PAPER

On Optimality of the Round Function of Rocca

Nobuyuki TAKEUCHI[†a)], ***Nonmember***, **Kosei SAKAMOTO**[††], ***and*** **Takanori ISOBE**[†,††,†††,††††], ***Members***

SUMMARY At ToSC 2021, Sakamoto et al. proposed Rocca, an AES-based encryption scheme, for Beyond 5G applications. They presented a class of round functions that achieved impressive performance in software by improving the design strategy for constructing an efficient AES-based round function that was proposed by Jean and Nikolić at FSE 2016. In this paper, we revisit their design strategy for finding more efficient round functions. We add new requirements further to improve speed of Rocca. Specifically, we focus on the number of temporary registers for updating the round function and search for round functions with the minimum number of required temporary registers. As a result, we find a class of round functions with only one required temporary register, while round function of Rocca requires two temporary registers. We show that new round functions are significantly faster than that of Rocca on the latest Ice Lake and Tiger Lake architectures. We emphasize that, regarding speed, our round functions are optimal among the Rocca class of round functions because the search described in this paper covers all candidates that satisfy the requirements of Rocca.

key words: *AES-NI, fast software implementation, AEAD, round function*

1. Introduction

1.1 Background

High-efficiency cryptographic software schemes are essential for the 5th generation mobile communication system (5G) and the 6th generation mobile (6G), or Beyond 5G, communication systems. Indeed, several high-efficiency cryptographic schemes targeted for 5G, such as SNOW-V [5] and ZUC-256 [4] have recently been proposed.

Many high-efficiency cryptographic schemes used on software, including SNOW-V, make use of Advanced Encryption Standard New Instructions (AES-NI) [3], [6] to achieve impressively high execution speed. AES-NI is a special set in a Single Instruction Multiple Data (SIMD) instruction designed to accelerate AES encryption and decryption that has been incorporated as a standard instruction set in almost all recent Intel and AMD CPUs. Since the release of AES-NI, many high-speed cryptographic schemes have been proposed, such as AEGIS-128 [15], which has been selected in the final portfolio for high-performance applications in the CAESAR competition [1], and Tiaoxin-346 [12], which is the third-round candidate for CAESAR, for an Authenticated Encryption with Associated Data (AEAD), Haraka-v2 [10] for a hash function, and Pholkos [2], which was accepted to CT-RSA 2022, for a tweakable block cipher.

At FSE 2016, Jean and Nikolić showed how to construct an efficient round function based on sponge construction using AES-NI [8]. They offered several requirements for constructing efficient and secure AES-based round functions, for example, the ratio of the number of AES calls and inserted message blocks in each round and the 128-bit security against forgery attacks caused by internal collision. They proposed several classes of round functions that satisfied these requirements. Their constructions utilize only one round of AES (aesenc) and a 128-bit XOR operation.

Subsequently, Sakamoto et al. improved it regarding speed and state size. To minimize the critical path of one round, they removed the case of cascading both aesenc and XOR for one round [14]. By introducing a cost-free block permutation in the round function, they could search for candidates in a larger space without sacrificing performance. Consequently, they obtained more efficient constructions with a smaller state size than the Jean and Nikolić candidates. Based on this, Sakamoto et al. presented a new AEAD scheme named Rocca for Beyond 5G systems. The designers of Rocca set several speed and security requirements as criteria for a suitable round function for Beyond 5G environments and searched for round functions that met their requirements. However, because of the vast search space, they did not search all candidates; that is, they did not show anything about the optimally of the round function of Rocca.

1.2 Our Contribution

In this paper, we revisit the design strategy by Sakamoto et al. for finding efficient and secure round functions in a software environment where AES-NI is available. We add new requirements to further improve speed of Rocca. More specifically, we focus on *the number of required temporary registers to update the round function*, which is not fully considered by Sakamoto et al. The number of instructions to load and store data into SIMD registers is reduced by minimizing the number of temporary registers, which leads to a more efficient round function. Consequently, we present a class of round functions with only one required temporary

Manuscript received November 26, 2021.
Manuscript revised March 30, 2022.
Manuscript publicized July 7, 2022.

[†]The authors are with the Graduate School of Information Science, University of Hyogo, Kobe-shi, 650-0047 Japan.
[††]The authors are with the Graduate School of Applied Informatics, University of Hyogo, Kobe-shi, 650-0047 Japan.
[†††]The author is with National Institute of Information and Communications Technology, Koganei-shi, 184-8795 Japan.
[††††]The author is with PRESTO, Japan Science and Technology Agency, Kawaguchi-shi, 332-0012 Japan.
a) E-mail: nobuyuki.takeuchi0722@gmail.com

DOI: 10.1587/transfun.2021EAP1158

resister among the class of Rocca, while the round function of Rocca requires two temporary registers. Moreover, we restrict ourselves to searching only among the bijective round functions to reduce the search space. These new requirements enable us to search for the fastest among all the candidates in the same class as the round function of Rocca.

