SHA Extensionsを用いたAES-basedハッシュ関数の拡張

白矢 琢朗[1,a)]　石川 達也[1]　伊藤 竜馬[2,1]　Mostafizar Rahman[4]　阪本 光星[5,1]　田中 篤[1]
内海 潮音[3]　五十部 孝典[1,2]

概要：本稿では，多くの Intel および AMD プロセッサに搭載されている SHA の組込み関数（SHA Extensions）を，AES の組込み関数である AES-NI と統合することで，AES ベースの暗号方式を改良する．まず，最新の Intel プロセッサにおいて SHA Extensions が AES-NI と並列実装可能である特性に着目し，並列実装の条件や呼び出し回数の比率を調査し，両者を効率的に並列化するためのガイドラインを策定する．次に，策定したガイドラインを用いて，AES ベースで設計されたハッシュ関数である Areion および Simpira に SHA のラウンド関数を統合する．統合後のアルゴリズムに対しては，安全性評価を行い，さらに Intel，ARM，および AMD プロセッサ上で性能評価を実施する．その結果，ラウンド数の大幅な削減が可能となり，最大で約 50% の高速化を達成できることを示す．

キーワード：SHA Extensions, AES-NI, ハッシュ関数，衝突攻撃

Exploring SHA Instructions and Its Application to AES-based Hash Function

Takuro Shiraya[1,a)]　Tatsuya Ishikawa[1]　Ryoma Ito[2,1]　Mostafizar Rahman[4]
Kosei Sakamoto[5,1]　Atsushi Tanaka[1]　Shion Utsumi[3]　Takanori Isobe[1,2]

Abstract: In this paper, we explore the potential of improving AES-based schemes by integrating SHA instructions alongside AES instructions, starting from the key observation that SHA instructions can be executed in parallel with AES instructions on modern processors. We investigate conditions for parallel execution, the invocation ratio, and then provide guidelines for efficient SHA instruction usage with AES instructions. Applying these guidelines, we integrate SHA round functions into the AES-based short-input hash functions of Simpira and Areion, resulting in approximately 50% faster performance by achieving security with fewer instances.

Keywords: SHA Instructions, AES Instructions, hash function, forgery attacks

1. はじめに

1.1 背景

近年，AES をベースにして認証暗号（AEGIS [1], Rocca [2], [3], Tiaoxin-346 [4]）やワイドブロック暗号（Pholkos [5], Ghidle [6]），短入力ハッシュ関数（Simpira [7], Haraka v2 [8], Areion [9]）が設計されている．これらの暗号技術は AES-NI（Advanced Encryption Standard New Instructions）[10] などの命令を利用することで，ソフトウェア上での高効率化を実現している．AES-NI はブロック暗

[1] 大阪大学
The University of Osaka
[2] NICT
National Institute of Information and Communications Technology
[3] 兵庫県立大学
University of Hyogo
[4] 京都大学
Kyoto University
[5] 三菱電機株式会社
Mitsubishi Electric Corporation
[a)] takurou.shiraya727@gmail.com

号である AES の暗号化および復号処理を高速化するため，Intel 社によって提案された命令セットである．また，ARM および AMD プロセッサにおいては，AES-NI に対応する命令セットとして，それぞれ ARMv8 Cryptography Extensions [11] および AMD64 Architecture Programmer's Manual [12] が存在する．

一方，Intel，ARM，AMD の最新プロセッサは，SHA-1 や SHA-256 といった他のアルゴリズムも搭載している．2013 年頃から多くの Intel および AMD プロセッサには，SHA-1 と SHA-256 のハードウェアによる高速化を提供する SHA Extensions が搭載されている，ARM プロセッサにおいては，SHA Extensions は ARMv8.2-A 以降で導入された Cryptographic Extension の一部として提供されている，多くのハイエンド向けスマートフォンやタブレット，Qualcomm（Snapdragon シリーズ），Apple（A/M シリーズ），Samsung（Exynos）のプロセッサを搭載する高性能デバイスにおいて，SHA Extensions は採用されている．しかし AES-NI と比較して安全性への貢献が少ないため，ソフトウェア上で効率的に実行可能な暗号設計において，あまり着目されていない．

特筆すべき点は，最新プロセッサ上で AES-NI と SHA Extensions は並列に実装可能である，ということである．具体的には，AES-NI と SHA Extensions の実行ポートが異なるため，SHA Extensions の処理が AES-NI に干渉することはない．そのため AES をベースにして設計された既存技術に対して，SHA Extensions を並列実行させることで実装性能が向上すれば，高速化が必要不可欠な 6G 環境において重要な技術となる．また異なる処理を並列に実装することで，一方の命令に対して脆弱性が発見された場合でも，安全性が保証される可能性がある．

1.2 貢献

本研究では，AES-NI と SHA Extensions を併用した場合の特性を検証し，AES ベースの暗号方式における性能向上の可能性を考察する．本研究の具体的な貢献は，次のとおりである．

