Journal of Information Processing Vol.21 No.4 599–606 (Oct. 2013)

[DOI: 10.2197/ipsjjip.21.599]

**Regular Paper**

# Slide Property of RAKAPOSHI and Its Application to Key Recovery Attack

Takanori Isobe[1,a)] Toshihiro Ohigashi[2] Masakatu Morii[1]

Received: November 29, 2012, Accepted: June 14, 2013

***Abstract:*** This paper gives a first security evaluation of a lightweight stream cipher RAKAPOSHI. In particular, we analyze a slide property of RAKAPOSHI such that two different Key-IV pairs generate the same keystream but $n$-bit shifted. To begin with, we demonstrate that any Key-IV pair has a corresponding *slide* Key-IV pair that generates an $n$-bit shifted keystream with a probability of $2^{-2n}$. In order to experimentally support our results, some examples of such pairs are given. Then, we show that this property is able to be converted into key recovery attacks on RAKAPOSHI. In the related-key setting, our attack based on the slide property can recover a 128-bit key with a time complexity of $2^{41}$ and $2^{38}$ chosen IVs. Moreover, by using a variant of slide property called partial slide pair, this attack is further improved, and then a 128-bit key can be recovered with a time complexity of $2^{33}$ and $2^{30}$ chosen IVs. Finally, we present a method for speeding up the brute force attack by a factor of 2 in the single key setting.

***Keywords:*** stream cipher, slide attack, related-key attack, RAKAPOSHI, initialization process, partial slide pair

## 1. Introduction

With the recent large deployment of low resource devices such as RFID tags and sensor nodes, the demand for security in resource-constrained environments has been dramatically increased. As a result, design and analysis of lightweight ciphers has received a lot of attention from the cryptographic community. A number of lightweight primitives are proposed, e.g., block ciphers: PRESENT [6], KATAN [8], LED [15] and Piccolo [25], and hash functions: Quark [3], PHOTON [14] and SPONGENT [5]. As for lightweight stream ciphers, Grain v1 [17], Trivium [7] and MICKY2.0 [4] are selected by the eSTREAM project for hardware applications with highly restricted resources [13]. In spite of considerable efforts in a multi-years project, it is still debatable that design and analysis of stream ciphers are mature enough. Indeed, after the end of this project in 2008, F-FCSR-H [2], which is initial selected in the final portfolio, and the 128-bit version of Grain were broken [12], [16].

Cid, Kiyomoto, and Kurihara proposed a lightweight stream cipher RAKAPOSHI [10] after the eSTREAM project. RAKAPOSHI is a hardware-oriented stream cipher accepting a 128-bit key and a 192-bit IV, and employs a bit-oriented Dynamic Linear Feedback Shift Register. This structure is also adopted in K2 v2.0 [20], which is recently selected in ISO standard stream ciphers [1]. RAKAPOSHI is considered as a variant of the K2 v2.0 for the low-cost hardware implementation. Its performance properties in the hardware are comparable to stream ciphers selected in the eSTREAM, e.g., the circuit size of RAKAPOSHI is estimated as about 3K gates. In addition, RAKAPOSHI can provide a 128 bit security while Grain and Trivium have only an 80-bit security. Thus, designers claim that RAKAPOSHI can complement the eSTREAM portfolio, and increase the choice of secure lightweight stream ciphers. Despite its notable features of design and implementations, there exist only designers' self evaluations, i.e., no external cryptanalysis has been published so far.