These round functions that are up to 14% faster than those of Rocca, are demonstrated on the latest Ice Lake and Tiger Lake architectures. We emphasize that, among the round functions in the same class as those of Rocca, ours are optimal regarding speed.

1.3 Organization

This paper is organized as follows. In Sect. 2, we describe AES-NI and the security evaluation against a forgery attack by internal collision which is used as the security criterion in this paper. We describe the design requirements of the round function of Jean and Nicolić [8] and Sakamoto et al. [14], and the round function of Rocca is explained in Sect. 3. We provide new representation of the round function of Rocca using matrices and present the new requirements in Sect. 4. In Sect. 5, we describe the search space and how to evaluate the lower bound for the number of active S-boxes. In Sect. 6, we present the search results and software benchmarks and compare our round functions with those of Rocca. Sect. 7 concludes this paper.

2. Preliminaries

This section describes AES-NI and the security evaluation of the round function on the sponge construction.

2.1 AES-NI

The AES New Instructions (AES-NI) [3], [6], announced by Intel in 2008 is a special set of SIMD instructions that aim to accelerate AES encryption and decryption. The AES-NI can execute AES approximately 10 times faster than non-AES-NI approaches for parallelizable modes such as CTR. In this study, we utilize only AESENC, which executes one round of AES (not the last round) with an input state S and a round key K:

$$\mathrm{AESENC}(S, K) = (\mathrm{MixColumns} \circ \mathrm{ShiftRows} \circ \mathrm{SubBytes}(S)) \oplus K.$$

The performance of AES-NI can be evaluated using *latency* and *throughput*. Latency is the number of clock cycles required to execute a single AES-NI instruction. Throughput is the number of clock cycles required to call the same instructions consecutively during pipeline processing. Latency and throughput depend on the CPU architectures. Figure 1 shows an example of AESENC parallel execution during pipeline processing on an Intel Ice Lake [13] series CPU where the latency and throughput respectively of AESENC are 3 and 0.5. Note the throughput of 0.5, which means that AESENC with a throughput of 1 is executed on two ports

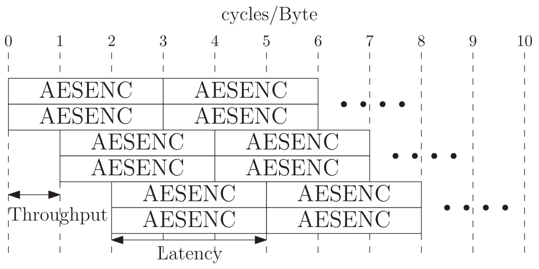


Fig. 1 The pipeline processing of AESENC on Intel Ice Lake.

simultaneously.

2.2 Security Evaluation of the Round Function Based on a Sponge Construction

Because we do not verify the optimality of an entire encryption scheme but only the round function based on sponge construction, we employ the same security criteria as Sakamoto et al. [14]. They searched for round functions that, when applied as a component in AEADs, guarantee 128-bit security against a forgery attack. In particular, they evaluated the security against a forgery attack by internal collision based on the differentially active S-box [9], [16]. We similarly evaluated the internal collision based on the differentially active S-box of the round functions.

2.2.1 Security Evaluation of Internal Collision by Differentially Active S-Box

Let $F_R(M) = f_R \circ f_{R-1} \circ \cdots \circ f_1(M)$ and M_i be the encryption function and the corresponding messages inserted into F_R where $i \in \{0, 1\}$, respectively. f_R denotes one round of the round function in F_R, where R is the round number. When the following equation is satisfied, there is an internal collision in the internal state on F_t:

$$f_t \circ f_{t-1} \circ \cdots \circ f_1(M_0) = f_t \circ f_{t-1} \circ \cdots \circ f_1(M_1).$$

This equation shows that there is an internal collision on a certain round of F_R when the differences in all states in the last round is 0, that is, $f_t \circ f_{t-1} \circ \cdots \circ f_1(\Delta M) = 0$, where $M_0 \oplus M_1 = \Delta M$. In round functions that we consider in this paper, the differences in all states in the first round are 0 because we assume the round functions that will be used as a component in AEADs.

In general, differential propagation can be probabilistic only when the differences pass a non-linear function. Because the round functions that we consider in this paper are based on the AES round function, we must regard only an S-box in AES as a non-linear function. Therefore, the differential probability decreases only when the differences pass an S-box. The S-box with a non-zero input difference is called an “active S-box”. Basically, when all S-boxes are independent of each other, we can estimate the differential probability of the entire round function by the product of the differential probability of all active S-boxes. We can apply this method to our round function because, for differential propagation, all S-boxes are independent of each other [8], [14]. Let DP_{F_R} and DP_s be the differential probabilities of the whole round function and S-box, respectively. We can calculate DP_{F_R} as follows.

$$DP_{F_R} = \prod_{i=1}^{n} DP_s,$$

where n is the number of active S-boxes in this differential characteristic, which indicates a certain differential propagation. DP_{F_R} is equivalent to the probability of an internal collision in a certain round of F_R.

