The Hash Function Hamsi

Özgül Küçük

Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC

February 27, 2009
Outline

Hamsi Specification
   General Design
   Message Expansion
   Hamsi-256

Design Choices and Analysis
   How to Analyze Hamsi?

Implementation
   Software-Hardware

Conclusion
Outline

Hamsi Specification
  General Design
  Message Expansion
  Hamsi-256

Design Choices and Analysis
  How to Analyze Hamsi?

Implementation
  Software-Hardware

Conclusion
General Design

message block

chain value
General Design

message block → expanded message → chain value
General Design

- message block
- expanded message
- chain value
- Concatenation
General Design

- Message block
- Expanded message
- Chain value
- Concatenation
- Truncated non-linear
- Permutation $P$
General Design

- Message block
- Expanded message
- Concatenation
- Truncated non-linear permutation
- Chain value
General Design

- message block
- expanded message
- chain value
- Concatenation
- Truncated non-linear
- Permutation $P$
- chain value
General Design

M₁    M₂    M₃
General Design

\[ M_1 \quad \quad M_2 \quad \quad M_3 \]
The Hash Function Hamsi
General Design

The Hash Function Hamsi

M!

M

h0

Concatenation

P

h1

Concatenation

P

M1

M2

M1

M2

M1


Özgül Küçük

Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC

The Hash Function Hamsi
General Design

\[ h_0 \] Concatenation \[ P \] Concatenation \[ h_1 \] 

\[ M_1 \] 

\[ M_2 \] 

\[ M_i \]
General Design

The Hash Function Hamsi

\[ \begin{align*}
M_1 &\rightarrow h_0 \rightarrow \text{Concatenation} \rightarrow P \\
M_2 &\rightarrow h_1 \rightarrow \text{Concatenation} \rightarrow P \\
\vdots &\rightarrow \vdots \rightarrow \text{Concatenation} \rightarrow P \\
M_l &\rightarrow h_{l-1} \rightarrow \text{Concatenation} \rightarrow P_f
\end{align*} \]
The Hash Function Hamsi
General Design

- General design is a sponge-like construction and influenced by Grindahl
  - Message expansion
  - Feedforward of the previous chaining value
  - 2 different non-linear permutations; $P$, $P_f$
General Design

- General design is a sponge-like construction and influenced by Grindahl
  - Message expansion
  - Feedforward of the previous chaining value
  - 2 different non-linear permutations; $P$, $P_f$
- Hamsi iteration operates over small message blocks
  - 32-bit → Hamsi-256
  - 64-bit → Hamsi-512
  - Hamsi-256 → 512-bit internal state size
  - Hamsi-512 → 1024-bit internal state size
Outline

Hamsi Specification
  General Design
  Message Expansion
  Hamsi-256

Design Choices and Analysis
  How to Analyze Hamsi?

Implementation
  Software-Hardware

Conclusion
Message Expansion

Hamsi-256

Hamsi-512
Message Expansion

Hamsi-256

32-bit

Hamsi-512
Message Expansion

Hamsi-256
Linear code $[128,16,70]$ over $F_4$
32-bit

Hamsi-512
Message Expansion

Hamsi-256

Linear code $[128,16,70]$ over $F_4$

32-bit

256-bit

Hamsi-512
Message Expansion

Hamsi-256

32-bit

Linear code $[128,16,70]$ over $F_4$

256-bit

Hamsi-512

64-bit
Message Expansion

Hamsi-256

32-bit

256-bit

Linear code $[128, 16, 70]$ over $F_4$

Hamsi-512

64-bit

512-bit
Message Expansion

Hamsi-256

32-bit

256-bit

Linear code \([128,16,70]\) over \(F_4\)

Hamsi-512

64-bit

512-bit

Linear code \([256,32,131]\) over \(F_4\)
Message Expansion
Message Expansion

- **Best Known Linear Codes;** optimal choice for given parameters
- Minimum distance → active Sboxes for one round
  - 70 for Hamsi-256
  - 131 for Hamsi-512
- Provides **strong diffusion** against differential type attacks
Message Expansion

- **Best Known Linear Codes**; optimal choice for given parameters
- Minimum distance → active Sboxes for one round
  - 70 for Hamsi-256
  - 131 for Hamsi-512
- Provides **strong diffusion** against differential type attacks
- Can be computed in parallel
- Can be implemented efficiently in software and hardware
Outline

Hamsi Specification
   General Design
   Message Expansion
Hamsi-256

Design Choices and Analysis
   How to Analyze Hamsi?