- AES-NI と SHA Extensions の並列実装の条件や実装比率，オーバーヘッドを調査し，両命令を効率的に活用するためのガイドラインを策定した．
- このガイドラインの有効性を検証するために，短入力ハッシュ関数である Areion および Simpira に SHA-256 ラウンド関数を統合した．オーバーヘッドを回避するため，SHA-256 の使用回数を最適化し，オリジナル方式と比較して衝突攻撃に対する安全性が大幅に向上することを示した．つまり，対象となるハッシュ関数の基礎となる暗号学的置換に関し，オリジナル方式と同等の安全性を維持する範囲でラウンド数を削減することにより，実装性能の向上が可能となる．
- Intel および ARM プロセッサで実装性能を評価した結果，SHA-256 ラウンド関数を統合した短入力ハッシュ関数は，オリジナル方式と比較して約 50%の高速化を達成した．

Areion-256 は低遅延ハッシュ関数として，IETF において標準化の議論が行われている．多様な性能指標の中でも，処理速度が最も重要な評価項目であり，本提案手法における性能向上の実現によって，遠隔ロボット手術や V2X（Vehicle-to-Everything）通信，データセンターなど，高性能が求められる分野での活用が期待される．

本稿の構成は以下の通りである．まず，第 2 章で本研究で使用する組込み関数と調査対象のハッシュ関数，および性能評価の測定環境について説明する．次に，第 3 章では調査した命令特性を示す．第 4 章では，第 3 章で示した特性を用いて，並列実装の構成を決定する．決定した構成に対して，第 5 章で性能評価を実施する．最後に，第 6 章で結論を述べる．

2. 準備

本章では，本研究で使用する組込み関数について説明する．次に，ハッシュ関数である Areion-256 および Simpira-256 について述べる．最後に，性能評価を実施する Intel，ARM，AMD プロセッサについて説明する．

2.1 組込み関数

近年の多くのプロセッサは，並列処理の一種である SIMD（Single Instruction, Multiple Data）命令セットを搭載しており，これらの命令は専用レジスタに格納されたデータを用いて，並列演算などの高度な処理を単一命令で実行を可能にする．SIMD 命令の性能は通常，レイテンシ，スループット，および使用する実行ポートによって評価される．レイテンシは，命令の実行に必要なクロックサイクル数を指す．スループットは，同一ポートが同じ命令を再び実行するまでのクロックサイクル数である．本研究では，AES-NI と SHA Extensions に着目する．以下に，各命令の詳細について説明する．

2.1.1 AES-NI

AES-NI（AES New Instructions）は，ブロック暗号 AES の演算を実行できる命令セットであり，Intel プロセッサおよび AMD プロセッサで使用可能である．AES-NI には，暗号化のラウンド関数を実行する aesenc，最終ラウンドを実行する aesenclast，復号命令，およびラウンド鍵の生成を支援する命令が存在する．

一方，本研究において使用する ARMv8 プロセッサでは，AES-NI は NEON 命令セット [11] に含まれている．NEON 命令セットにおける AES-NI には，AES のラウンド関数ではある AddRoundkey, Subbytes, ShiftRows を実行する vaeseq と，MixColumns を実行する vaesmcq ，お

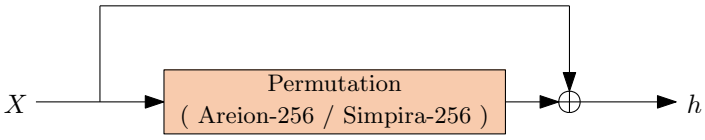

図 1 DM モードに基づく Areion-256 および Simpira-256 の構成

よび復号命令が存在する．なおラウンド鍵の生成を支援する命令は存在しない．

2.1.2 SHA Extensions

SHA Extensions は，x86-64 アーキテクチャ上でハッシュ関数である SHA-1 と SHA-256 の演算を高速化するために導入された命令セットである．この命令セットには，SHA-1 と SHA-256 のメッセージスケジュールとラウンド関数の処理も含まれる．本研究では，この命令セットのうち，SHA-256 の圧縮関数を 2 ラウンド実行する命令である `sha256rnds2` に着目する．また，ARM プロセッサでも同様の命令である `sha256h` と `sha256h2` が提供されている．ラウンド関数は `sha256h` と `sha256h2` によって並列に実装され，256 ビットの内部ステートのうち 128 ビットずつ更新される．具体的には，`sha256h` が上位 128 ビットに対して圧縮関数を 1 ラウンド分更新し，`sha256h2` が下位 128 ビットを更新することで，1 ラウンドの処理を完了する．

2.1.3 その他の命令

x86-64 アーキテクチャ上では，`pxor` は 2 つのオペランドに対してビット単位の XOR 演算を実施する．ARM プロセッサでは，NEON 命令セットの `veor` が同様の命令である．

2.2 ハッシュ関数

Areion-256 [9] および Simpira-256 [7] はラウンド関数を用いて内部ステートを並べ替えるハッシュ関数である．これらのハッシュ関数は，入力データを X，出力されるハッシュ値を h とすると，**図 1** に示すように，DM（Davies-Meyer）モードに基づいて構成されている．本研究では，ラウンド関数に着目して拡張するため，主にラウンド関数について説明する．

