Cryptanalysis of CubeHash

ACNS 2009

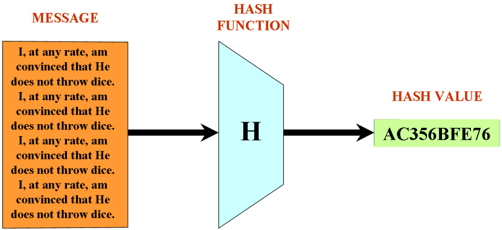
Eric Brier and Thomas Peyrin
Ingenico

June 4th 2009 - Paris

Outline

Outline

What is a hash function ?



Should be resistant to (at least):

- collision attacks ($2^{n/2}$)
- 2nd preimage attacks (2^n)
- preimage attacks (2^n)

Current state of the art

- MD4, MD5, SHA-0, SHA-1 are broken. SHA-2 is unbroken yet but presents the same ”design core” as the MD-SHA family.
- SHA-2 is not resistant to length extension attacks or multicollision attacks (because of Merkle-Damgard).
- NIST response is **SHA-3 competition**:
  - from October 2008 until end 2012.
  - 64 submitted candidates.
  - 51 accepted for 1st round.
  - among them : CubeHash !

CubeHash (Dan Bernstein - 2008)

Good points:

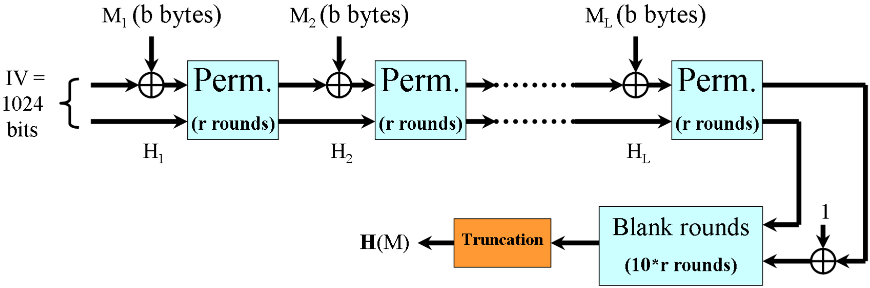
- very easy to understand
- very easy to analyze
- very easy to implement
- very easily tunable
- quite fast depending on the version considered (Cubehash-1/8: 2.5 c/B)

Bad points:

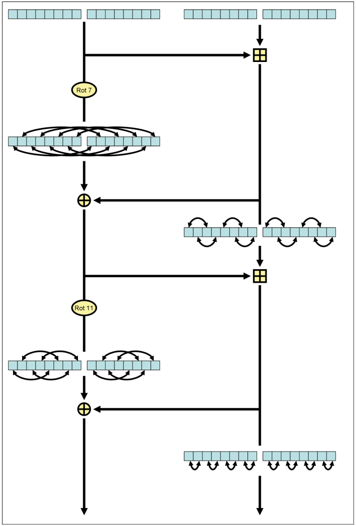
- too simple ?
- too much tunable (too many different versions to analyze)
- originally lacks security analysis
- quite slow depending on the version considered (Cubehash-8/1: 160 c/B)

CubeHash-r/b algorithm

$$\textbf{message + padding} = \mathbf{M} = \mathbf{M_1} \,||\, \mathbf{M_2} \,||\, \dots \,||\, \mathbf{M_L}$$



CubeHash round function



CubeHash security claims and previous work

Known results:

- meet-in-the-middle attack for preimage resistance when *b* is big (submission document)
- some symmetric states are stable (ACISP 2009)
- fixed points found (ACISP 2009)
- some biases can be detected after 8 rounds (ACISP 2009)
- collision for very reduced variants (NIST forum 2008)

New results:

- collision attacks to many CubeHash variants (some of them slower than 20c/B)