Implementation
   Software-Hardware

Conclusion
Permutations $P$ and $P_f$
Permutations $P$ and $P_f$

- XOR of constants and a counter
- Substitution is by $4 \times 4$-bit Sboxes
- Diffusion is by a linear transformation $L$
Permutations $P$ and $P_f$

- XOR of constants and a counter
- Substitution is by $4 \times 4$-bit Sboxes
- Diffusion is by a linear transformation $L$
- Sbox and $L$ are from the block cipher Serpent
- $P$ and $P_f$ only differ in the number of rounds and constants
  - $P$ has 3 rounds, $P_f$ has 6 rounds
  - $P_f$ is only applied at the finalization
Concatenation

\[(m_0, m_1, \ldots, m_7, c_0, c_1, \ldots, c_7)\]
Concatenation

\((m_0, m_1, \ldots, m_7, c_0, c_1, \ldots, c_7)\) → \(C\)

Hamsi-256 state

\[
\begin{array}{cccc}
  m_0 & m_1 & c_0 & c_1 \\
  c_2 & c_3 & m_2 & m_3 \\
  m_4 & m_5 & c_4 & c_5 \\
  c_6 & c_7 & m_6 & m_7 \\
\end{array}
\]
Each Sbox inputs 2-bits from the chain value and the expanded message.

- Chaining value and expanded message words appear in all bit positions of the Sboxes.
Substitution

Hamsi-256 state

\[
\begin{array}{cccc}
S_0 & S_1 & S_2 & S_3 \\
S_4 & S_5 & S_6 & S_7 \\
S_8 & S_9 & S_{10} & S_{11} \\
S_{12} & S_{13} & S_{14} & S_{15}
\end{array}
\]
Substitution

Hamsi-256 state

\[
\begin{array}{cccc}
S_0 & S_1 & S_2 & S_3 \\
S_4 & S_5 & S_6 & S_7 \\
S_8 & S_9 & S_{10} & S_{11} \\
S_{12} & S_{13} & S_{14} & S_{15}
\end{array}
\]

\[S\]
### Substitution

**Hamsi-256 state**

<table>
<thead>
<tr>
<th>S0</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
</tr>
</thead>
<tbody>
<tr>
<td>S4</td>
<td>S5</td>
<td>S6</td>
<td>S7</td>
</tr>
<tr>
<td>S8</td>
<td>S9</td>
<td>S10</td>
<td>S11</td>
</tr>
<tr>
<td>S12</td>
<td>S13</td>
<td>S14</td>
<td>S15</td>
</tr>
</tbody>
</table>

- **4 x 4 Serpent Sbox**
- Confusion/diffusion across the rows
- Can be processed in parallel, word size up to 128-bits
- Suitable for bitsliced implementation
Diffusion

Hamsi-256 state

\[
\begin{array}{cccc}
A_0 & B_0 & C_0 & D_0 \\
D_1 & A_1 & B_1 & C_1 \\
C_2 & D_2 & A_2 & B_2 \\
B_3 & C_3 & D_3 & A_3 \\
\end{array}
\]

\[L\]
**Diffusion**

Hamsi-256 state

<table>
<thead>
<tr>
<th>A₀</th>
<th>B₀</th>
<th>C₀</th>
<th>D₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>D₁</td>
<td>A₁</td>
<td>B₁</td>
<td>C₁</td>
</tr>
<tr>
<td>C₂</td>
<td>D₂</td>
<td>A₂</td>
<td>B₂</td>
</tr>
<tr>
<td>B₃</td>
<td>C₃</td>
<td>D₃</td>
<td>A₃</td>
</tr>
</tbody>
</table>

Diffusion by \( L \)
Diffusion

Hamsi-256 state

<table>
<thead>
<tr>
<th></th>
<th>A₀</th>
<th>B₀</th>
<th>C₀</th>
<th>D₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>D₁</td>
<td>A₁</td>
<td>B₁</td>
<td>C₁</td>
<td></td>
</tr>
<tr>
<td>C₂</td>
<td>D₂</td>
<td>A₂</td>
<td>B₂</td>
<td></td>
</tr>
<tr>
<td>B₃</td>
<td>C₃</td>
<td>D₃</td>
<td>A₃</td>
<td></td>
</tr>
</tbody>
</table>

Diffusion by $L$

\[ L \]
### Diffusion

<table>
<thead>
<tr>
<th>Hamsi-256 state</th>
<th>Diffusion by $L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_0$</td>
<td>$B_0$</td>
</tr>
<tr>
<td>$D_1$</td>
<td>$A_1$</td>
</tr>
<tr>
<td>$C_2$</td>
<td>$D_2$</td>
</tr>
<tr>
<td>$B_3$</td>
<td>$C_3$</td>
</tr>
</tbody>
</table>
Diffusion

- \( L \) consists of XOR, rotation and shift operations
- \( L \) operates on 32-bit words; inputs and outputs 4 words
- Any single bit difference amplified by \( L \) affects a different Sbox
Truncation
Truncation