When evaluating security against an internal collision on t rounds of F_R, the maximum differential probability must be evaluated that the differences in states at t rounds will all be 0. This can be calculated by searching for the differential characteristics with the minimum number of active S-boxes among all the differential characteristics. Conversely, the maximum differential probability of F_R, can be estimated by searching for the lower bound for the number of active S-boxes. Let $DP_{F_R max}$ and DP_{smax} be the maximum differential probabilities of the differential characteristics with the minimum number of active S-boxes on F_R and the S-box, respectively. $DP_{F_R max}$ can be calculated as follows.

$$DP_{F_k max} = \prod_{i=1}^{m} DP_{smax},$$

where m is the lower bound of the number of active S-boxes.

In general, there are two methods to guarantee a lower bound for the number of active S-boxes. One is to guarantee it by mathematical proofs, and the other is to evaluate it using the specific algorithm, such as Mixed-Integer Linear Programming (MILP) and Satisfiability (SAT) solvers. In this study, we calculate the lower bound for the number of active S-boxes using the latter method with an MILP.

2.2.2 Evaluation of the Lower Bound for the Number of Active S-Boxes by Mixed-Integer Linear Programming

Mixed-Integer Linear Programming (MILP) is an efficient solver for finding variables to maximize or minimize a particular objective function on some constraints, both of which are expressed by a linear inequality. At Inscrypt 2011, Mouha et al. presented a method for calculating the lower bound for the number of active S-boxes with an MILP [11]. To construct an MILP model for evaluation, their method expresses all operations in a cryptographic scheme as linear inequalities and assigns them to an MILP model as constraints. Then, the total number of active S-boxes is assigned to the MILP model as the objective function. We obtain the lower bound for the number of active S-boxes to minimize it. In this study, we used the Gurobi Optimizer [7] as the MILP solver and evaluated the lower bound for the number of differentially active S-boxes using the same method as Mouha et al.

3. Previous Work

In this section, we describe two previous works by Jean and Nicolić [8] and Sakamoto et al. [14], both of which focus on the construction of the round function based on sponge construction for AEADs.

Before beginning to explain their work, let $\#state$, $\#AESENC$, and $\#message$ respectively be the number of states, AESENCs, and inserted messages in a single round. Hereafter, we use the above notations for all round functions.

3.1 Requirements for the Round Functions of Jean and Nicolić

Jean and Nicolić studied the round function based on a sponge construction consisting only of AESENC and XOR for speed and security. Figure 2 shows the general structure of the round function studied by Jean and Nicolić in [8], where A means AESENC shown in Sect. 2.1.

They defined the following "*rate*" as an important parameter for estimating the speed of the round function.

Definition 1(*Rate*). *Rate is the required number of AESENC calls to encrypt a 128-bit message. The rate is expressed by the following equation,*

$$rate = \#AESENC \;/\; \#message.$$

Because a smaller *rate* leads to a more efficient round function, for speed, a round function with a low *rate* is preferable.

Moreover, Jean and Nicolić showed that the number of AESENCs of $\geq (latency/throughput)$ in a single round must be used for the most efficient execution of AESENC in pipeline processing. As an example, we consider the case of running the round functions with $\#AESENC = 6$ and 3 on Intel Ice Lake (latency: 3, throughput: 0.5) [13]. The processes of AESENC for $\#AESENC = 6$ and 3 are shown in Figs. 3 and 4, respectively. Figure 3 shows that AESENC is executed without any waiting cycles except for the throughput when $\#AESENC$ in a single round is 6 or more. Conversely, Fig. 4 shows that AESENC needs to wait for some cycles other than throughput when $\#AESENC$ is less than 6.

Jean and Nicolić searched for round functions that satisfy

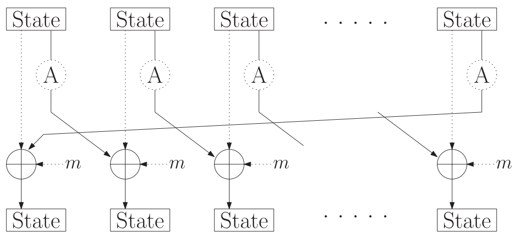


Fig. 2 The general construction of Jean and Nicolić.

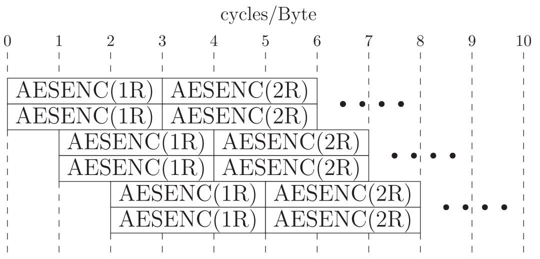


Fig. 3 The pipeline process of $\#AESENC = 6$ on Intel Ice Lake.