Outline

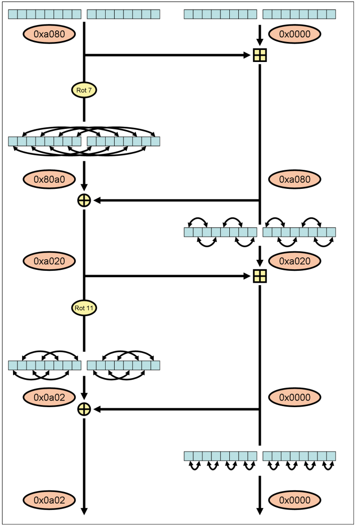
Introduction to CubeHash

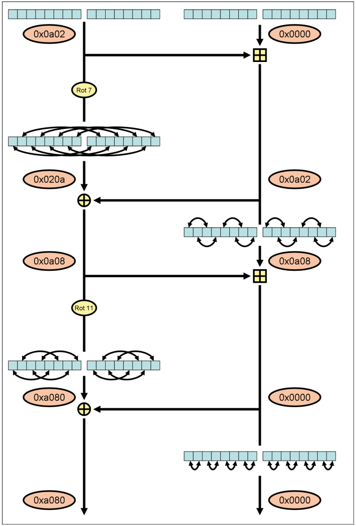
Truncated differentials paths

Linear differential paths

Results

Truncated differential paths





Derived equations

System 1 (for Δ_1) :

$$
\begin{aligned}
(X_{24} + X_8) \oplus X_0^{\lll 7} &= (X_{24} + X'_8) \oplus X'^{\lll 7}_0 \\
X_8 + [(X_{26} + X_{10}) \oplus X_2^{\lll 7}] &= X'_8 + [(X_{26} + X_{10}) \oplus X'^{\lll 7}_2] \\
X_0 + [(X_{18} + X_2) \oplus X_{10}^{\lll 7}] &= X'_0 + [(X_{18} + X'_2) \oplus X_{10}^{\lll 7}] \\
X_2 + [(X_{16} + X_0) \oplus X_8^{\lll 7}] &= X'_2 + [(X_{16} + X'_0) \oplus X'^{\lll 7}_8]
\end{aligned}
$$

System 2 (for Δ_2) :

$$
\begin{aligned}
(X_{30} + X_{14}) \oplus X_6^{\lll 7} &= (X_{30} + X'_{14}) \oplus X'^{\lll 7}_6 \\
X_{14} + [(X_{28} + X_{12}) \oplus X_4^{\lll 7}] &= X'_{14} + [(X_{28} + X_{12}) \oplus X'^{\lll 7}_4] \\
X_6 + [(X_{20} + X_4) \oplus X_{12}^{\lll 7}] &= X'_6 + [(X_{20} + X'_4) \oplus X_{12}^{\lll 7}] \\
X_4 + [(X_{22} + X_6) \oplus X_{14}^{\lll 7}] &= X'_4 + [(X_{22} + X'_6) \oplus X'^{\lll 7}_{14}]
\end{aligned}
$$

Solving equations

$$
\begin{aligned}
(A + X_8) \oplus X_0^{\lll 7} &= (A + X'_8) \oplus X'_0{}^{\lll 7} \\
X_8 + [(B + C) \oplus X_2^{\lll 7}] &= X'_8 + [(B + C) \oplus X'_2{}^{\lll 7}] \\
X_0 + [(D + X_2) \oplus C^{\lll 7}] &= X'_0 + [(D + X'_2) \oplus C^{\lll 7}] \\
X_2 + [(E + X_0) \oplus X_8^{\lll 7}] &= X'_2 + [(E + X'_0) \oplus X'_8{}^{\lll 7}]
\end{aligned}
$$

Solving equations

$$
\begin{aligned}
(A + X_8) \oplus X_0^{\lll 7} &= (A + X'_8) \oplus X'^{\lll 7}_0 \\
X_8 + [(B + C) \oplus X_2^{\lll 7}] &= X'_8 + [(B + C) \oplus X'^{\lll 7}_2] \\
X_0 + [(D + X_2) \oplus C^{\lll 7}] &= X'_0 + [(D + X'_2) \oplus C^{\lll 7}]
\end{aligned}
$$

- Pick random values for X_2 and X'_2
- We set $X'_8 - X_8 = \Delta_8$ and $X'_0 - X_0 = \Delta_0$
- We set $Y = X_8 + A$ and $Y' = X'_8 + A$
- We get: $Y \oplus (\Delta_8 + Y) = X_0^{\lll 7} \oplus (\Delta_0 + X_0)^{\lll 7}$.

