A Hash Function Family *Luffa*

@The First SHA3 Candidate Conference
25th February 2009

**Dai Watanabe**  
Hisayoshi Sato

**Christophe De Cannière**

Systems Development Laboratory,  
Hitachi, Ltd.

ESAT-COSIC,  
Katholieke Universiteit Leuven
Outline -

- Specification
  - Chaining -
  - Non-linear components
- Security status
  - Generic attack
  - Differential based attack
- Implementations
  - Software
  - Hardware
Introduction to *Luffa* (spec.)
Cryptographic sponge function

- Newer coming construction of a hash function from a random permutation
- It is indifferentiable from a RO

A Hash Function Family *Luffa*
C. De Canniére, H. Sato, D. Watanabe
**Chaining of Luffa**

- **Luffa** is a variant of sponge
  - But, fixed length permutations for all hash length -
    - The number of Qj increases if the hash length gets long (w=3, 4, 5 for hash_len=256, 384, 512)
  - Insert message and mix the state by the linear map MI
  - A blank round
  - The hash value is the sum of the outputs of Qj
Non-linear permutation Q -

- **Input/Output**
  - 256 bits
    (8 32-bit words)

- **Functions**
  - **tweak** -
    - Applied before step functions -
  - **Step functions** -
    - 8 steps

A Hash Function Family *Luffa*
C. De Cannière, H. Sato, D. Watanabe
Step function

4-bit Sbox (bit slice)  Sbox

Feistel ladder of 4 rounds

Constant addition
(1-bit / Sbox)

A Hash Function Family \textit{Luffa}
C. De Canniére, H. Sato, D. Watanabe
Security status
Summary of security status -

- **Sponge function features**
  - Not based on CR compression function
  - Finding inner collision is the best attack

- **Current security status of **Luffa**
  - No security proof for the chaining (yet)
  - Several generic attacks concerned, none of them are serious
  - Differential based attack
    - Seems secure under a reasonable assumption
Long message attack -

- **Sponge’s case**
  - Finding a message \( s.t. \) \( S^{(i)} = S^{(N)} \)
  - Prob. of the event
    - capacity: \( c = \text{len}(S) - \text{len}(M) \)
    - prob = \( 2^{-c/2} \)
  - Complexity -
    - Queries to the permutation: \( 2^{c/2} \)
    - Num. of nodes: \( 2^{c/2} \)
Long message attack (conti.)

- **Luffa’s case**
  - $1/w$ of input bits to each $Q_j$ is controllable by message injection

- **Complexity**
  - **Queries to $Q_j$**
    - $2^{(w-1)/w*256}$
  - **Num. of nodes**
    - $2^{(w-1)/2*256}$
  - **Calc. Complexity**
    - **MA**: $2^{(w-1)/2*256}$
    - **MI calls**: $2^{(w-1)/2*256}$
Differential characteristics of Qj:

- **4 steps (half-block)**
  - Approach: exhaustive truncated path search
  - Possible min. num. of active Sbox: 31
  - MDCP $\leq 2^{-62}$

- **8 steps (full)**
  - Approach: Leon’s algorithm to find the lowest code word
  - Min. active Sbox = 112
  - DCP = $2^{-224} (> 2^{-256})$
  - Not useful to find an inner collision
Differential based attack scenario

- (Seems) the best scenario
  - 2 rounds attack to find an inner collision
- Limitation of modification technique
  - Assumption
    - 1 bit modification doubles the diff. prob.
  - Message block $M^{(i)}$
    - Any, up to 256 bits
  - State $H^{(i)}$
    - Assumed random, up to $(w-1)/2 \times 256$ bits
- (Our) conclusion
  - *Luffa* is secure against this attack if $\text{MDCP}(Q_j) < 2^{-171}$
Implementation aspects -
Software implementations -

<table>
<thead>
<tr>
<th>hash length</th>
<th>ANSI C (cycle/byte)</th>
<th>assembly with SSE2 (cycle/byte)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>32-bit</td>
<td>64-bit</td>
</tr>
<tr>
<td>224</td>
<td>33.9</td>
<td>32.0</td>
</tr>
<tr>
<td></td>
<td>32-bit</td>
<td>64-bit</td>
</tr>
<tr>
<td>256</td>
<td>33.4</td>
<td>32.0</td>
</tr>
<tr>
<td></td>
<td>32-bit</td>
<td>64-bit</td>
</tr>
<tr>
<td>384</td>
<td>45.2</td>
<td>39.0</td>
</tr>
<tr>
<td></td>
<td>32-bit</td>
<td>64-bit</td>
</tr>
<tr>
<td>512</td>
<td>59.7</td>
<td>50.3</td>
</tr>
</tbody>
</table>

Evaluation environment

- **CPU:** Intel Core2Duo E6600 (2.4GHz)
- **Memory:** 2GB
- **ANSI C:** Windows Vista + Visual Studio 2005
- **Assembly:** Ubuntu Linux 8.04 + gas
Hardware implementations (ASIC)

<table>
<thead>
<tr>
<th>Hash length (bit)</th>
<th>Opt.</th>
<th>Gate count (gate)</th>
<th>Frequency (MHz)</th>
<th>Cycles</th>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>256/224 size</td>
<td></td>
<td>10,157</td>
<td>100</td>
<td>891</td>
<td>28.7</td>
</tr>
<tr>
<td>256/224 speed</td>
<td></td>
<td>26,849</td>
<td>444</td>
<td>9</td>
<td>12,642</td>
</tr>
<tr>
<td>384 speed</td>
<td></td>
<td>34,985</td>
<td>444</td>
<td>9</td>
<td>12,642</td>
</tr>
<tr>
<td>512 speed</td>
<td></td>
<td>44,163</td>
<td>444</td>
<td>9</td>
<td>12,642</td>
</tr>
</tbody>
</table>

- Evaluation environment
  - 0.13µm CMOS standard cell library
- Optimization
  - Small gate size: with 1 Sbox and 1 MixWord
  - Speed: 3 step functions in parallel
Thanks you for your attention!
FAQ -

- Q1. What is *Luffa*?
  - A vegetable sponge
  - Scientific name: *Luffa cylindrica* (See picture)

- Q2. Why *Luffa*?
  - Because it is a kind of sponge
  - And very useful (like as hash function)
    - High-quality sponge from dried fruit
    - The young fruit is edible
    - Face lotion from the juice
    - Educational material (in Japan)
    - This is the first trial to use *Luffa* in cryptography

Photo reprented from Wikipedia