Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers

Nicky Mouha\textsuperscript{1}, Bart Mennink\textsuperscript{1}, Anthony Van Herrewege\textsuperscript{1}, Dai Watanabe\textsuperscript{2}, Bart Preneel\textsuperscript{1}, Ingrid Verbauwhede\textsuperscript{1}

\textsuperscript{1}ESAT/COSIC, KU Leuven and iMinds, Belgium  
\textsuperscript{2}Yokohama Research Laboratory, Hitachi, Japan

SAC 2014 — August 15, 2014
MAC Algorithm for Microcontrollers

Message Authentication Code (MAC)

- $\text{MAC}_K(m) = \tau$
- Authenticity, no confidentiality
- Same key for MAC generation and verification
MAC Algorithm for Microcontrollers

**Message Authentication Code (MAC)**

- \( \text{MAC}_K(m) = \tau \)
- Authenticity, no confidentiality
- Same key for MAC generation and verification

**Microcontroller**

- Cheap 8/16/32-bit processor: USD 25-50¢
- Applications: home, medical, industrial, ...
- Ubiquitous: 30-100 in any recent car
Design

Requirements

- Drop-in replacement for AES-CMAC (variant of CBC-MAC for variable-length messages)
- Same functionality and security
Design

Requirements

- Drop-in replacement for AES-CMAC (variant of CBC-MAC for variable-length messages)
- Same functionality and security

Speed

- “Ten times faster than AES”
Design

Requirements

- Drop-in replacement for AES-CMAC (variant of CBC-MAC for variable-length messages)
- Same functionality and security

Speed

- “Ten times faster than AES”

Approach

- Dedicated design for microcontrollers
Commonly used MACs

Based on (cryptographic) hash function

- **Example**: HMAC, SHA3-MAC
- Large block size, collision resistance unnecessary
Commonly used MACs

Based on (cryptographic) hash function

- **Example**: HMAC, SHA3-MAC
- Large block size, collision resistance unnecessary

Based on universal hashing

- **Examples**: UMAC, GMAC, Poly1305
- **Requires**: nonce, constant-time multiply, long tags
Commonly used MACs

Based on (cryptographic) hash function

- Example: HMAC, SHA3-MAC
- Large block size, collision resistance unnecessary

Based on universal hashing

- Examples: UMAC, GMAC, Poly1305
- Requires: nonce, constant-time multiply, long tags

Based on block cipher

- Example: CMAC
Commonly used MACs

Based on (cryptographic) hash function
- **Example**: HMAC, SHA3-MAC
- Large block size, collision resistance unnecessary

Based on universal hashing
- **Examples**: UMAC, GMAC, Poly1305
- **Requires**: nonce, constant-time multiply, long tags

Based on block cipher
- **Example**: CMAC
- **Problem**: ten times too slow!
Our Approach

Every cycle counts!

- Avoid load/store: keep data in registers
- Avoid bit masking
- Make optimal use of instruction set
Our Approach

**Every cycle counts!**
- Avoid load/store: keep data in registers
- Avoid bit masking
- Make optimal use of instruction set

**Bridging the gap**
- Cryptanalysis
- Implementation
Primitive

Which primitive?
- Cryptographic hash function \( X \)
Primitive

Which primitive?

- Cryptographic hash function ✗
- Universal hash function ✗
Primitive

Which primitive?

- Cryptographic hash function
- Universal hash function
- Block cipher
Primitive

Which primitive?

- Cryptographic hash function
- Universal hash function
- Block cipher
- Ideal permutation
Which primitive?

- Cryptographic hash function ✗
- Universal hash function ✗
- Block cipher ✗
- Ideal permutation ✗

\[
\begin{align*}
K & \downarrow \\
\odot & \pi \\
C & \downarrow \quad K
\end{align*}
\]

\[ P \rightarrow \text{Even-Mansour Block Cipher} \rightarrow C \]
Primitive

**Which primitive?**

- Cryptographic hash function $\times$
- Universal hash function $\times$
- Block cipher $\times$
- Ideal permutation $\times$

$\xrightarrow{\text{Even-Mansour Block Cipher}}$

**Related-key attacks**

- Insecure: choose uniformly random keys!
Chaskey: Mode of Operation

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$
- $K_1 = 2K$

![Diagram](image-url)
Chaskey: Mode of Operation

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$
Chaskey: Mode of Operation: Phantom XORs

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$
Chaskey: Mode of Operation: Phantom XORs

- Split \( m \) into \( \ell \) blocks of \( n \) bits
- Top: \( |m_\ell| = n \), bottom: \( 0 \leq |m_\ell| < n \)
- \( K_1 = 2K, K_2 = 4K \)
Chaskey: Mode of Operation: Block-cipher-based

