Side-Channel Countermeasure for SHA-3 At Almost-Zero Area Overhead

Mostafa Taha and Patrick Schaumont
Virginia Tech

This research was supported in part by the VT-MENA program of Egypt, and by NSF grant no. 1115839.
Outline

- Side-Channel Analysis
- SHA-3, the story and its applications
- SHA-3 Countermeasure
- New Design Concept
Outline

• Side-Channel Analysis
• SHA-3, the story and its applications
• SHA-3 Countermeasure
• New Design Concept
Side-Channel Analysis

• Information Leakage
Side-Channel Analysis

• Example
  – Password Checker (8 bytes)

\[(I[0] = K[0])\] is False \(\Rightarrow\) one loop

\[(I[0] = K[0])\] is True
\[(I[1] = K[1])\] is False \(\Rightarrow\) two loops

255 values of \(I[0]\) need one loop, while one correct value needs two loops

\[
\text{for } (i = 0 ; i < 8 ; i++)\{
  \text{if}(I[i] = K[i])\{
    \text{Password} = \text{true};
  \}
  \text{else}\{
    \text{Password} = \text{false};
    \text{break};
  \}
\}
\]

Brute Force security reduced from \(256^8\) to \(256*8\)
If 1 try = 1 \(\mu\)sec \(\Rightarrow\) from 585K years to 2 msec!
Side-Channel Analysis

- Information Leakage
  - Computation Time
  - Power Consumption
  - Electromagnetic
  - Acoustic Waves
  - Photonic Emission
  - Faulty Ciphertext
Side-Channel Analysis

• One Trace ➔ Simple .......... Analysis
• Many Traces ➔ Differential .......... Analysis

<table>
<thead>
<tr>
<th>Actual Leakage</th>
<th>Hypothesis Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Eq. Leakage</th>
<th>Variable</th>
<th>Eq. Leakage</th>
<th>Variable</th>
<th>Eq. Leakage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0F</td>
<td>4</td>
<td>0x82</td>
<td>2</td>
<td>0xF1</td>
<td>5</td>
</tr>
<tr>
<td>0xAA</td>
<td>4</td>
<td>0x51</td>
<td>3</td>
<td>0x4E</td>
<td>4</td>
</tr>
<tr>
<td>0xD3</td>
<td>5</td>
<td>0xA3</td>
<td>4</td>
<td>0x0B</td>
<td>3</td>
</tr>
<tr>
<td>0x31</td>
<td>3</td>
<td>0xC7</td>
<td>5</td>
<td>0x92</td>
<td>3</td>
</tr>
</tbody>
</table>

Correlation
Side-Channel Analysis

Differential Analysis

I - Var. Affect Traces
II - Predict Variables
III - Combine Traces

Hiding
Masking
Leakage Resiliency
Outline

- Side-Channel Analysis
- SHA-3, the story and its applications
- SHA-3 Countermeasure
- Why the countermeasure works!
SHA-3 (The Story)

• SHA-3 is a hashing standard

• SHA-0, SHA-1, SHA-2 → SHA-3

• Competition started in Nov. 2007 and ended in Oct. 2012 with Keccak as the winner.

• Chosen for
  – Superior performance in hardware
  – The Sponge construction
SHA-3 (The *Sponge* construction)
SHA-3 (Applications)

- Regular Hashing
SHA-3 (Applications)

- Salted Hashing

\[
\begin{align*}
SALT & \quad M_0 \quad M_1 \quad M_2 \\
0 & \quad \oplus \quad f \quad f \quad f \\
0 & \quad \oplus \quad \oplus \quad \oplus \\
\end{align*}
\]

\[
\begin{align*}
& \quad r \\
& \quad c \\
& \quad MAC_0 \quad MAC_1
\end{align*}
\]
SHA-3 (Applications)

- Message Authentication Codes

- Hashing, salted hashing and MACs are ordinary applications. But!
SHA-3 (Applications)

- Random Number Generation New!
SHA-3 (Applications)

- Stream Encryption **New!**
SHA-3 (Applications)

• Authenticated Encryption New!

And many more.
SHA-3

- Keccak permutation function

Also, PHOTON, QUARK, SPONGENT,…
SHA-3 (Single Core)

- Regular Hashing
- Salted Hashing
- RNG
- MAC-generation
- Stream Encryption
- Authenticated Encryption

One HW Core
SHA-3 (Single Core)

- Regular Hashing
- Salted Hashing
- RNG

- MAC-generation
- Stream Encryption
- Authenticated Encryption

- Side-Channel Protection
- 3x implementation cost
SHA-3 (Single Core)

• Permutation Functions

AES Block-cipher

MAC-Keccak
SHA-3 (Single Core)