2.2.1 Areion-256

AES のラウンド関数である SubBytes, ShiftRows, MixColumns をそれぞれ SB, SR, MC とする．AC は，AES のラウンド関数である AddRoundKey と同様の処理であるが，ラウンド鍵ではなく定数が加えられる．ラウンド関数内で使用される関数 F_1, F_2 は，以下の式で定義される．

$$F_1(X) = SR \circ SB(X),$$
$$F_2(X) = MC \circ SR \circ SB \circ AC \circ MC \circ SR \circ SB(X).$$

図 2 に示すラウンド関数 $R(S)$ は，内部ステート S を入力とし，以下で定義する処理を 10 ラウンド繰り返す．

$$S'[0] = S[1] \oplus F_2(S[0]), \qquad S'[1] = F_1(S[0]).$$

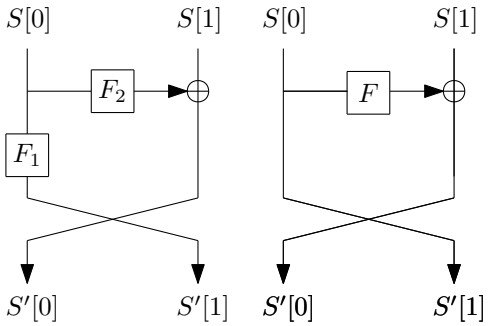

図 2 Areion-256 （左）および Simpira-256 （右）のラウンド関数

2.2.2 Simpira-256

ラウンド関数内で使用される関数 F は以下の式で定義される．

$$F(X) = AC \circ MC \circ SR \circ SB \circ AC \circ MC \circ SR \circ SB(X).$$

図 2 に示すラウンド関数 $R(S)$ は，内部ステート S を入力とし，以下で定義する処理を 15 ラウンド繰り返す．

$$S'[0] = S[1] \oplus F(S[0]), \qquad S'[1] = S[0].$$

2.3 測定環境

先行研究（Areion-256 [9], Simpira-256 [7]）との評価の一貫性を確保するため，ソフトウェア上での性能評価用プロセッサとして，Intel プロセッサおよび ARM プロセッサを選択した．また，網羅性を高めるため，AMD プロセッサでも性能評価を実施した．以下に，測定に用いた各プロセッサの詳細を述べる．

2.3.1 Intel プロセッサ

本研究において，Intel プロセッサでの性能評価はすべて Alder Lake (Intel(R) Core(TM) i9-12900K processor @ 3.20 GHz with a performance-core (P-core)) で実施した．また，同様の評価を Lunar Lake (Intel(R) Core(TM) Ultra 5 226V @ 2.1 GHz with a performance-core (P-core)) でも実施した．なお，Lunar Lake は命令特性が非公開でありノート型パソコン向けの設計であるため，本研究の主たる評価対象外としたが，参考として測定を行った．また，評価の信頼性を確保するため，並列処理を向上させる HTT（Hyper-Threading Technology）と CPU の動作周波数を引き上げる TBT（Intel Turbo Boost Technology）を無効化した．

2.3.2 ARM プロセッサ

本研究において，ARM プロセッサでの性能評価はすべて ARMv8.6-A（Apple M3 processor @ 4.05 GHz）で実施した．ARM プロセッサでは HTT は使用できないが，Apple M3 では TBT を無効化できないため，TBT を有効のまま性能評価を行った．

2.3.3 AMD プロセッサ

本研究において，AMD プロセッサでの性能評価はすべ

表 1 Intel プロセッサである Ice Lake，Tiger Lake，Alder Lake における，命令のレイテンシ，スループット，実行ポート

組込み関数	命令	レイテンシ	スループット	実行ポート
_mm_aesenc_si128	aesenc	3	0.5	0,1
_mm_sha256rnds2_epu32	sha256rnds2	4	3	5
_mm_xor_si128	pxor	1	0.33	0,1,5

表 2 2 つの aesenc を 12 回繰り返し，並列実装する sha256rnds2 の呼び出し回数ごとの速度（Gbps）

命令数	1	2	3	4	5	6	7	8	9	**10**	11
Speed (CPB)	0.562	0.562	0.562	0.562	0.564	0.562	0.562	0.562	0.562	**0.625**	0.687
Speed (Gbps)	45.479	45.518	45.496	45.500	45.381	45.503	45.495	45.482	45.510	**40.918**	37.230
命令数	12	13	14	15	16	17	18	19	20	21	22
Speed (CPB)	0.750	0.812	0.874	0.939	0.999	1.062	1.124	1.127	1.250	1.312	1.375
Speed (Gbps)	34.132	31.495	29.259	27.243	25.608	24.095	22.758	21.712	20.466	19.504	18.612