$x \oplus (x + k)$ is always equal to `0xffffffff` when $x = \overline{k}/2$ and when the least significant bit of k is equal to one.

Truncated differential attack results

Results:

- a collision for CubeHash-1/36 in 2^{32} operations.
- a collision for CubeHash-2/36 in 2^{96} operations.
- ... seems hard to go further !

Outline

Introduction to CubeHash

Truncated differentials paths

Linear differential paths

Results

Linear differential paths

- try to **linearize** the scheme ... well, simply replace additions by XORs (only two addition phases per round).
- hopefully, when the round number per iteration is a power of two, very good differential paths exist !
- **mutiblock technique:** don't limitate yourself to only one iteration, but aim for a differential path using several message blocks.
- **the collision attack:** once a differential path found (with success probability P), simply choose $1/P$ random message pairs with the appropriate difference mask.

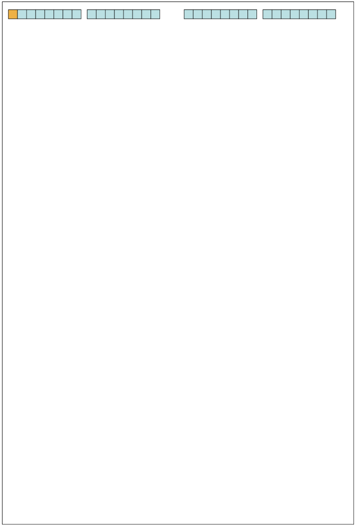
Complexity computation

- **two situations** have to be considered in order to compute the success probability of the differential path in the non-linearized case (both with probability 1/2):
  - **move:** a perturbation at a certain bit position is added to another bit containing no difference.
  - **correction:** a perturbation at a certain bit position is added to another bit containing a difference.

- for the addition of two words $A + B$, the probability of a linear behavior is $\mathrm{HW}(\Delta_A \vee \Delta_B)$.

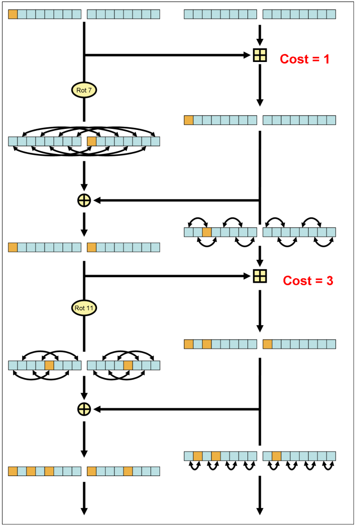
- the probability can be further increased since the carry created at the MSB of $A + B$ can be ignored, i.e. $\mathrm{HW}((\Delta_A \vee \Delta_B) \wedge \texttt{0x7fffffff})$.

Example for CubeHash-2/4



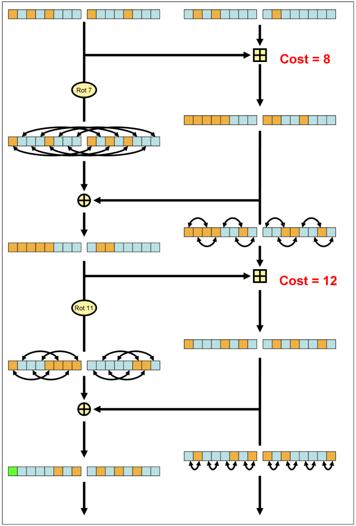
- add a one bit difference on X_0 (at position i).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at positions i+4, i+14, i+22).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at position i+4).

Example for CubeHash-2/4



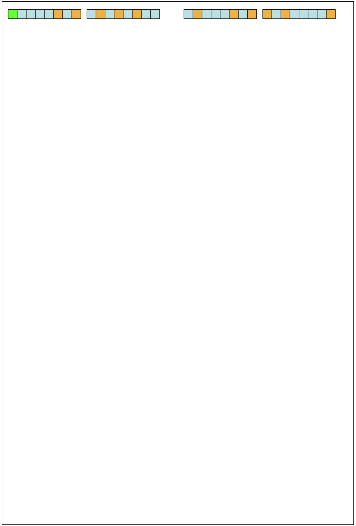
- add a one bit difference on X_0 (at position i).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at positions i+4, i+14, i+22).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at position i+4).

