: 6 active
- 3rd round: 6 active

**log2 proba**
- INBOUND: \([-24,-12]\]
- 1st round: -12
- 2nd round: -12
- 3rd round: 0

**Number of paths**
- INBOUND: \(2^{23.3}\)
- 1st round: \(2^{25}\)
- 2nd round: \(2^6\)
- 3rd round: \(2^6\)
- OUTBOUND: \(2^{18}\)
The backward paths

1st round | 2nd round | 3rd round

\[ \lambda \chi \lambda \chi \lambda \chi \]

Active bits

\[ \log_2 \text{proba} \]

Number of paths
The backward paths

1st round | 2nd round | 3rd round

\( \lambda \) | \( \chi \) | \( \lambda \) | \( \chi \) | \( \lambda \) | \( \chi \)

16 ← 16 16 → 16

-32 -32

\( 2^{77.7} \) \( 2^{77.7} \) \( 2^{77.7} \)

Active bits

log2 proba

Number of paths
The backward paths

1st round 2nd round 3rd round

\( \lambda \) \( \chi \) \( \lambda \) \( \chi \) \( \lambda \) \( \chi \)

\( \ast \leftarrow 16 \)

16 \rightarrow 16

0 -32

\leq 2^{128.4} \quad 2^{77.7} \quad 2^{77.7}

Active bits

\log_2 \text{proba}

Number of paths
The backward paths

1st round | 2nd round | 3rd round

$\lambda$ | $\chi$ | $\lambda$ | $\chi$ | $\lambda$ | $\chi$

$\ast \leftarrow 16$ | $16 \rightarrow 24$

0 | -32

$\leq 2^{128.4}$ | $2^{77.7}$ | $2^{99.4}$

Active bits

$log_2$ proba

Number of paths
The backward paths

1st round 2nd round 3rd round

\[ \lambda \chi \quad \lambda \chi \quad \lambda \chi \]

\[ \text{Active bits} \]

\[ * \leftarrow 16 \quad 16 \to 24 \quad * \]

\[ 0 \quad -32 \quad \geq -418 \]

\[ \leq 2^{128.4} \quad 2^{77.7} \quad 2^{99.4} \quad 2^{478} \]

\[ \text{Number of paths} \]
Overall complexity

The differential matching probability is $p_{\text{match}} = 2^{-491.5}$

The number of solutions obtained per match is $N_{\text{match}} = 2^{486.8}$

The total complexity is $2^{491.5}$ computations
Distinguishers on KECCAK-1600 permutation

\[ \Delta_{in} \quad \Delta_{out} \]

- \[ \text{IN} \]
  - BACKWARD DIFFERENTIAL PATH
  - INBOUND
  - FORWARD DIFFERENTIAL PATH

\[ \text{OUT} \]

Limited birthday problem on a 1600-bit permutation with:
- \(|IN| \leq 2^{128.4}\)
- \(|OUT| = 2^{18}\)

We have a generic complexity of \(2^{1453.6} \approx 2^{491.5}\) computations.

⇒ 7 rounds can be distinguished
Limited birthday problem on a 1600-bit permutation with:
- $|IN| \leq 2^{128.4}$
- $|OUT| \leq 2^{414}$

We have a generic complexity of $2^{1057.6} > 2^{491.5}$ computations.

$\Rightarrow$ 8 rounds can be distinguished
Distinguishers on KECCAK-1600 permutation

Limited birthday problem on a 1600-bit permutation with:

- $|IN| \leq 2^{1142.8}$
- $|OUT| \leq 2^{414}$

We have a generic complexity of $2^{228.6} < 2^{491.5}$ computations.

$\Rightarrow$ 9 rounds cannot be distinguished
| Introduction | Building differential paths for KECCAK | The rebound attack | The unaligned rebound attack for KECCAK | Results and future works | Results |
Introduction
Differential paths
Rebound on KECCAK
Rebound on KECCAK
Results

Overall results

**Table:** Best differential distinguishers complexities for each version of KECCAK internal permutations, for 1 up to 8 rounds.

<table>
<thead>
<tr>
<th>$b$</th>
<th>1 rd</th>
<th>2 rds</th>
<th>3 rds</th>
<th>4 rds</th>
<th>5 rds</th>
<th>6 rds</th>
<th>7 rds</th>
<th>8 rds</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$2^2$</td>
<td>$2^8$</td>
<td>$2^{19}$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>200</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$2^2$</td>
<td>$2^8$</td>
<td>$2^{20}$</td>
<td>$2^{46}$</td>
<td>-</td>
</tr>
<tr>
<td>400</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$2^2$</td>
<td>$2^8$</td>
<td>$2^{24}$</td>
<td>$2^{84}$</td>
<td>-</td>
</tr>
<tr>
<td>800</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$2^2$</td>
<td>$2^8$</td>
<td>$2^{32}$</td>
<td>$2^{109}$</td>
<td>-</td>
</tr>
<tr>
<td>1600</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$2^2$</td>
<td>$2^8$</td>
<td>$2^{32}$</td>
<td>$2^{142}$</td>
<td>$2^{491.5}$</td>
</tr>
</tbody>
</table>

Our method and our model have been **verified in practice** on reduced versions of KECCAK.
Future works

Use the differential path search tool and the unaligned rebound for

- the **recent collision/preimage KECCAK challenges**:  
  - the variants with little number of rounds seem clearly reachable (we already found collisions for 1 and 2-round challenges)
  - we need to find a smart way to use the freedom degrees when several blocks are needed

- differential distinguisher on the hash function, so far we have:
  - 3-round fixed-IV distinguisher
  - 5-round chosen-IV distinguisher

Analyze other functions with our framework:

- PRESENT
- SPONGENT
- JH
Thank you!