Grøstl – a SHA-3 candidate

Krystian Matusiewicz

Wroclaw University of Technology

CECC 2010, June 12, 2010
Talk outline

- Cryptographic hash functions
- NIST SHA-3 Competition
- Grøstl
Cryptographic hash functions
Cryptographic hash functions: why?

- We want to have a short, fixed length “fingerprint” of any piece of data
- Different fingerprints – certainly different data
- Identical fingerprints – most likely the same data
- No one can get any information about the data from the fingerprint
Random Oracle

Construction:

- Box with memory
- On a new query: pick randomly and uniformly the answer, remember it and return the result
- On a repeating query, repeat the answer (function)
Random Oracle

Construction:

- Box with memory
- On a new query: pick randomly and uniformly the answer, remember it and return the result
- On a repeating query, repeat the answer (function)

Properties:

- No information about the data
- To find a preimage: $2^n$ queries for $n$-bit outputs
- To find collisions: $2^{n/2}$ queries
- Random behaviour
Random Oracle

Construction:

- Box with memory
- On a new query: pick randomly and uniformly the answer, remember it and return the result
- On a repeating query, repeat the answer (function)

Properties:

- No information about the data
- To find a preimage: $2^n$ queries for $n$-bit outputs
- To find collisions: $2^{n/2}$ queries
- Random behaviour

We want to construct concrete algorithms which are as close to this behaviour as possible.
NIST SHA-3 Competition
Cryptographic hash functions

NIST SHA-3 Competition

Grøstl

Grøstl – a SHA-3 candidate
The world after Wang (after 2004)

<table>
<thead>
<tr>
<th>hash</th>
<th>designed</th>
<th>Wang-inspired attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD4</td>
<td>1990</td>
<td>collisions by hand calculation</td>
</tr>
<tr>
<td>MD5</td>
<td>1992</td>
<td>collisions in ms on a computer</td>
</tr>
<tr>
<td>SHA-0</td>
<td>1993</td>
<td>collisions in (\approx 1) hr</td>
</tr>
<tr>
<td>SHA-1</td>
<td>1995</td>
<td>attacked, (2^{69})</td>
</tr>
<tr>
<td>SHA-2</td>
<td>2002</td>
<td>no attack</td>
</tr>
</tbody>
</table>

Many other hash functions similar to MD5/SHA-1 were subsequently broken using Wang’s method.
NIST SHA-3 competition

- There is only one hash function standardized by NIST which is not broken (SHA-256/SHA-512)
- But design principles are somehow similar to the broken ones, is it secure enough?
- Would be good to have a backup plan
- Let’s organize a competition like the one for Advanced Encryption Standard!
First-round SHA-3 candidates

<table>
<thead>
<tr>
<th>Abacus</th>
<th>ARIRANG</th>
<th>AURORA</th>
<th>BLAKE</th>
<th>Blender</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue Midnight Wish</td>
<td>BOOLE</td>
<td>Cheetah</td>
<td>CHI</td>
<td>CRUNCH</td>
</tr>
<tr>
<td>CubeHash</td>
<td>DCH</td>
<td>Dynamic SHA</td>
<td>Dynamic SHA2</td>
<td>ECHO</td>
</tr>
<tr>
<td>ECOH</td>
<td>EDON-R</td>
<td>EnRUPT</td>
<td>ESSENCE</td>
<td>FSB</td>
</tr>
<tr>
<td>Fugue</td>
<td>Grøstl</td>
<td>Hamsi</td>
<td>JH</td>
<td>Keccak</td>
</tr>
<tr>
<td>Khichidi-1</td>
<td>LANE</td>
<td>Lesamnta</td>
<td>Luffa</td>
<td>LUX</td>
</tr>
<tr>
<td>MCSSHA-3</td>
<td>MD6</td>
<td>MeshHash</td>
<td>NaSHA</td>
<td>SANDstorm</td>
</tr>
<tr>
<td>Sarmal</td>
<td>Sgail</td>
<td>Shabal</td>
<td>SHAMATA</td>
<td>SHAvite-3</td>
</tr>
<tr>
<td>SIMD</td>
<td>Skein</td>
<td>Spectral Hash</td>
<td>StreamHash</td>
<td>SWIFFTX</td>
</tr>
<tr>
<td>Tangle</td>
<td>TIB3</td>
<td>Twister</td>
<td>Vortex</td>
<td>WaMM</td>
</tr>
<tr>
<td>Waterfall</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Second-round SHA-3 candidates