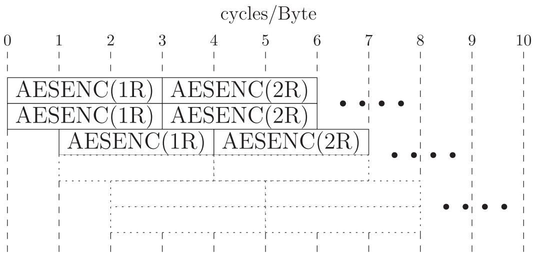


Fig. 4 The pipeline process of $\#AESENC = 3$ on Intel Ice Lake.

these requirements and guarantee 128-bit security against a forgery attack by internal collision.

The requirements of Jean et al. are summarized as follows.

Requirement 1. *rate* is as small as possible.

Requirement 2. $\#AESENC \geq$(latency)/(throughput) in a single round.

Requirement 3. 128-bit security against forgery attacks by internal collisions.

As a result, they showed an efficient round function with $rate = 2$ and $\#state = 12$.

3.2 Requirements for Round Functions of Sakamoto et al.

Sakamoto et al. attempted to find a more efficient round function than that of Jean and Nicolić to design an AEAD to meet the requirement of Beyond 5G. Their method is similar to that of Jean and Nicolić, but they added a new requirement, which concerns $\#state$, and improved the general structure of Jean and Nicolić as follows:

How to apply AESENC and XOR When both AESENC and XOR are applied consecutively, the critical path for a round function becomes worse than when only AESENC or XOR is applied. While Jean and Nicolić did not consider this case, Sakamoto et al. removed this case from their general structure. This means that, regarding the critical path, they considered the class of optimal round functions based on only AESENC and XOR.

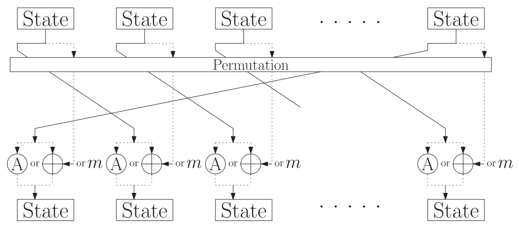


Fig. 5 The general construction of Sakamoto et al.

Table 1 The class of round functions that Sakamoto et al. searched where a bold style shows the class including the round function of Rocca.

class	$\#AESENC$	$\#state$	$\#message$
class-1	4	6	2
class-2	4	7	2
class-3	**4**	**8**	**2**
class-4	3	6	2
class-5	3	7	2
class-6	3	8	2

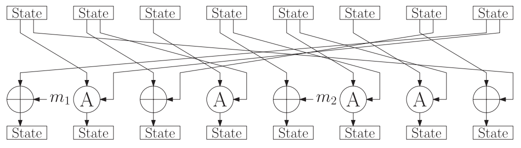


Fig. 6 the round function of Rocca.

Apply Permutation Sakamoto et al. applied a state-wise permutation before applying AESENC and XOR to their general structure. This provides stronger resistance against a forgery attack without an additional cost.

Figure 5 shows the general structure of Sakamoto et al. In their work, they searched for round functions that are suitable for execution on Intel Ice Lake architecture, namely, they targeted the round functions whose AESENCs of ≥ 6 in a single round. Their requirements are as follows.

Requirement 1. *rate* is as small as possible.

Requirement 2. $\#AESENC \geq 6$, in a single round.

Requirement 3. 128-bit security against forgery attacks by internal collisions.

Requirement 4. $\#state$ is around 7.

3.3 Round Function of Rocca and Search Method

Based on the method described in Sect. 3.2, Sakamoto et al. searched a more efficient round function than that of Jean and Nicolić. Table 1 shows the class of the round functions that Sakamoto et al. searched. As a result, they found it in class-3 in Table 1 and employed it as the round function of Rocca. The round function of Rocca is shown in Fig. 6. This round function does not satisfy requirement 2, but Rocca applies

AESENC to each ciphertext block, and the round function of Rocca generates 2 ciphertext blocks per a round function update. Hence, this round function satisfies requirement 2 because the total number of AESENCs in a single round is $4 + 2 = 6$.

4. New Speed Requirements for an Efficient Round Function

In this section, we describe adding new requirements to improve the efficiency of the round function and reduce the search space. First, the structure of the round function of Sakamoto et al. [14] is generalized as a matrix expression. Then, we show the new requirements for finding more efficient round functions effectively. Finally, the conditions on the generalized matrix to meet these new requirements are described.