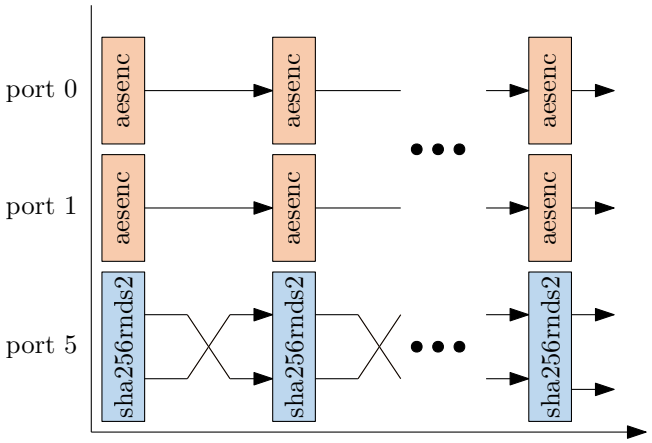

図 3 aesenc と sha256rnds2 の並列実装

て Zen 4（AMD EPYC 9654 @2.4 GHz）で実施した．同様の評価を Zen 5（AMD Ryzen 9 9950X @ 4.3GHz）でも実施した．また，評価の信頼性を確保するため，Intel プロセッサの HTT および TBT に相当する SMT（Simultaneous Multi-Threading）および PBO（Precision Boost Overdrive）を無効化した．

3. SHA Extensions の命令特性

本章では，SHA Extensions の命令特性，特に Intel プロセッサ（x86-64）において AES-NI と並列実装時の特性を調査する．

3.1 レイテンシおよびスループット

本研究では，AES のラウンド関数を 1 ラウンド実行する aesenc，SHA-256 の圧縮関数を 2 ラウンド実行する sha256rnds2, 128 ビットの XOR である pxor の 3 つの命令を検討する．**表 1** に Intel プロセッサである Ice Lake, Tiger Lake，Alder Lake における，各命令のレイテンシ，スループット，および実行ポートを示す．表 1 より，sha256rnds2 はポート 5 を使用する．aesenc はポート 0 またはポート 1 を使用するため，sha256rnds2 と aesenc は並列に実行可能である．しかし aesenc のスループットを維持するため，sha256rnds2 の実行回数を考慮する必要がある．

3.2 並列実装

aesenc と sha256rnds2 の並列実装における特性を検証する．表 1 より，理論的には sha256rnds2 を 3 回繰り返す実行時間は，aesenc を 4 回繰り返す実行時間と同等である．この理論を検証するため，aesenc , sha256rnds2 の並列実装を実施する．具体的には，**図 3** に示すように，12 回繰り返す 2 つの aesenc に対して，sha256rnds2 の呼び出し回数を変化させながら並列実装した場合の性能を評価する．aesenc は 2 つの独立したポート（ポート 0 と 1）を使用して，2 つの aesenc を同時に実行する．

表 2 に Intel プロセッサにおいて，12 回繰り返す 2 つの aesenc に対して，sha256rnds2 の回数を変化させながら並列実装した場合の性能評価結果を示す．これらの結果は，Github[*1]上のソースコードを用いて，サイクルカウンタ（CPB: cycles per byte）を測定して得たものである．サイクルカウンタとクロック周波数（例：Intel 環境では 3.20 GHz）を式 (1) に代入することで，速度（Gbps: gigabitper second）に換算した．

$$\text{速度（Gbps）} = \frac{\text{クロック周波数（GHz）} \times 8}{\text{サイクルカウンタ（CPB）}}. \quad (1)$$

以降の性能評価でも，速度は同様の手法で算出する．表 2 より，12 回繰り返す 2 つの aesenc に対して，sha256rnds2 は最大 9 回まで速度に影響なく並列実装可能であることが分かる．この結果は理論上の予測どおり，Y 回繰り返す aesenc に対して，sha256rnds2 は最大で $X = \frac{3}{4}Y$ 回実装できることを示している．

3.3 統合のガイドライン

本節では，AES ベースの暗号方式に SHA Extensions を統合する方法について述べる．まず，対象とする暗号方式

[*1] https://github.com/seb-m/cycles/blob/master/cycle.h

表 3　Areion-256 および Simpira-256 の各ラウンド数に対して，性能に影響を与えない SHA-256 のラウンド数

対象	Areion-256					Simpira-256					
# Rounds	4R	5R	6R	7R	8R	10R	11R	12R	13R	14R	15R
# sha256rnds2	6	7	9	10	12	15	16	18	19	21	22
(SHA-256 Rounds)	(12)	(14)	(18)	(20)	(24)	(30)	(32)	(36)	(38)	(42)	(44)

