Simpira v2: A Family of Efficient Permutations Using the AES Round Function

Shay Gueron\textsuperscript{1,2} and Nicky Mouha\textsuperscript{3,4,5}

\textsuperscript{1}Department of Mathematics, University of Haifa, Israel
\textsuperscript{2}Intel Corporation, Israel Development Center, Haifa, Israel
\textsuperscript{3}ESAT/COSIC, KU Leuven and iMinds, Belgium
\textsuperscript{4}Project-team SECRET, Inria, France
\textsuperscript{5}NIST, Gaithersburg, MD, USA

ASIACRYPT 2016
December 5, 2016

nicky@mouha.be
I’m Joint Author Of:

APE (PRIMATEs) (FSE 2014)
- Lightweight permutation-based authenticated encryption
- On-line misuse resistance, Releasing Unverified Plaintext (RUP) security
- PRIMATEs: Second-round CAESAR competition

Chaskey (SAC 2014)
- MAC algorithm for microcontrollers
- # 1 according to FELICS figure of merit
- ISO/IEC 29192-6 (draft)

Simpira (ASIACRYPT 2016)
- Family of permutations based on AES round function
- This presentation...
Background

**AES Instructions**

- Introduced by Intel (later AMD, recently ARM)
- On Intel Skylake: **AESENC** (1 round of AES)
  - Latency: 4 cycles
  - Throughput: 1 cycle
Background

AES Instructions

- Introduced by Intel (later AMD, recently ARM)
- On Intel Skylake: AESENC (1 round of AES)
  - Latency: 4 cycles
  - Throughput: 1 cycle

Focus: Throughput, Not Latency

- Requires parallelizable mode or independent data
- Problem inherent to AESENC!
Background

**AES Instructions**

- Introduced by Intel (later AMD, recently ARM)
- On Intel Skylake: AESENC (1 round of AES)
  - Latency: 4 cycles
  - Throughput: 1 cycle

**Focus: Throughput, Not Latency**

- Requires parallelizable mode or independent data
- Problem inherent to AESENC!

**Example to Motivate AESENC: Google Chrome**

- Recent 64-bit processors: AES-128-GCM
- If no AES instructions: ChaCha20-Poly1305
Limitations of AES

**Key Schedule: Round Keys**

- Calculate on-the-fly or store securely
- Tweak: not supported
Limitations of AES

Key Schedule: Round Keys
- Calculate on-the-fly or store securely
- Tweak: not supported

Block Size: Always 128 Bits
- Most modes of operation: insecure after $\sim 2^{64}$ blocks
- No secure hashing
Limitations of AES

Key Schedule: Round Keys
- Calculate on-the-fly or store securely
- Tweak: not supported

Block Size: Always 128 Bits
- Most modes of operation: insecure after $\sim 2^{64}$ blocks
- No secure hashing

Alternatives?
- Rijndael with 256-bit block size? SHA-2? ...
- Faster solution: Simpira
Design of Simpira

**Family of Permutations**

- $128 \times b$ bits, $b \in \mathbb{N}^+$
Design of Simpira

Family of Permutations
- $128 \times b$ bits, $b \in \mathbb{N}^+$

Building Block
- $b \geq 2$: (generalized) Feistel structure
Design of Simpira

**Family of Permutations**
- $128 \times b$ bits, $b \in \mathbb{N}^+$

**Building Block**
- $b \geq 2$: (generalized) Feistel structure
- Feistel $F$-function: two rounds of AES
Design of Simpira

Family of Permutations
- $128 \times b$ bits, $b \in \mathbb{N}^+$

Building Block
- $b \geq 2$: (generalized) Feistel structure
- Feistel $F$-function: two rounds of AES
- AESENCE: “free” XOR: add constant, combine branches
Design of Simpira

Family of Permutations
- $128 \times b$ bits, $b \in \mathbb{N}^+$

Building Block
- $b \geq 2$: (generalized) Feistel structure
- Feistel $F$-function: two rounds of AES
- AESENC: “free” XOR: add constant, combine branches