Example for CubeHash-2/4



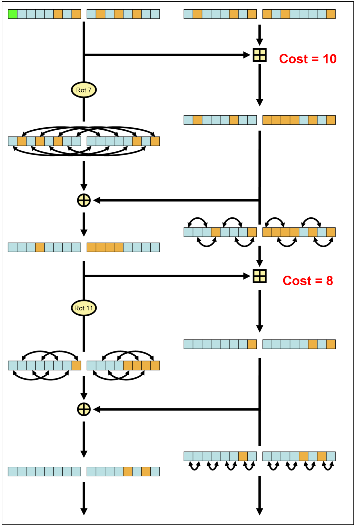
- add a one bit difference on X_0 (at position i).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at positions i+4, i+14, i+22).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at position i+4).

Example for CubeHash-2/4



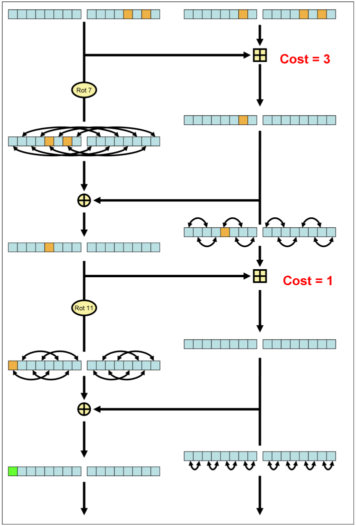
- add a one bit difference on X_0 (at position i).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at positions i+4, i+14, i+22).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at position i+4).

Example for CubeHash-2/4



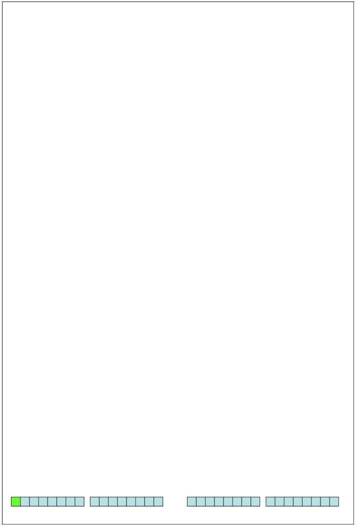
- add a one bit difference on X_0 (at position i).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at positions i+4, i+14, i+22).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at position i+4).

Example for CubeHash-2/4



- add a one bit difference on X_0 (at position i).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at positions i+4, i+14, i+22).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at position i+4).

Example for CubeHash-2/4



- add a one bit difference on X_0 (at position i).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at positions i+4, i+14, i+22).
- do one iteration (2 rounds).
- erase all the differences in X_0 (at position i+4).

Outline

CubeHash collision attack results

<table>
<tr><th>r</th><th>b</th><th>max nb. it.</th><th>probability</th></tr>
<tr><td rowspan="7">1</td><td>64</td><td>3</td><td>2^{3}</td></tr>
<tr><td>32</td><td>5</td><td rowspan="4">2^{32}</td></tr>
<tr><td>16</td><td>5</td></tr>
<tr><td>8</td><td>5</td></tr>
<tr><td>4</td><td>5</td></tr>
<tr><td>2</td><td>7</td><td>2^{221}</td></tr>
<tr><td>1</td><td>15</td><td>2^{1225}</td></tr>
<tr><td rowspan="7">2</td><td>64</td><td>3</td><td rowspan="5">2^{32}</td></tr>
<tr><td>32</td><td>3</td></tr>
<tr><td>16</td><td>3</td></tr>
<tr><td>8</td><td>3</td></tr>
<tr><td>4</td><td>3</td></tr>
<tr><td>2</td><td>4</td><td>2^{221}</td></tr>
<tr><td>1</td><td>8</td><td>2^{1225}</td></tr>
</table>