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$
Chaskey: Mode of Operation: Block-cipher-based

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$

variant of FCBC [BR’00]
Chaskey: Mode of Operation: Compared to CMAC

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$

variant of CMAC [IK’03]
Chaskey: Mode of Operation: Compared to CMAC

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$

$E_K(0^n) \rightarrow K$

variant of CMAC [IK’03]
Chaskey: Mode of Operation: Compared to CMAC

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$

$\begin{array}{c} E_K(0^n) \rightarrow K \\ E_K\|K \rightarrow E_K\|K \rightarrow \cdots \rightarrow E_K\|K \rightarrow E_K\|K \rightarrow K \oplus K_1 \rightarrow \tau \end{array}$

variant of CMAC [IK’03]

$\begin{array}{c} E_K\|K \rightarrow E_K\|K \rightarrow \cdots \rightarrow E_K\|K \rightarrow E_K\|K \rightarrow K \oplus K_2 \rightarrow \tau \end{array}$

Even-Mansour
Chaskey: Mode of Operation: Compared to CMAC

- Split $m$ into $\ell$ blocks of $n$ bits
- Top: $|m_\ell| = n$, bottom: $0 \leq |m_\ell| < n$
- $K_1 = 2K$, $K_2 = 4K$

\[ \begin{array}{c}
1 \quad E_K(0^n) \rightarrow K \\
\end{array} \]

- Even-Mansour

variant of CMAC [IK’03]

\[ \begin{array}{c}
3 \quad \text{not in CMAC}
\end{array} \]
Cryptanalysis

**MAC forgery**: find new valid \((m, \tau)\)
- \(D\): data complexity (\# chosen plaintexts)
- \(T\): time complexity (\# permutation eval.)

**Attacks**
- Internal collision: \(D \approx 2^{n/2}\)
- Key recovery: \(T \approx 2^n/D\)
- Tag guessing: \(\approx 2^t\) guesses

**Chaskey parameters**
- Key size, block size: \(n = 128\), tag length: \(t \geq 64\)
Permutation

Design
- Add-Rot-XOR (ARX)
- Inspired by SipHash
- 32-bit words
- 8 rounds

Properties
- Rotations by 8, 16: faster on 8-bit μC
- Fixed point: $0 \rightarrow 0$
- Cryptanalysis: rotational, (truncated) differential, MitM, slide,... see paper!
# Chaskey: Speed Optimized (gcc -O2)

<table>
<thead>
<tr>
<th>Microcontroller</th>
<th>Algorithm</th>
<th>Data [byte]</th>
<th>ROM [byte]</th>
<th>Speed [cycles/byte]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cortex-M0</td>
<td>AES-128-CMAC</td>
<td>16</td>
<td>13492</td>
<td>173.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>13492</td>
<td>136.5</td>
</tr>
<tr>
<td></td>
<td>Chaskey</td>
<td>16</td>
<td>1308</td>
<td>21.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>1308</td>
<td>18.3</td>
</tr>
<tr>
<td>Cortex-M4</td>
<td>AES-128-CMAC</td>
<td>16</td>
<td>28524</td>
<td>118.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>28524</td>
<td>105.0</td>
</tr>
<tr>
<td></td>
<td>Chaskey</td>
<td>16</td>
<td>908</td>
<td>10.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>908</td>
<td>7.0</td>
</tr>
</tbody>
</table>
## Chaskey: Size Optimized (gcc -Os)

<table>
<thead>
<tr>
<th>Microcontroller</th>
<th>Algorithm</th>
<th>Data [byte]</th>
<th>ROM [byte]</th>
<th>Speed [cycles/byte]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cortex-M0</td>
<td>AES-128-CMAC</td>
<td>16</td>
<td>11664</td>
<td>176.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>11664</td>
<td>140.0</td>
</tr>
<tr>
<td></td>
<td>Chaskey</td>
<td>16</td>
<td>414</td>
<td>21.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>414</td>
<td>16.9</td>
</tr>
<tr>
<td>Cortex-M4</td>
<td>AES-128-CMAC</td>
<td>16</td>
<td>10925</td>
<td>127.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>10925</td>
<td>89.4</td>
</tr>
<tr>
<td></td>
<td>Chaskey</td>
<td>16</td>
<td>402</td>
<td>16.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>402</td>
<td>11.2</td>
</tr>
</tbody>
</table>
Summary

Chaskey:
MAC algorithm for 32-bit microcontrollers

- Addition-Rotation-XOR (ARX)
- Even-Mansour block cipher
- ARM Cortex-M: 7-15× faster than AES-128-CMAC
Questions?