Design Goal
- Secure up to $2^{128}$ queries, very easy analysis
- Throughput: $\#$ cycles $\approx \#$ AESENC instructions
Design Requirements

Number of Rounds

• $\geq (\# \text{ rounds: 25 active S-boxes}) \times 3$
• $\geq (\# \text{ rounds: full bit diffusion}) \times 3$
• Note: same security for $\pi$ and $\pi^{-1}$
Design Requirements

Number of Rounds

- $\geq (\# \text{ rounds: 25 active S-boxes}) \times 3$
- $\geq (\# \text{ rounds: full bit diffusion}) \times 3$
- Note: same security for $\pi$ and $\pi^{-1}$

Efficiency

- Smallest number of $F$-functions
Design Requirements

Number of Rounds

- \( \geq (\# \text{ rounds: 25 active S-boxes}) \times 3 \)
- \( \geq (\# \text{ rounds: full bit diffusion}) \times 3 \)
- Note: same security for \( \pi \) and \( \pi^{-1} \)

Efficiency

- Smallest number of \( F \)-functions

Extra

- Multiple options: choose simplest design!
Simpira: $b = 1$

AES Permutation

- Rounds: 6
- # AESENC: 12
Simpira: $b = 2$

Feistel

- Rounds: 15
- # AESENC: 30
Simpira: $b = 3$

Type-1 GFS (Zheng et al.)

- Rounds: 21
- $\#$ AESENC: 42
Simpira: \( b \geq 4 \) (except \( b = 6 \) and \( b = 8 \))

Yanagihara-Iwata: Type 1.x (b,2) GFS

- Feistel rounds: \( 6b - 9 \)
- \# AESENC: \( 24b - 36 \)
Simpira: $b = 6$

Suzuki-Minematsu Improved Type-2 GFS

- Feistel rounds: 15
- $\#$ AESENC: 90
Simpira: $b = 8$

**Suzuki-Minematsu Improved GFS**

- Feistel rounds: 18
- $\#$ AESENC: 144
Attack on Simpira v1

Dobraunig et al. (SAC 2016)

- Collision on Simpira-based hash function \((b = 4)\)
- Full-round attack, complexity \(2^{83} \ (< 2^{128})\)
Attack on Simpira v1

**Dobraunig et al.** (SAC 2016)
- Collision on Simpira-based hash function \( b = 4 \)
- Full-round attack, complexity \( 2^{83} \) (<\( 2^{128} \))

**Rønjom** (ePrint 2016/248)
- Invariant subspace attack \( b = 4 \)
- Independent of # rounds, complexity: 2 queries (!)
Attack on Simpira v1

Dobraunig et al. (SAC 2016)

- Collision on Simpira-based hash function ($b = 4$)
- Full-round attack, complexity $2^{83}$ ($<2^{128}$)

Rønjom (ePrint 2016/248)

- Invariant subspace attack ($b = 4$)
- Independent of # rounds, complexity: 2 queries (!)

Underlying Problem: Yanagihara-Iwata Type 1.x GFS

- Careful with independences! Cryptographic permutation \( \neq \) Markov cipher with independent subkeys
- Invariant subspace attacks: often overlooked
- Fix: strengthen round constants, replace Type 1.x GFS
Simpira: \( b = 4 \)

**Type-2 GFS (Zheng et al.)**

- Feistel rounds: 15
- \# AESENC: 60
Simpira: $b \geq 5$ (except $b = 6$ and $b = 8$)

**Dedicated Construction**

- Iterate this three times
- $\#\ A{ESENC}: 24b - 36$
Simpira: $b \geq 5$ (except $b = 6$ and $b = 8$)

Dedicated Construction

- Iterate this three times
- $\# \text{ AESENC}: 24b - 36$
Simpira: \( b \geq 5 \) (except \( b = 6 \) and \( b = 8 \))

\[
\begin{align*}
128 & \\
& \vdots
\end{align*}
\]

**Dedicated Construction**

- Iterate this three times
- \# AESENC: \( 24b - 36 \)
Simpira: $b \geq 5$ (except $b = 6$ and $b = 8$)