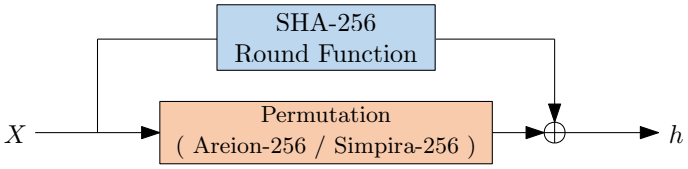

図 4　Areion-256 および Simpira-256 と sha256rnds2 を統合した構成概要

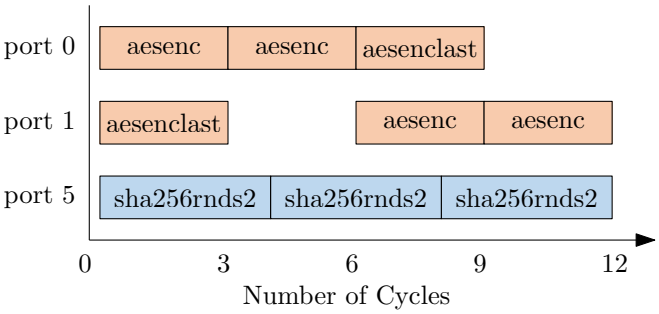

図 5　Intel プロセッサである Ice Lake/Tiger Lake/Alder Lake での Areion-256 の 2 ラウンドと sha256rnds2 の 3 回呼び出しにおけるパイプライン処理の可視化

で使用される命令の詳細を調査する．具体的には，特定のプロセッサにおける各命令のレイテンシ，スループット，および実行ポートを対象とする．この調査に基づき，SHA Extensions との並列実装を検証し，どのポートに命令を割り当て可能かを判断する．最後に，新しい命令をポートに割り当て，安全性を維持したままラウンド数の削減可能性を探索する．この手順において，満たすべき要件は以下の 2 つである．

(1) 新しい命令を追加する場合，対象となる暗号方式の命令処理に影響を与えてはならない．対象となる暗号方式の命令処理は最適化されていることが多く，影響を与えることは性能を大幅に低下させる可能性がある．

(2) 性能と安全性を保証するため，AES-NI と SHA Extensions は独立にかつ並列で処理する必要がある．新たに追加した命令を対象となる暗号方式に組み込む場合，暗号方式が本来保証している安全性が低下し，詳細な安全性評価を再度実施する必要がある．

上記のガイドラインに従い，第 4 章では短入力ハッシュ関数である Areion-256 [9] および Simpira-256 [7] に SHA Extensions を統合する．

4. 短入力ハッシュ関数への統合

本章では，短入力ハッシュ関数である Areion-256 [9] および Simpira-256 [7] に SHA Extensions を統合する．

4.1 オリジナル構成との差異

3.3 節のガイドラインに従い，DM モードの Areion-256 および Simpira-256 に sha256rnds2 を統合する．**図 4** に示すように，提案する構成では SHA-256 と Areion-256 および Simpira-256 は並列に実行する．SHA-256 への入力は Areion-256 および Simpira-256 と同じであるため，オリジナル構成と比較して衝突攻撃に対する安全性が大幅に向上することを示す．その結果，ラウンド数を削減することで，安全性を維持しながら性能を向上させることができる．

4.2 オーバーヘッドの調査

Areion-256 および Simpira-256 の性能に影響を与えないよう，SHA-256 のラウンド数を選択する必要がある．**表 3** に Areion-256 および Simpira-256 のラウンド数に対して，性能に影響を与えない SHA-256 のラウンド数を示す．例として，Areion-256/Simpira-256 の 7/10 ラウンドに対して，sha256rnds2（sha256rnds2 は SHA-256 を 2 ラウンド実行） は 10/15 回の範囲であれば性能に影響を与えることなく統合可能である．**図 5** に Areion-256 の 2 ラウンドと sha256rnds2 の 3 回呼び出しにおけるパイプライン処理の可視化を示す．これより，sha256rnds2 は Areion-256 の命令処理に影響を与えていないことが分かる．表 3 の結果をもとに，4.3 節では衝突攻撃に対する安全性に着目して統合した構成のラウンド数を決定する．なお，X ラウンドの Areion-256/Simpira-256 と Y ラウンドの SHA-256 を組み合わせたものを，Xareion-Ysha256/Xsimpira-Ysha256 とする．

4.3 ラウンド数の決定

本節では，ハッシュ関数に対して最も重要な攻撃の一つである衝突攻撃の安全性に基づき，Xareion-Ysha256/Xsimpira-Ysha256 のラウンド数を決定する．

4.3.1 衝突攻撃への評価手法

衝突攻撃に対する安全性を正確に評価するため，Xareion-Ysha256/Xsimpira-Ysha256 について衝突が発生するときの差分特性確率（DCP）を導出する必要がある．しかし，現実的な時間内で評価することか計算量的に不可能である．そこで，Areion-256 /Simpira-256 と SHA-256 の DCP を足し合わせることにより，

表 4　Areion-256 および Simpira-256 における active S-box の下界

Target \ Rounds	1R	2R	3R	4R	5R	6R	7R	8R	9R	10R	11R	12R	13R	14R	15R	References
Areion-256	0	6	12	38	46	53	60	86	92	99	108	135	–	–	–	byte-wise
Areion-256	0	5	10	25	30	38	44	59	64	71	78	93	–	–	–	[13]
Simpira-256	0	5	10	25	30	35	40	55	60	65	70	85	90	95	100	byte-wise