- Regular Hashing
- Salted Hashing
- RNG
  - MAC-generation (m-1) + 1
  - Stream Encryption (m-1) + 1
  - Authenticated Encryption (m-1) + 1

Design a new countermeasure
- Minimal changes to the implementation
- Can be turned ON / OFF New!
Outline

- Side-Channel Analysis
- SHA-3, the story and its applications
- SHA-3 Countermeasure
- Why the countermeasure works!
SHA-3 Countermeasure

1. The Key goes to a separate input.
2. Mandate the use of IV in all keyed applications.
3. Squeeze the rate to “one bit” during IV.
4. Use Round-Reduced version of Keccak during IV, except the last bit.
5. Then, proceed normally.
SHA-3 Countermeasure, **WHY?**

1. The Key goes to a separate input.
2. Mandate the use of IV in all keyed applications.
3. Squeeze the rate to “one bit” during IV.
4. Use Round-Reduced version of Keccak during IV, except the last bit.
5. Then, proceed normally.

**Tree Structure**

```
Key State
  K0         IV0
  K1
  K00
  K01
  K10
  K11
```

**Diagram**

```
Key | IV0 | IV1 | IV|IV|-2 | IV|IV|-1 | M0 | M1
---|-----|-----|---|-----|-----|-----|----|----
r  0  |     |     |   |     |     |     |    |    
c  0  |     |     |   |     |     |     |    |    
```

**Mathematical Formula**

```
rate = \frac{r}{n}
```
SHA-3 Countermeasure, \textbf{WHY?}

1. The Key goes to a separate input. \textit{Increase size of the secret}
2. Mandate the use of IV in all keyed applications. \textit{Nonce}
3. Squeeze the rate to “one bit” during IV.
4. Use Round-Reduced version of Keccak during IV, except the last bit.
5. Then, proceed normally.

| Key | IV₀ | IV₁ | IV\_{|IV|\!\!-\!2} | IV\_{|IV|\!\!-\!1} | M₀ | M₁ |
|-----|-----|-----|----------------|----------------|----|----|
| r   |     |     |                |                |    |    |
| c   | 0   |     |                |                |    |    |

Rate = \( r \)  
Rate = \( 1 \)  
Rate = \( r \)

3 Rounds 24 Rounds
SHA-3 Countermeasure

• Implementation
  – Rate Reduction

  – Round-Reduced Keccak Using 2 Gates at 3.7 GE
    (Synopsys Design Compiler at 130nm technology)
SHA-3 Countermeasure

• Implementation:
  – Performance
    \[(|IV| - 1) \times 3\] extra rounds
    \[(|IV| - 1)/8\] extra Keccak runs
  – Trading SCA-protection for performance New!
    Use \(s\) bits of IV per step
    \(s\) is an SCA-security parameter in \([1: |IV|]\)
    New performance: \((|IV| - 1)/8s\) extra Keccak runs
SHA-3 Countermeasure

• Comparison:
  against masked implementations of Keccak designers
  – Area

[Bar chart showing relative area for different implementations compares SHA-3 countermeasure to various methods, with the y-axis labeled as 'Relative Area' and the x-axis showing various implementations such as 'Unprotected reference', '3.81 One-cycle (f)', '2.65 One-cycle (c)', '2.4 Three-cycle (f)', '2.2 Three-cycle (m)', '1.98 Three-cycle (c)', and '1.0008 Our Work'.]
SHA-3 Countermeasure

- Comparison:
  - Performance
SHA-3 Countermeasure

• Comparison:
  – Flexibility
    SCA-protection can be tuned by software.
  – Compatibility
    Can be applied to already built modules
  – Portability
    Protect the Sponge with any permutation function
Outline

• Side-Channel Analysis
• SHA-3, the story and its applications
• SHA-3 Countermeasure
• New Design Concept
New Design Concept

Practical Methods
Masking & Hiding
- Fix current implementations
- Efficient solutions
- Practically verifiable

But
- Limited scope of protection
- Turns SCA harder
- Occasionally, breaks-down

Theoretical Methods
Leakage Resilient Cryptography
- Design new primitives
- Provable prevention against all differential attacks

Our Solution
- But
- No solution to current primitives
- Very high implementation cost
New Design Concept

Practical Methods

Masking & Hiding
- Fix current implementations
- Efficient solutions
- Practically verifiable

Theoretical Methods

Leakage Resilient Cryptography
- Design new primitives
- Provable prevention against all differential attacks

Our Solution
- Use LRC as a tool to protect current primitives
- Achieve a fine granularity of SCA-protection vs performance

Yes, our countermeasure belongs to LRC
Conclusion

- Proposed a countermeasure for the applications of SHA-3 at almost-zero area overhead.
- The performance overhead can be trivialized at long message lengths.
- The countermeasure is flexible, compatible and portable.
- Use theoretical ideas through practical experience.
Thank You
Questions?