Dedicated Construction

- Iterate this three times
- $\# \text{ AESENC: } 24b - 36$
Simpira: $b \geq 5$ (except $b = 6$ and $b = 8$)

Dedicated Construction

- Iterate this three times
- $\#\ AESENC: 24b - 36$
Benchmarks ($\pi$ and $\pi^{-1}$)

**Speed**

- **Theory:** up to 512 bits: < 1 c/B, large $b$: 1.5 c/B
- **Non-interleaved inputs:** up to 1024 bits: overhead < 3%
- **Interleaved inputs:** overhead < 3%, even for 4 kB inputs
Some Applications

Permutation-based Hashing

- Efficient processing of long and short messages
Some Applications

Permutation-based Hashing

- Efficient processing of long and short messages

(Tweakable) Even-Mansour Block Cipher

- $E_K = \text{Simpira}_b(P \oplus K \cdot T) \oplus K \cdot T$
- Tweak: $T \geq 1$, no tweak: $T = 1$
- Permutation size larger than $K \cdot T$: zero padding
Some Applications

Permutation-based Hashing
- Efficient processing of long and short messages

(Tweakable) Even-Mansour Block Cipher
- \( E_K = \text{Simplira}_b (P \oplus K \cdot T) \oplus K \cdot T \)
- Tweak: \( T \geq 1 \), no tweak: \( T = 1 \)
- Permutation size larger than \( K \cdot T \): zero padding

Robust Authenticated Encrypted Encryption
- Encode-then-encipher with Even-Mansour
- Encoding: multiple of 128 bits
Conclusion

**Simpira**

- Family of permutations
- $128 \times b$ bits, $b \in \mathbb{N}^+$
Conclusion

Simpira

- Family of permutations
- $128 \times b$ bits, $b \in \mathbb{N}^+$

Design

- Building block: two rounds of AES
- Security up to $2^{128}$ queries
- Simple design: easy security analysis
Conclusion

**Simpira**
- Family of permutations
- $128 \times b$ bits, $b \in \mathbb{N}^+$

**Design**
- Building block: two rounds of AES
- Security up to $2^{128}$ queries
- Simple design: easy security analysis

**Speed**
- Theoretical optimum: one `AESENC` every clock cycle
- Short inputs: 1 cycle/byte, large inputs: 1.5 cycles/byte
- Benchmarks: negligible overhead ($<3\%$)
Questions?
Supporting Slides
Algorithm 1 AESENC

1: procedure AESENC(state, key)
2: \hspace{1em} tmp \leftarrow state
3: \hspace{1em} tmp \leftarrow ShiftRows(tmp)
4: \hspace{1em} tmp \leftarrow SubBytes(tmp)
5: \hspace{1em} tmp \leftarrow MixColumns(tmp)
6: \hspace{1em} state \leftarrow tmp \oplus key
7: \hspace{1em} return state
8: end procedure

Algorithm 2 $F_{c,b}(x)$

1: procedure $F_{c,b}(x)$
2: \hspace{1em} $C \leftarrow SETR_EPI32(c, b, 0, 0)$
3: \hspace{1em} return AESENC(AESENC($x, C$), 0)
4: end procedure
Algorithm 3 Simpira \((b = 1)\)

1: **procedure** Simpira\((x_0)\)
2: \(R \leftarrow 6\)
3: **for** \(c = 1, \ldots, R\) **do**
4: \(x_0 \leftarrow F_{b,c}(x_0)\)
5: **end for**
6: **return** \(x_0\)
7: **end procedure**

Algorithm 4 Simpira\(^{-1}\) \((b = 1)\)

1: **procedure** Simpira\((x_0)\)
2: \(R \leftarrow 6\)
3: **MixColumns**\((x_0)\)
4: **for** \(c = R, \ldots, 1\) **do**
5: \(x_0 \leftarrow F_{b,c}^{-1}(x_0)\)
6: **end for**
7: **return** \(x_0\)
8: **end procedure**