表 5　SHA-256 の最大差分特性確率 $[-\log_2]$

Target \ Rounds	1R	2R	3R	4R	5R	6R	7R	8R	9R	10R	11R	12R
SHA-256	0	1	1	5	9	13	20	42	55	77	91 -	98 - 142

表 6　Xareion-Ysha256 の差分特性確率（DCP）の上界 $[-\log_2]$

Target	DCP
10R Areion-256	**426** (#AS = 71)
7R Areion-256	264 (#AS = 44)
7areion-20sha256	**418**
8R Areion-256	354 (#AS = 59)
8areion-24sha256	**550**

表 7　Xsimpira-Ysha256 の差分特性確率（DCP）の上界 $[-\log_2]$

Target	DCP
15R Simpira-256	**600** (#AS = 100)
10R Simpira-256	390 (#AS = 65)
10simpira-30sha256	**621**
11R Simpira-256	420 (#AS = 70)
11simpira-32sha256	**679**

Xareion-Ysha256/Xsimpira-Ysha256 の DCP の上限を導出する．DCP の合計を数学的に解析する手法は Asiacrypt 2024 [14] で提案され，Orthros [15] でも同様の評価が行われ，Gleeok [16] では設計者による評価が実施された．

Sun らによって提案された SAT によるモデリング手法 [17], [18] を用いて，評価を実施する．具体的には，SHA-256 はビット単位で DCP を探索する．Areion-256 および Simpira-256 に関しては，ビット単位の評価では計算時間がかかるため，active S-box による下界評価をバイト単位で探索する．active S-box による下界から計算した DCP の上界と SHA-256 の DCP によって，Xareion-Ysha256/Xsimpira-Ysha256 の衝突攻撃による DCP の上限を導出する．

4.3.2　衝突攻撃への評価結果

表 4 に Areion-256 および Simpira-256 の active S-box による下界を示す．AES の S-box の差分確率は 2^{-6} または 2^{-7} であるが，本論文では保守的に 2^{-6} として扱う．**表 5** に SHA-256 の最大差分特性確率を示す．

4.3.2.1　Areion-256

表 6 に Xareion-Ysha256 の DCP の上限を示す．ここで，X と Y のラウンド数は，オリジナル構成の性能に影響を与えないように決定している．表 6 より，7areion-20sha256 はフルラウンドである 10 ラウンドの Areion-256 に近い安全性を有することが確認できる．

4.3.2.2　Simpira-256

表 7 に Xsimpira-Ysha256 の DCP の上限を示す．ここで，X と Y のラウンド数は，オリジナル構成の性能に影響を与えないように決定している．表 7 より，10simpira-30sha256 はフルラウンドである 15 ラウンドの Simpira-256 より高い安全性を有することが確認できる．

5.　性能評価

本章では，7areion-20sha256 および 10simpira-30sha256 の性能評価を実施する．統合された構成を検証するため，7areion-20sha256 は 7/10 ラウンドの Areion-256 と比較する．10simpira-30sha256 は 7/10 ラウンドの Simpira-256 と比較する．また既存のハッシュ関数である BLAKE3 , SHA2-256 , SHA3-256 や Ascon-Hash との性能比較も実施する．これらの評価では，SUPERCOP[*2] と GitHub[*3,*4,*5] で公開されているソースコードを用いて実施する．

表 8 に Intel，ARM，AMD プロセッサでの性能評価結果を示す．7areion-20sha256 および 10simpira-30sha256 の入力は最大 32 バイトであるため，全ての評価は 32 バイトの入力で実施している．

5.1　Intel プロセッサ

速度は式 (1) で算出した．表 8 より，本研究で提案した 7areion-20sha256 および 10simpira-30sha256 は 7 ラウンドの Areion-256 および 10 ラウンドの Simpira-256 に匹敵する性能を達成した．つまり，我々が提案した 7areion-20sha256 および 10simpira-30sha256 は，フルラウンドの Areion-256 および Simpira-256 と比較して約 40%高速であることが確認できる．したがって，我々が提案した構成はオリジナル構成のフルラウンドに匹敵する安全性を維持しつつ，Intel プロセッサ上で高い性能を実現

[*2] https://bench.cr.yp.to/supercop.html
[*3] https://github.com/BLAKE3-team/BLAKE3
[*4] https://github.com/XKCP/XKCP
[*5] https://github.com/ascon/ascon-c