<table>
<thead>
<tr>
<th>Abacus</th>
<th>ARIRANG</th>
<th>AURORA</th>
<th>BLAKE</th>
<th>Blender</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue Midnight Wish</td>
<td>BOOLE</td>
<td>Cheetah</td>
<td>CHI</td>
<td>CRUNCH</td>
</tr>
<tr>
<td>CubeHash</td>
<td>DCH</td>
<td>Dynamic SHA</td>
<td>Dynamic SHA2</td>
<td>ECHO</td>
</tr>
<tr>
<td>ECOH</td>
<td>EDON-R</td>
<td>EnRUPT</td>
<td>ESSENCE</td>
<td>FSB</td>
</tr>
<tr>
<td>Fugue</td>
<td>Grøstl</td>
<td>Hamsi</td>
<td>JH</td>
<td>Keccak</td>
</tr>
<tr>
<td>Khichidi-1</td>
<td>LANE</td>
<td>Lesamnta</td>
<td>Luffa</td>
<td>LUX</td>
</tr>
<tr>
<td>MCSSSHA-3</td>
<td>MD6</td>
<td>MeshHash</td>
<td>NaSHA</td>
<td>SANDstorm</td>
</tr>
<tr>
<td>Sarmal</td>
<td>Sgail</td>
<td>Shabal</td>
<td>SHAMATA</td>
<td>SHAvite-3</td>
</tr>
<tr>
<td>SIMD</td>
<td>Skein</td>
<td>Spectral Hash</td>
<td>StreamHash</td>
<td>SWIFFTX</td>
</tr>
<tr>
<td>Tangle</td>
<td>TIB3</td>
<td>Twister</td>
<td>Vortex</td>
<td>WaMM</td>
</tr>
<tr>
<td>Waterfall</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What happens next?

- August 2010 – second NIST workshop
- Around 5 finalists will be announced before the end of the year
- The winner selected in 2012
The SHA-3 Zoo / Hash function Zoo

Everything you ever wanted to know about SHA-3 candidates

- ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo
- ehash.iaik.tugraz.at/wiki/The_Hash_Function_Zoo
Grøstl
The Grøstl team

Søren S. Thomsen

Martin Schläffer

Christian Rechberger

Florian Mendel

Lars R. Knudsen

Praveen Gauravaram

& K.M.
Grøstl hash function

- Iterated wide-pipe
- Output transformation
Grøstl Compression Function

- $P$, $Q$ – large, fixed permutations
- Security reductions: collision, preimage resistance provided that permutations behave like ideal ones
Grøstl Output Transformation

- Provides protection against some attacks
- Random behaviour
Designing an “ideal permutation”

- How to design an “ideal permutation”?
- No way of telling it apart from randomly chosen permutation
- No visible structure
- No differential trail with significant probability
- Wide-trail strategy
Grøstl: One round of $P, Q \times 10$
Grøstl: AddRoundConstant

- Permutations must be different (security)
- Different constants for $P$ and $Q$
Grøstl: SubBytes
Grøstl: ShiftBytes

Diagram showing the shift bytes operation in Grøstl.
Grøstl: MixBytes

\[ B = \text{circ}(02, 02, 03, 04, 05, 03, 05, 07) \]
Cryptographic hash functions

NIST SHA-3 Competition

Grøstl

Cryptanalytic Results

<table>
<thead>
<tr>
<th>Hash function collisions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Grøstl-256</td>
<td>4/10</td>
</tr>
<tr>
<td>Grøstl-512</td>
<td>5/14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Compression function collisions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Grøstl-256</td>
<td>7/10</td>
</tr>
<tr>
<td>Grøstl-512</td>
<td>7/14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Permutation non-randomness</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Grøstl-256</td>
<td>8/10</td>
</tr>
</tbody>
</table>
Grøstl: Enjoy!