Hamsi-256 state $\rightarrow$ After truncation $T$
Design Choices and Analysis

Özgül Küçük

Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC

The Hash Function Hamsi
Design Choices and Analysis

- Design choices are due to preliminary analysis
Design Choices and Analysis

- Design choices are due to preliminary analysis
  - Strong message expansion
  - Concatenation can be viewed as an initial mixing
Design Choices and Analysis

- Design choices are due to preliminary analysis
  - Strong message expansion
  - Concatenation can be viewed as an initial mixing
  - Sboxes have good differential/linear and avalanche properties
    - Algebraic degree of all but one output bit is 3; we are not aware of any attacks based on this.
Design Choices and Analysis

- Design choices are due to preliminary analysis
  - Strong message expansion
  - Concatenation can be viewed as an initial mixing
  - Sboxes have good differential/linear and avalanche properties
    - Algebraic degree of all but one output bit is 3; we are not aware of any attacks based on this.
  - Feedforward is added against generic preimage attacks
  - $L$ is applied in such a way to destroy symmetry
  - $P_f$ is to prevent against length extension and slide attacks
Outline

Hamsi Specification
  General Design
  Message Expansion
  Hamsi-256

Design Choices and Analysis
  How to Analyze Hamsi?

Implementation
  Software-Hardware

Conclusion
How to break Hamsi?
How to Analyze Hamsi?

How to break Hamsi?

- Internal collisions
How to Analyze Hamsi?

How to break Hamsi?

- **Internal collisions**
  - Any difference on the message very quickly activates many Sboxes (thanks to message expansion...)
  - Message modification can not help to cancel difference propagation (message block is concatenated to the chain value)
How to Analyze Hamsi?

How to break Hamsi?

- Internal collisions
  - Any difference on the message very quickly activates many Sboxes (thanks to message expansion...)
  - Message modification can not help to cancel difference propagation (message block is concatenated to the chain value)
- External collision seems much harder to find... ($P_f$ has 6 rounds)
How to Analyze Hamsi?

How to break Hamsi?

▶ Analyze the following:
How to break Hamsi?

► Analyze the following:
  ► Apply a difference only on the chain value
  ► Propagate as far as you find a collision
  ► Work backwards to find a matching difference on the message
  ► Show that overall complexity is lower than the birthday bound
How to break Hamsi?

- Analyze the following:
  - Apply a difference only on the chain value
  - Propagate as far as you find a collision
  - Work backwards to find a matching difference on the message
  - Show that overall complexity is lower than the birthday bound

- break 1 round of Hamsi-256; appreciated! (easy...?)
How to Analyze Hamsi?

◮ Analyze the following:
  ◮ Apply a difference only on the chain value
  ◮ Propagate as far as you find a collision
  ◮ Work backwards to find a matching difference on the message
  ◮ Show that overall complexity is lower than the birthday bound

◮ break 1 round of Hamsi-256; appreciated! (easy...?)
◮ break 2 rounds of Hamsi-256; very much appreciated!
How to break Hamsi?

- Analyze the following:
  - Apply a difference only on the chain value
  - Propagate as far as you find a collision
  - Work backwards to find a matching difference on the message
  - Show that overall complexity is lower than the birthday bound

- break 1 round of Hamsi-256; appreciated! (easy...?)
- break 2 rounds of Hamsi-256; very much appreciated!
- break 3 rounds of Hamsi-256; (well... will be good to know...)

Özgül Küçük

Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC
How to break Hamsi?

- Analyze the following:
  - Apply a difference only on the chain value
  - Propagate as far as you find a collision
  - Work backwards to find a matching difference on the message
  - Show that overall complexity is lower than the birthday bound

- break 1 round of Hamsi-256; appreciated! (easy...?)
- break 2 rounds of Hamsi-256; very much appreciated!
- break 3 rounds of Hamsi-256; (well... will be good to know...)

- Use your imagination!
Outline

Hamsi Specification
   General Design
   Message Expansion
   Hamsi-256

Design Choices and Analysis
   How to Analyze Hamsi?

Implementation
   Software-Hardware

Conclusion
Software-Hardware

- Hamsi is a software and hardware friendly algorithm

- Hamsi-256 runs at 25cpb on Intel Core2 (in C using SSE2 instructions)
  - Further improvements are in progress

- Hardware results for Hamsi-256:
  - Area: 14.5KGate, UMC 130 nm library, typical case
  - Timing: 3.2ns as critical path delay, 310MHz
  - Throughput: 3.3Gbps
Conclusion

- Hamsi is a secure and fast hash function
- Feel free to analyze it
- Feel free to make it faster

http://homes.esat.kuleuven.be/~okucuk/hamsi/