4.1 Generalization of the Round Function

Let R, X, Y, and M be the round function, the input state, the state after updating the round function, and the message, respectively. The round function shown in Fig. 5 can be expressed as follows:

$$R \cdot X \oplus M = Y, \tag{1}$$

where R is the $s \times s$ matrix and X, Y, and M are the column vectors with s entries. Furthermore, R can be decomposed into C and P, both of which are $s \times s$ matrices, as follows:

$$R = C \oplus P \tag{2}$$

C represents the positions where XOR and AESENC are applied in Fig. 5. Therefore, the $(i+1, i)$th and $(1, s)$th entry in C, where $1 \leq i \leq s-1$ is 1 or A, which denotes AESENC, and the other entries are all zero. Note that the product of A and x_i denotes $A(x_i)$, where $A(x_i) = MixColumns \circ ShiftRows \circ SubBytes(x_i)$.

P represents a state-wise permutation applied before AESENC and XOR. Let $p_{i,j}$ be the (i, j)th entry in P. P satisfies the following conditions:

- $p_{i,j} \in \{0, 1\}$
- $\sum_{i=1}^{s} p_{i,j} \leq 1$
- $\sum_{j=1}^{s} p_{i,j} \leq 1$
- $\sum_{i=1}^{s} \sum_{j=1}^{s} p_{i,j} = s - \#message$

Let m_i be the ith entry in M. M satisfies the following conditions:

$$m_i = \begin{cases} m_{pos} & if\ \sum_{j=1}^{s} p_{i,j} = 1 \\ 0 & otherwise, \end{cases}$$

where m_{pos} denotes the position of the inserted messages.

(1) Round Function of Rocca

For a better understanding, we present the round function of Rocca with Eqs. (1) and (2). Let x_i and y_i be the ith entries in X and Y, respectively. The round function of Rocca can be expressed as follows:

$$\begin{bmatrix} 0&0&0&0&0&0&0&1 \\ A&0&0&0&0&0&0&1 \\ 0&1&0&0&0&0&1&0 \\ 0&1&A&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&1&A&0&0&0 \\ 0&0&0&0&1&A&0&0 \\ 1&0&0&0&0&0&1&0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} \oplus \begin{bmatrix} m_{pos} \\ 0 \\ 0 \\ 0 \\ m_{pos} \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \end{bmatrix}.$$

With Eq. (2), R can be represented by the following matrix:

$$R = \begin{bmatrix} 0&0&0&0&0&0&0&1 \\ A&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&A&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&A&0&0&0 \\ 0&0&0&0&0&A&0&0 \\ 0&0&0&0&0&0&1&0 \end{bmatrix} \oplus \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 1&0&0&0&0&0&0&0 \end{bmatrix}.$$

4.2 Our New Requirements

In this section, we describe our new requirements to find a more efficient round function and reduce the search space.

4.2.1 Requirement for Reducing the Search Space

As the general requirement, a round function in AEADs must be bijective. However, Sakamoto et al. also searched non-bijective round functions. In our search, a new requirement was added that avoids searching for unnecessary space based on the following proposition.

Proposition 1. *A round function is bijective if and only if R is non-singular with $A(x_i) \oplus x_i = 0$.*

Proof. In a real number, non-singular matrix permutes the input to the output one-to-one as such it is a bijection. Because all of our matrix entries are in GF(2), we also need to consider the case where all input entries with 0 and 1 lead to the same output even if the matrix is non-singular. This case occurs only when the sum of the non-zero entries in each row of R is all even numbers. However, this situation never occurs in R under the conditions of P and C. Hence, a round function is bijective if and only if R is non-singular with $A(x_i) \oplus x_i = 0$.□

For Proposition 1, we add the following requirement to reduce the search space.

Requirements 5. R is non-singular, with $A(x_i) \oplus x_i = 0$.

4.2.2 Requirement for More Efficient Round Function

To achieve faster execution (cycles/Byte), it is desirable to minimize the total number of registers to update the round function. The execution speed mainly depends on the *rate*, but the number of required resisters also affects the speed. In fact, Sakamoto et al. tried to minimize it in Requirement 4. However, there is still room for improvement regarding the total number of required registers.

Because the number of required temporary registers varies in the applied permutations, the total number of required registers to update the round function depends on $\#state$ and the applied permutation. A lower number of required resisters can lead to a faster round function with the same *rate* as the total number of instructions for updating the round function decreases.

As mentioned, Sakamoto et al. considered $\#state$ in requirement 4. Our search also focuses on the number of required temporary registers for the permutation and minimizes it, which is not considered by Sakamoto et al.

Let $\#tmp$ be the number of required registers to update the round function. We add the following new requirement.

Requirement 6. $\#tmp$ is 1.

4.2.3 Conditions for New Requirements

We show the conditions to satisfy requirements 5 and 6 with matrices described in Sect. 4.1. Note that we assume that the update order of state starts from y_s to y_1 in matrix Y, where s is $\#state$.

Condition for requirement 5. Matrix C is always non-singular because the column vectors in C are linearly independent of each other. Therefore, whether the matrix R is non-singular depends entirely on the matrix P. When the matrix P satisfies the following condition, the matrix R is always non-singular.