表 8 Intel，ARM，AMD プロセッサにおける性能評価結果（全ての評価は 32-byte の入力で実施）

プロセッサ	評価対象	実装方法	サイクルカウンター（CPB）	速度（Gbps）
Intel	7areion-20sha256	AES + SHA	**1.365**	**18.751**
Alder Lake	(7R Areion-256)	AES	1.364	18.757
	10R Areion-256	AES	1.983	12.904
	10simpira-30sha256	AES + SHA	**1.968**	**13.004**
	(10R Simpira-256)	AES	1.967	13.010
	15R Simpira-256	AES	2.812	9.101
	BLAKE3	AVX2	10.375	2.467
	SHA2-256	SHA	5.781	4.428
	SHA3-256	AVX2	37.469	0.683
	Ascon-Hash	opt64	44.156	0.580
Intel	7areion-20sha256	AES + SHA	**2.033**	**8.263**
Lunar Lake	(7R Areion-256)	AES	2.015	8.337
	10R Areion-256	AES	2.961	5.673
	10simpira-30sha256	AES + SHA	**2.995**	**5.609**
	(10R Simpira-256)	AES	2.932	5.729
	15R Simpira-256	AES	4.188	4.011
	BLAKE3	AVX2	10.750	1.562
	SHA2-256	SHA	5.938	2.829
	SHA3-256	AVX2	32.125	0.522
	Ascon-Hash	opt64	37.750	0.445
ARM	7areion-20sha256	NEON	**2.287**	**14.163**
	(7R Areion-256)	NEON	2.237	14.480
	10R Areion-256	NEON	3.295	9.831
	10simpira-30sha256	NEON	**2.808**	**11.538**
	(10R Simpira-256)	NEON	2.646	12.240
	15R Simpira-256	NEON	3.838	8.441
	BLAKE3	NEON	8.156	3.973
	SHA2-256	NEON	2.625	12.343
	SHA3-256	opt64	18.469	1.754
	Ascon-Hash	opt64	34.812	0.931
AMD	7areion-20sha256	AES + SHA	**1.692**	**11.195**
Zen 4	(7R Areion-256)	AES	1.662	11.546
	10R Areion-256	AES	2.541	7.556
	10simpira-30sha256	AES + SHA	**2.541**	**7.556**
	(10R Simpira-256)	AES	2.541	7.556
	15R Simpira-256	AES	3.672	5.228
	BLAKE3	AVX2	11.031	1.740
	SHA2-256	SHA	5.438	3.530
	SHA3-256	AVX2	45.750	0.419
	Ascon-Hash	opt64	40.656	0.472
AMD	7areion-20sha256	AES + SHA	**1.715**	**20.058**
Zen 5	(7R Areion-256)	AES	1.697	20.271
	10R Areion-256	AES	2.554	13.469
	10simpira-30sha256	AES + SHA	**2.546**	**13.511**
	(10R Simpira-256)	AES	2.545	13.516
	15R Simpira-256	AES	3.797	9.059
	BLAKE3	AVX2	12.562	2.738
	SHA2-256	SHA	6.625	5.192
	SHA3-256	AVX2	40.188	0.855
	Ascon-Hash	opt64	31.781	1.082

する．

5.2 ARM プロセッサ

速度は macOS kernel（XNU）[*6]の kperf ライブラリで測定した．本研究で提案した 7areion-20sha256 および 10simpira-30sha256 は，10 ラウンドの Areion-256 および 15 ラウンドの Simpira-256 よりも約 40%高速である．したがって，我々が提案した構成は ARM プロセッサでも安全性と性能を両立できる．Apple M3 プロセッサのレイテンシとスループットは公開されていないが，$X \leq \frac{3}{4}Y$ の場合，Intel プロセッサと同様の条件で aesenc に性能に影響を与えることなく sha256rnds2 を追加し，独立して実行できることが確認できる．

[*6] https://github.com/apple-oss-distributions/xnu

5.3 AMD プロセッサ

7areion-20sha256 および 10simpira-30sha256 は，10 ラウンドの Areion-256 および 15 ラウンドの Simpira-256 よりも約 50%高速である．したがって，我々が提案した構成は AMD プロセッサでも安全性と性能を両立できる．

6. 結論

本論文では，AES ベースの暗号方式について，AES-NI に SHA Extensions を統合することによる性能向上を検討した．AES-NI と SHA Extensions を並列実装をするための条件や呼び出し回数の比率を調査し，SHA Extensions を統合するためのガイドラインを策定した．策定したガイドラインを用いて，短入力ハッシュ関数である Areion および Simpira に SHA Extensions を統合した．その結果，オリジナル構成の安全性を維持したままラウンド数を削減し，最大で約 50%の高速化を達成した．

謝辞 本研究は, JSPS 科研費 JP24H00696 と JST AIP 加速課題 JPMJCR24U1 の支援を受けたものである．

参考文献

[1] Wu, H. and Preneel, B.: AEGIS: A Fast Authenticated Encryption Algorithm, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers* (Lange, T., Lauter, K. E. and Lisonek, P., eds.), Lecture Notes in Computer Science, Vol. 8282, Springer, pp. 185–201 (online), DOI: 10.1007/978-3-662-43414-7_10 (2013).

[2] Sakamoto, K., Liu, F., Nakano, Y., Kiyomoto, S. and Isobe, T.: Rocca: An Efficient AES-based Encryption Scheme for Beyond 5G, *IACR Trans. Symmetric Cryptol.*, Vol. 2021, No. 2, pp. 1–30 (online), DOI: 10.46586/tosc.v2021.i2.1-30 (2021).

[3] Sakamoto, K., Liu, F., Nakano, Y., Kiyomoto, S. and Isobe, T.: Rocca: An Efficient AES-based Encryption Scheme for Beyond 5G (Full version), *IACR Cryptol. ePrint Arch.*, p. 116 (online), available from ⟨https://eprint.iacr.org/2022/116⟩ (2022).