<table>
<tr><th>r</th><th>b</th><th>max nb. it.</th><th>probability</th></tr>
<tr><td rowspan="7">4</td><td>64</td><td>3</td><td rowspan="5">2^{189}</td></tr>
<tr><td>32</td><td>3</td></tr>
<tr><td>16</td><td>3</td></tr>
<tr><td>8</td><td>3</td></tr>
<tr><td>4</td><td>3</td></tr>
<tr><td>2</td><td>4</td><td>2^{964}</td></tr>
<tr><td>1</td><td>9</td><td>2^{2614}</td></tr>
<tr><td rowspan="7">8</td><td>64</td><td>3</td><td>2^{650}</td></tr>
<tr><td>32</td><td>3</td><td rowspan="2">2^{830}</td></tr>
<tr><td>16</td><td>3</td></tr>
<tr><td>8</td><td>3</td><td rowspan="2">2^{1009}</td></tr>
<tr><td>4</td><td>3</td></tr>
<tr><td>2</td><td>5</td><td rowspan="2">2^{2614}</td></tr>
<tr><td>1</td><td>5</td></tr>
</table>

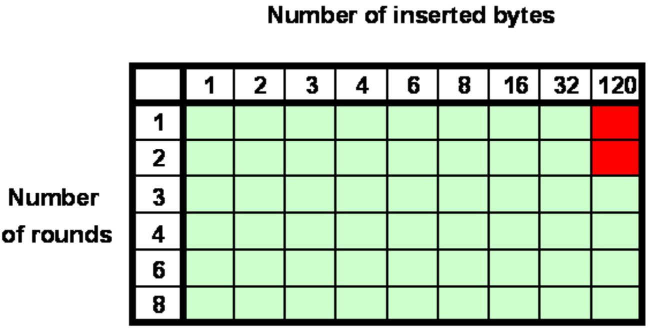
Parameter map

Number of inserted bytes

Number of rounds

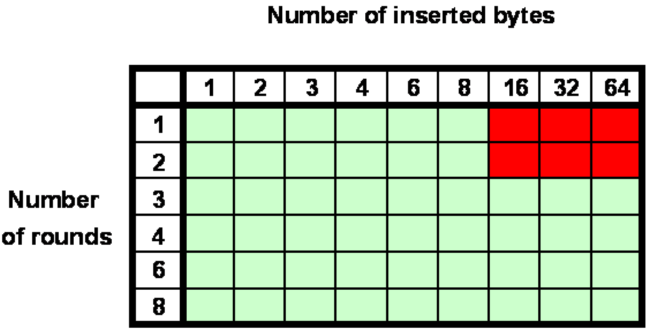
	1	2	3	4	6	8	16	32	120
1									
2									
3									
4									
6									
8									

Parameter map



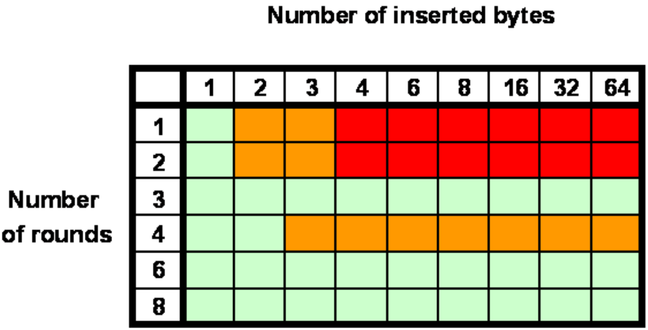
- Aumasson (NIST Hash Forum, 2008)

Parameter map



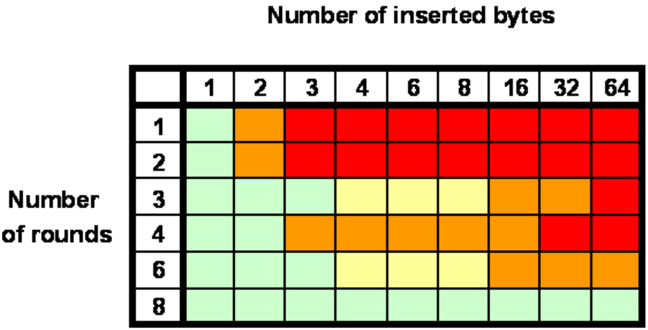
- Dai (NIST Hash Forum, 2008)

Parameter map



- Brier and Peyrin (ACNS 2009)

New results?



- Brier, Khazaei, Peyrin and Meier (yet unpublished)