- the $\{(i \bmod s) + 1\}$th entry in the ith column vector of P is zero, where $1 \le i \le s$.

Condition for requirement 6. Similar to the case in requirement 5, $\#tmp$ depends entirely on the matrix P because the positions of the non-zero entries are all fixed in the matrix C. As the first step of the round function update, state x_s is saved in the temporary register. Therefore, we can no longer use an additional temporary register to save a state. To update the remaining states without the additional temporary register, the state must only be updated with the remaining states (non-updated states) and x_s in the temporary register. Hence, when the matrix P satisfies the following condition, $\#tmp$ is always one.

- the ith entry in the jth column vector of P is zero, where $1 \le i \le (j-1)$ and $2 \le j \le (s-1)$.

For a better understanding, we provide an example of the matrix P that satisfies the conditions for requirements 5 and 6 as follows:

$$P = \begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 & 0 & 0 & a \\ a & 0 & a & 0 & 0 & 0 & 0 & a \\ a & a & 0 & a & 0 & 0 & 0 & a \\ a & a & a & 0 & a & 0 & 0 & a \\ a & a & a & a & 0 & a & 0 & a \\ a & a & a & a & a & 0 & a & a \\ a & a & a & a & a & a & 0 & a \end{bmatrix},$$

where a denotes the element that can be 0 or 1 under the condition of P described in Sect. 4.1.

5. How to Search for Round Functions

In this section, we describe how to search for the round functions that have resistance to internal collision and reveal the search space.

5.1 How to Find the Lower Bound for the Number of Differentially Active S-Boxes

We evaluate the resistance against a forgery attack based on an internal collision. As described in Sect. 2.2, our evaluation was based on a differentially active S-box. Since Rocca claims 128-bit security against forgery attacks, the maximum probability of an S-box is 2^{-6}, it is secure if there are ≥ 22 active S-boxes, as it gives $2^{(-6\times 22)} < 2^{-128}$ as an estimate of differential probability [14]. On a block cipher, the lower bound for the number of differentially active S-boxes in a certain round always outnumbers that of the previous rounds. However, the transition of the lower bound for the number of differentially active S-boxes in our evaluation does not follow this situation because the difference of each state in the first and last rounds is fixed to zero, as the differences are injected through the messages. On the contrary, it drops as the number of rounds increases. Moreover, the number of active S-boxes of ≥ 22 over any round must be guaranteed because our round function will be used as a component in AEADs.

Therefore, we set the following criterion to determine the lower bound for the number of active S-boxes in our evaluation.

Lower bound for the number of active S-boxes When the lower bound for the number of active S-boxes is the same over 3 rounds for more than 15 rounds, we regard it as the lower bound for the number of active S-boxes in a round function.

Based on the above criterion, we search for round functions that satisfy requirement 3.

5.2 Search Space

We evaluate the resistance against an internal collision of all the round functions that satisfy requirements 1, 2, 4, 5, and 6 using an MILP. In particular, we search for the same class as the round function of Rocca, that is, $\#state = 8$,

Table 2 Search spaces of the class with $\#state = 8$, $\#AESENC = 4$, and $\#message = 2$.

Requirement	# total candidates	# searched candidates	Reference
1, 2, 4	79027200 ($\approx 2^{26.23}$)	1000000	[14]
1, 2, 4, 5, 6	553140 ($\approx 2^{19.07}$)	All	Our

$\#AESENC = 4$, and $\#message = 2$. Table 2 shows the search spaces of the method of Sakamoto et al. [14] and ours. Compared to the method of Sakamoto et al., the proposed method reduces the total number of candidates by approximately $2^{26.22}$. Thanks to this reduction, we can evaluate all candidates, while Sakamoto et al. were unable to evaluate all candidates.

6. Search Results and Implementation

In this section, we first present the results of our search used in Sect. 5. Then, we show the software implementation results of our round functions and other round functions in [8], [12], [14], [15]. Finally, we briefly compare our round functions with that of Rocca regarding performance and security.

6.1 Search Results

Table 3 shows the total number of the round functions that guarantee the number of differentially active S-boxes of ≥ 22. Of 553140 candidates, 56 round functions were found that satisfy all requirements. Besides, we also found that these 56 round functions include "equivalent round functions". Equivalent round functions mean the identical round functions except that the inserted message positions m_1 and m_2 are reversed. Since it is not necessary to distinguish between m_1 and m_2, we call these round functions as equivalent round functions. After eliminating the equivalent round functions, we eventually obtained 28 round functions. This evaluation required two days on three computers equipped with Intel(R) Xeon(R) Platinum 8260 (24-Core)×2 with 256 GB RAM and two AMD Ryzen Threadripper 3990X (64-Core) with 256 GB RAM.