[4] Nikolic, I.: Tiaoxin-346, *Submission to the CAESAR competition* (2014).

[5] Bossert, J., List, E., Lucks, S. and Schmitz, S.: Pholkos - Efficient Large-State Tweakable Block Ciphers from the AES Round Function, *Topics in Cryptology - CT-RSA 2022 - Cryptographers' Track at the RSA Conference 2022, Virtual Event, March 1-2, 2022, Proceedings* (Galbraith, S. D., ed.), Lecture Notes in Computer Science, Vol. 13161, Springer, pp. 511–536 (online), DOI: 10.1007/978-3-030-95312-6_21 (2022).

[6] Nakahashi, M., Shiba, R., Anand, R., Rahman, M., Sakamoto, K., Liu, F. and Isobe, T.: Ghidle: Efficient Large-State Block Ciphers for Post-quantum Security, *Information Security and Privacy - 28th Australasian Conference, ACISP 2023, Brisbane, QLD, Australia, July 5-7, 2023, Proceedings* (Simpson, L. and Baee, M. A. R., eds.), Lecture Notes in Computer Science, Vol. 13915, Springer, pp. 403–430 (online), DOI: 10.1007/978-3-031-35486-1_18 (2023).

[7] Gueron, S. and Mouha, N.: Simpira v2: A Family of Efficient Permutations Using the AES Round Function, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I* (Cheon, J. H. and Takagi, T., eds.), Lecture Notes in Computer Science, Vol. 10031, pp. 95–125 (online), DOI: 10.1007/978-3-662-53887-6_4 (2016).

[8] Kölbl, S., Lauridsen, M. M., Mendel, F. and Rechberger, C.: Haraka v2 - Efficient Short-Input Hashing for Post-Quantum Applications, *IACR Trans. Symmetric Cryptol.*, Vol. 2016, No. 2, pp. 1–29 (online), DOI: 10.13154/TOSC.V2016.I2.1-29 (2016).

[9] Isobe, T., Ito, R., Liu, F., Minematsu, K., Nakahashi, M., Sakamoto, K. and Shiba, R.: Areion: Highly-Efficient Permutations and Its Applications to Hash Functions for Short Input, *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, Vol. 2023, No. 2, pp. 115–154 (online), DOI: 10.46586/TCHES.V2023.I2.115-154 (2023).

[10] Gueron, S.: Advanced Encryption Standard AES New Instructions Set, Technical report, Intel Corporation (2010).

[11] Limited, A.: *ARMv8 Cryptography Extensions* (2022).

[12] AMD: AMD64 Architecture Programmer' s Manual Volume 4: 128-Bit and 256-Bit Media Instructions (2021).

[13] Taiyama, K., Sakamoto, K., Shiba, R. and Isobe, T.: Collision Attacks on Hashing Modes of Areion, *Cryptology and Network Security - 23rd International Conference, CANS 2024, Cambridge, UK, September 24-27, 2024, Proceedings, Part II* (Kohlweiss, M., Pietro, R. D. and Beresford, A., eds.), Lecture Notes in Computer Science, Vol. 14906, Springer, pp. 265–285 (online), DOI: 10.1007/978-981-97-8016-7_12 (2024).

[14] Flórez-Gutiérrez, A., Grassi, L., Leander, G., Sibleyras, F. and Todo, Y.: General Practical Cryptanalysis of the Sum of Round-Reduced Block Ciphers and ZIP-AES, *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part IX* (Chung, K. and Sasaki, Y., eds.), Lecture Notes in Computer Science, Vol. 15492, Springer, pp. 280–311 (online), DOI: 10.1007/978-981-96-0947-5_10 (2024).

[15] Taka, K., Ishikawa, T., Sakamoto, K. and Isobe, T.: An Efficient Strategy to Construct a Better Differential on Multiple-Branch-Based Designs: Application to Orthros, Cryptology ePrint Archive, Paper 2023/674 (2023).

[16] Anand, R., Banik, S., Caforio, A., Ishikawa, T., Isobe, T., Liu, F., Minematsu, K., Rahman, M. and Sakamoto, K.: Gleeok: A Family of Low-Latency PRFs and its Applications to Authenticated Encryption, *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, Vol. 2024, No. 2, pp. 545–587 (online), DOI: 10.46586/TCHES.V2024.I2.545-587 (2024).

[17] Sun, L., Wang, W. and Wang, M.: More Accurate Differential Properties of LED64 and Midori64, *IACR Trans. Symmetric Cryptol.*, Vol. 2018, No. 3, pp. 93–123 (online), DOI: 10.13154/tosc.v2018.i3.93-123 (2018).

[18] Sun, L., Wang, W. and Wang, M.: Accelerating the Search of Differential and Linear Characteristics with the SAT Method, *IACR Trans. Symmetric Cryptol.*, Vol. 2021, No. 1, pp. 269–315 (online), DOI: 10.46586/tosc.v2021.i1.269-315 (2021).