In this paper, we give a first security evaluation of the lightweight stream cipher RAKAPOSHI [*1]. In particular, we deeply analyze a slide property of RAKAPOSHI such that two different Key-IV pairs generate the same keystream but $n$-bit shifted. This property mainly exploits a weakness of an initialization algorithm, and has been applied to Grain v1 stream cipher [9], [21]. To begin with, by exploiting the self-similarity of the initialization algorithm of RAKAPOSHI, we show that any Key-IV pair has a corresponding *slide* Key-IV pair that generates an $n$-bit shifted keystream with a probability of $2^{-2n}$. For $n = 1$, a Key-IV pair has a corresponding Key-IV pair that generates only a 1-bit shifted keystream with a probability of $2^{-2}$. Besides, we introduce a variant of the slide property, which is called *partial slide property*, that occurs with a higher probability than the basic slide property. Then we utilize these properties in order to construct related-key attacks on RAKAPOSHI. Our attack using the slide property can recover a 128-bit key with a time complexity of $2^{41}$ and $2^{38}$ chosen IVs. Moreover, by using the partial slide property, this attack is further improved, and then a 128-bit key can be recovered with a time complexity of $2^{33}$ and $2^{30}$ chosen IVs. This result reveals that RAKAPOSHI is practically vulnerable to the related-key attack based on the slide property. Finally, using this variant of the slide property, we give a method for speeding up the brute force attack in the single key setting by

[1] Kobe University, Kobe, Hyogo 657–8511, Japan
[2] Hiroshima University, Higashi-Hiroshima, Hiroshima 739–8511, Japan
[a)] Takanori.Isobe@jp.sony.com

[*1] This is the full version of the IWSEC 2012 paper [19]. After the publication of Ref. [19], two results similar to our results appeared [11], [22].

a factor of 2.

This paper is organized as follows. Brief descriptions of RAKAPOSHI are given in Section 2. In Section 3, we introduce a slide property of stream ciphers, and we analyze a slide property of RAKAPOSHI stream cipher in Section 4. Then, related-key attacks and a method for speeding up a keysearch on RAKAPOSHI are given in Sections 5 and 6, respectively. Finally, we conclude in Section 7.

## 2. Description of RAKAPOSHI

RAKAPOSHI is a stream cipher supporting a 128-bit key and a 192-bit IV. At time $t$, RAKAPOSHI consists of a 128-bit Nonlinear Feedback Shift Register (NFSR) : $A^t = \{a_t, a_{t+1}, \ldots, a_{t+127}\}$ ($a_t \in \{0,1\}$), a 192-bit Linear Feedback Shift Register (LFSR) : $B^t = \{b_t, b_{t+1}, \ldots, b_{t+191}\}$ ($b_t \in \{0,1\}$) and an 8-to-1 nonlinear function $v$ (see **Fig. 1**). Since RAKAPOSHI employs the bit-oriented Dynamic Linear Feedback Shift Register (DLFSR), two bits of the registers $A$ are used for dynamically updating the feedback function of the register $B$.

The NFSR $A^t$ and the LFSR $B^t$ are updated as follows:

$$\begin{aligned}
a_{t+128} &= g(a_t, a_{t+6}, a_{t+7}, a_{t+11}, a_{t+16}, a_{t+28}, a_{t+36}, a_{t+45}, a_{t+55}, a_{t+62}) \\
&= 1 \oplus a_t \oplus a_{t+6} \oplus a_{t+7} \oplus a_{t+11} \oplus a_{t+16} \oplus a_{t+28} \oplus a_{t+36} \\
&\quad \oplus a_{t+45} \oplus a_{t+55} \oplus a_{t+62} \oplus a_{t+7}a_{t+45} \oplus a_{t+11}a_{t+55} \\
&\quad \oplus a_{t+7}a_{t+28} \oplus a_{t+28}a_{t+55} \oplus a_{t+6}a_{t+45}a_{t+62} \oplus a_{t+6}a_{t+11}a_{t+62}, \\
b_{t+192} &= f(b_t, b_{t+14}, b_{t+37}, b_{t+41}, b_{t+49}, b_{t+51}, b_{t+93}, b_{t+107}, b_{t+120}, \\
&\quad b_{t+134}, b_{t+136}, b_{t+155}, b_{t+158}, b_{t+176}, c_0, c_1) \\
&= b_t \oplus b_{t+14} \oplus b_{t+37} \oplus b_{t+41} \oplus b_{t+49} \oplus b_{t+51} \oplus b_{t+93} \\
&\quad \oplus \overline{c_0} \cdot \overline{c_1} \cdot b_{t+107} \oplus \overline{c_0} \cdot c_1 \cdot b_{t+120} \oplus c_0 \cdot \overline{c_1} \cdot b_{t+134} \\
&\quad \oplus c_0 \cdot c_1 \cdot b_{t+136} \oplus \overline{c_0} \cdot b_{t+155} \oplus c_0 \cdot b_{t+158} \oplus b_{t+176},
\end{aligned}$$