6.2 Software Evaluation

We evaluated the performance of some round functions and show that ours are faster than other round functions given in [8], [12], [14], [15]. Table 4 shows software evaluation result. These round functions are all implemented using SIMD instructions, and the results are the average of 1000000 iterations with a message length of 96 KB on Intel(R) Core(TM) i7-1068NG7 @ 2.30 GHz (Ice Lake) and Intel(R) Core(TM) i7-11375H @ 3.30 GHz (Tiger Lake). Structure-1 and Structure-2 are the fastest round functions that we found on Ice Lake and Tiger Lake, respectively. Structure-1 and Structure-2 are shown in Figs. 7 and 8, respectively.

6.3 Comparison with the Round Function of Rocca

Our round functions, Structure-1 and Structure-2, are compared for performance and security with one of Rocca. Regarding performance, on Ice Lake and Tiger Lake, Structure-1 and Structure-2 respectively are 7% and 14% faster than those of Rocca. We present a pseudocode of Structure-2 in Algorithm 1 to show that it can be implemented with $\#tmp = 1$. In Algorithm 1, let S and K be an input state and a round key, $AES(S, K)$ is defined as follows:

$$\mathrm{AES}(S, K) = (\mathrm{MixColumns} \circ \mathrm{ShiftRows} \circ \mathrm{SubBytes}(S)) \oplus K.$$

To discuss security, in Table 5, we show the lower bound for the number of active S-boxes in Structure-1, Structure-2, and round function of Rocca. Structure-1 and Structure-2 have less number of active S-boxes than round function of Rocca. However, this will never lead to security problems because the number of active S-boxes of ≥ 22 is sufficient to guarantee 128-bit security against a forgery attack. Therefore,

Table 4 Execution speed of the round functions.

Structure	Speed (cycles / Byte)	
	Ice Lake	Tiger Lake
AEGIS-128L [15]	0.188491	0.183929
Tiaoxin-346 [12]	0.189715	0.203112
Jean and Nicolić [8]	0.137449	0.133389
Rocca [14]	0.123663	0.121379
Structure-1 (Fig. 7)	**0.114738**	**0.117580**
Structure-2 (Fig. 8)	**0.121626**	**0.104886**

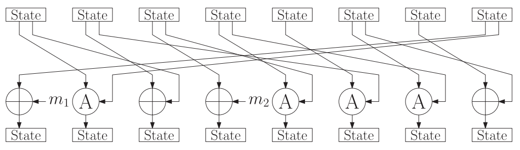


Fig. 7 Round function of Structure-1.

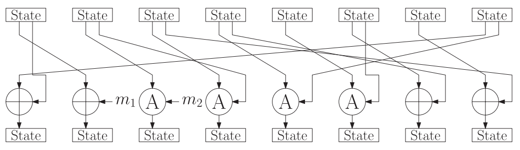


Fig. 8 Round function of Structure-2.

Table 3 The total number of the round functions that guarantee the number of differentially active S-boxes of ≥ 22.

#Rounds	7R	8R	9R	10R	11R	12R	13R	14R	15R	16R	17R	18R
#Structures	722	236	148	98	78	64	58	58	56	56	56	56

Table 5 The lower bound of the number of differentially active S-boxes for each structure.

#Rounds	1R	2R	3R	4R	5R	6R	7R	8R	9R	10R	11R	12R	13R	14R	15R
Rocca	-	-	-	36	25	25	25	**24**	24	24	24	24	24	24	24
Structure-1 (Fig. 7)	-	-	-	30	30	**22**	22	22	22	22	22	22	22	22	22
Structure-2 (Fig. 8)	-	-	-	-	32	23	23	**22**	22	22	22	22	22	22	22

Algorithm 1 Implementation of structure-2.

```
Input: message1, message2
1: tmp ← S[7]
2: S[7] ← S[6] ⊕ S[2]
3: S[6] ← S[5] ⊕ S[3]
4: S[5] ← AES(S[4], S[5])
5: S[4] ← AES(S[3], tmp)
6: S[3] ← AES(S[2], S[1])
7: S[2] ← AES(S[1], message2)
8: S[1] ← S[0] ⊕ message1
9: S[0] ← tmp ⊕ S[0]
```

our round functions achieve higher speed while maintaining security identical to that of Rocca.

In Table 4, Structure-1 is faster than Structure-2 on Ice Lake while Structure-2 is faster than Structure-1 on Tiger Lake. Our results have different behavior on Ice Lake and Tiger Lake even though these architectures are identical in terms of latency and throughput of AES-NI. However, these are different architectures, and we think that other factors might affect the performance. Unfortunately, Intel has not published a detailed white paper on AES-NI for each architecture. Actually, latency and throughput of AES-NI on Ice Lake and Tiger Lake are provided by not Intel but other research institution. For the above reason, we concluded that it is difficult to look more deeply into the relationship between these architectures, and it is our future work.

