A SAT-based preimage analysis of reduced KECCAK hash functions

Paweł Morawiecki
Section of Informatics, University of Commerce, Kielce, Poland
pawelm@wsh-kielce.edu.pl

and

Marian Srebrny
Inst of Comp Sci, Polish Academy of Sciences, Warsaw, Poland; and
Section of Informatics, University of Commerce, Kielce, Poland
marians@ipipan.waw.pl

The Second SHA-3 Candidate Conference
Santa Barbara, CA, August 23-24, 2010
Overview

- Highlights of our results (relevant to the SHA-3 Contest)
- Keccak – some necessary basics
- SAT-based cryptanalysis – some necessary basics
- Our CryptLogVer toolkit
- Our experimental results on Keccak
- 2 other SHA-3 candidates
- Related work
- Conclusion and future directions
Highlights of our results (relevant to the SHA-3 Contest)

- Keccak is strong w.r.t. our analysis.
  - Our results suggest the strength of Keccak SHA-3 design.
  - We found a preimage only for very much reduced versions of Keccak.
  - 3-round Keccak-f[1600] with 40 unknown message bits.

- Our CNF size estimates for Grøstl and CubeHash suggest their very strong resistance against SAT-based cryptanalysis.
Two main input parameters
- r (called bitrate, default 1024 bits)
- c (called capacity, default 576 bits)

These parameters determine the state size, default 1600 bits.

Keccak family variants operate also on smaller states:
- 25- 50- 100- 200- 400- and 800-bit state
SAT-based cryptanalysis

Relatively recent method

Express your cryptographic primitive (e.g. a cipher, a hash function) in CNF and let a SAT-solver search for a solution.

\[
\varphi(x_1, \ldots, x_{1024}, \ldots, y_1, \ldots, y_{256})
\]
3 applications:

1. Altera Quartus II Web Edition (free of charge)
2. Our CNF conversion:
   - From a system of boolean equations
   - To equisatisfiable CNF
   - In linear time, with many extra variables
3. SAT-solver PrecoSAT

\[
\begin{align*}
x_1 \oplus x_2 \oplus x_3 &= 0 \\
x_4 \oplus x_5 \oplus x_6 &= 0 \\
x_4 v_1 \oplus v_5 v_6 &= 0 \\
v_1 \oplus v_2 \oplus v_3 &= 0 \\
\ldots
\end{align*}
\[
\begin{align*}
x_1 \lor z_1 \lor \overline{z_2} \\
v_2 \lor \overline{z_2} \lor \overline{z_2} \lor x_2 \\
v_3 \lor z_1 \lor \overline{z_2} \\
x_1 \lor z_1 \\
\overline{v_1} \lor v_2 \lor v_3 \\
\ldots
\end{align*}
\[
\begin{align*}
x_1 &= 0 \\
x_2 &= 0 \\
v_1 &= 1 \\
z_1 &= 1 \\
z_2 &= 0 \\
\ldots
\end{align*}
\]
## Our experimental results on Keccak

<table>
<thead>
<tr>
<th>Function</th>
<th>Number of rounds</th>
<th>Message size [bits]</th>
<th>Hash size [bits]</th>
<th>SAT-solver attack</th>
<th>Exhaustive search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keccak[1024,576]</td>
<td>3</td>
<td>24</td>
<td>1024</td>
<td>$2^0$</td>
<td>$2^1$</td>
</tr>
<tr>
<td>Keccak[1024,576]</td>
<td>3</td>
<td>32</td>
<td>1024</td>
<td>$2^{3,3}$</td>
<td>$2^9$</td>
</tr>
<tr>
<td>Keccak[1024,576]</td>
<td>3</td>
<td>40</td>
<td>1024</td>
<td>$2^{10,8}$</td>
<td>$2^{17}$</td>
</tr>
<tr>
<td>Keccak[120,80]</td>
<td>3</td>
<td>24</td>
<td>80</td>
<td>$2^{2,5}$</td>
<td>$2^{15}$</td>
</tr>
<tr>
<td>Keccak[120,80]</td>
<td>3</td>
<td>32</td>
<td>80</td>
<td>$2^{5,7}$</td>
<td>$2^{5}$</td>
</tr>
<tr>
<td>Keccak[120,80]</td>
<td>3</td>
<td>40</td>
<td>80</td>
<td>$2^{15,7}$</td>
<td>$2^{13}$</td>
</tr>
<tr>
<td>Keccak[24,26]</td>
<td>4</td>
<td>24</td>
<td>24</td>
<td>$2^{12,1}$</td>
<td>$2^{-13,5}$</td>
</tr>
<tr>
<td>Keccak[24,26]</td>
<td>5</td>
<td>24</td>
<td>24</td>
<td>$2^{12,8}$</td>
<td>$2^{-13,1}$</td>
</tr>
</tbody>
</table>

On Intel Xeon 2.5 GHz with 48 hours time limit. Exhaustive search with speed-optimized implementation of Keccak Team.
2 other SHA-3 candidates

- Grøstl with 256-bit hash calls the AES S-box 1280 times. The AES S-box has been coded by our CryptLogVer toolkit as a formula with about 4800 clauses and 900 variables. A straightforward calculation gives at least 1280 * 4800 = 6 mln 144 thousand clauses in total.

- Bernstein’s CubeHash would have about 1 mln 760 thousand clauses and 270 thousand variables.

- Full Keccak-f[1600] has 775 thousand clauses and 181 thousand variables.

- Hence, no SAT-based attack looks feasible (with no extra financial effort). From this perspective: Keccak, Grøstl, and CubeHash seem to be very strong.

- Full SHA-1 has CNF with 181k clauses and 31k variables.
Related work

- The Keccak team used SAGE (computer algebra software) to solve the CICO problems with 12 unknown input bits, up to 8 rounds and with Keccak-f state widths from 40 to 200. As the number of unknown input bits grows, their method quickly becomes infeasible.

- Courtois and Bard, 2006: SAT-solvers can be a better option for cryptanalysis than computer algebra due to their much lower memory requirements.

- Rivest et al. tested their MD-6 with logical (SAT-based) analysis. They found collisions only for the first 11-rounds, as the best result.

- For full SHA-1, we obtained its CNF with 181k clauses and 31k variables. We found a short preimage for 27-round SHA-1 (out of its full 80 rounds).
Conclusion and future directions

- Our results suggest the strength of Keccak SHA-3 design.
  - We found a preimage only for very much reduced versions of Keccak.
  - E.g., 3-round Keccak-f[1600] with 40 unknown message bits.
- It might be interesting to try extrapolating our results for the full Keccak-f function (using a technique of Soos–Nohl–Castelluccia 2009)
- Grøstl and CubeHash seem to be also very strong against SAT-based cryptanalysis.
- SAT-based analysis of the other SHA-3 candidates might be interesting.
Thank you for your attention!

Questions, remarks, comments?