7. Conclusion

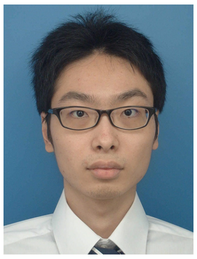
In this study, we verified the optimality of the round function of Rocca. By minimizing the number of required registers to update the round function and evaluating only the bijective round functions, we found faster round functions than those of Rocca and evaluated all candidates. We emphasize that, among the class of the round function of Rocca, ours are optimal for speed.

Acknowledgments

Kosei Sakamoto is supported by a Grant-in-Aid for JSPS Fellows (KAKENHI 20J23526) from the Japan Society for the Promotion of Science. Takanori Isobe is supported by JST, PRESTO Grant Number JPMJPR2031, and Grant-in-Aid for Scientific Research (B) (KAKENHI 19H02141) from the Japan Society for the Promotion of Science. This study is part of the results of "Research and development on new generation cryptography for secure wireless communication services", which is supported by the Ministry of Internal Affairs and Communications of Japan as part of the research program "Research and Development for Expansion of Radio Wave Resources (JPJ000254)".

References

[1] CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness, https://competitions.cr.yp.to/caesar.html, 2018.
[2] J. Bossert, E. List, S. Lucks, and S. Schmitz, "Pholkos – Efficient large-state tweakable block ciphers from the AES round function," IACR Cryptol. ePrint Arch., page 275, 2020.
[3] Intel Corporation, Intel intrinsics guide, Official webpage, https://software.intel.com/sites/landingpage/IntrinsicsGuide/
[4] The ZUC design team, The zuc-256 stream cipher, http://www.is.cas.cn/ztzl2016/zouchongzhi/201801/W020180126529970733243.pdf, 2018.
[5] P. Ekdahl, T. Johansson, A. Maximov, and J. Yang, "A new SNOW stream cipher called SNOW-V," IACR Trans. Symmetric Cryptol., vol.2019, no.3, pp.1–42, 2019.
[6] S. Gueron, Intel Advanced Encryption Standard (AES) New Instructions Set, 2010.
[7] Gurobi Optimization Inc., Gurobi optimizer 6.5, Official webpage, http://www.gurobi.com/, 2015.
[8] J. Jean and I. Nikolic, "Efficient design strategies based on the AES round function," Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, T. Peyrin, ed., volume 9783 of Lecture Notes in Computer Science, pp.334–353, Springer, 2016.
[9] D. Khovratovich and C. Rechberger, "The LOCAL attack: Cryptanalysis of the authenticated encryption scheme ALE," Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, T. Lange, K.E. Lauter, and P. Lisonek, eds., volume 8282 of Lecture Notes in Computer Science, pp.174–184, Springer, 2013.
[10] S. Kölbl, M. M. Lauridsen, F. Mendel, and C. Rechberger, "Haraka v2 – Efficient short-input hashing for post-quantum applications," IACR Trans. Symmetric Cryptol., vol.2016, no.2, pp.1–29, 2016.
[11] N. Mouha, Q. Wang, D. Gu, and B. Preneel, "Differential and linear cryptanalysis using mixed-integer linear programming," Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, C. Wu, M. Yung, and D. Lin, eds., volume 7537 of Lecture Notes in Computer Science, pp.57–76, Springer, 2011.
[12] I. Nikolic, Tiaoxin-346, Submission to the CAESAR competition, 2014.
[13] Real-Time and Embedded Sys Lab. uops.info. Official webpage, https://www.uops.info/
[14] K. Sakamoto, F. Liu, Y. Nakano, S. Kiyomoto, and T. Isobe, "Rocca: An efficient AES-based encryption scheme for beyond 5G," IACR Trans. Symmetric Cryptol., vol.2021, np.2, pp.1–30, 2021.
[15] H. Wu and B. Preneel, "AEGIS: A fast authenticated encryption algorithm," Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, T. Lange, K.E. Lauter, and P. Lisonek, ed., volume 8282 of Lecture Notes in Computer Science, pp.185–201, Springer, 2013.
[16] S. Wu, H. Wu, T. Huang, M. Wang, and W. Wu, "Leaked-state-forgery attack against the authenticated encryption algorithm ALE," Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, K. Sako and P. Sarkar, eds., volume 8269 of Lecture Notes in Computer Science, pp.377–404, Springer, 2013.



Nobuyuki Takeuchi received the B.E. degree from University of Hyogo, Japan, in 2021. He is currently a M.E. student at University of Hyogo, Japan. His research interest is cryptography.

Kosei Sakamoto received the B.E. degree from Kansai University, Japan, in 2017. In 2020, he received the M.E. degree from University of Hyogo. He is currently a Ph.D. student at University of Hyogo, Japan. His research interest is cryptography.

Takanori Isobe received the B.E., M.E., and Ph.D. degrees from Kobe University, Japan, in 2006, 2008, and 2013, respectively. From 2008 to 2017, he worked at the Sony Corporation. Since 2017, he has been an Associate Professor at University of Hyogo. His current research interests include information security and cryptography.