where $\oplus$ is a bit-wise XOR, $\overline{x}$ is complement of $x$, and $c_0$ and $c_1$ are $a_{t+41}$ and $a_{t+89}$, respectively. The 8-to-1 nonlinear function $v$ is expressed as

$$s_t = v(a_{t+67}, a_{t+127}, b_{t+23}, b_{t+53}, b_{t+77}, b_{t+81}, b_{t+103}, b_{t+128}).$$

The non-linear function $v$ is given as follows.

$$\begin{aligned}
&v(x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7) = \\
&x_0x_1x_2x_3x_4x_5x_6 + x_0x_1x_2x_3x_4x_5 + x_0x_1x_2x_3x_4x_6 + \\
&x_0x_1x_2x_3x_5x_6x_7 + x_0x_1x_2x_3x_5x_6 + x_0x_1x_2x_3x_5x_7 + \\
&x_0x_1x_2x_3x_5 + x_0x_1x_2x_3x_6x_7 + x_0x_1x_2x_4x_5x_6 + \\
&x_0x_1x_2x_4 + x_0x_1x_2x_5x_6 + x_0x_1x_2x_5x_7 + x_0x_1x_2x_7 + \\
&x_0x_1x_2 + x_0x_1x_3x_4x_5x_6x_7 + x_0x_1x_3x_4x_5x_7 + x_0x_1x_3x_4x_5 + \\
&x_0x_1x_3x_4x_7 + x_0x_1x_3x_4 + x_0x_1x_3x_6 + x_0x_1x_4x_5x_6x_7 + \\
&x_0x_1x_4x_5x_6 + x_0x_1x_4x_5x_7 + x_0x_1x_4x_6x_7 + x_0x_1x_4x_7 + \\
&x_0x_1x_5x_6x_7 + x_0x_1x_5x_6 + x_0x_1x_5 + x_0x_1x_6 + x_0x_1 + \\
&x_0x_2x_3x_4x_5x_6 + x_0x_2x_3x_4x_5x_7 + x_0x_2x_3x_4 + x_0x_2x_3x_5x_6x_7 + \\
&x_0x_2x_3x_5x_6 + x_0x_2x_3x_5x_7 + x_0x_2x_3x_6 + x_0x_2x_4x_5x_6x_7 + \\
&x_0x_2x_5x_6 + x_0x_2x_5 + x_0x_2x_6x_7 + x_0x_2x_7 + x_0x_3x_4x_5x_6x_7 + \\
&x_0x_3x_4x_5x_6 + x_0x_3x_4x_5x_7 + x_0x_3x_4x_5 + x_0x_3x_4x_7 + \\
&x_0x_3x_5x_6x_7 + x_0x_3x_5 + x_0x_3x_6 + x_0x_3 + x_0x_4x_5x_6 + \\
&x_0x_4x_6x_7 + x_0x_5x_6 + x_0x_6 + x_0 + x_1x_2x_3x_4 + x_1x_2x_3x_5x_6 + \\
&x_1x_2x_3x_5x_7 + x_1x_2x_3x_5 + x_1x_2x_3 + x_1x_2x_4x_5x_6 + x_1x_2x_4x_6 + \\
&x_1x_2x_4 + x_1x_2x_5 + x_1x_2 + x_1x_3x_4x_5x_6x_7 + x_1x_3x_4x_5x_7 + \\
&x_1x_3x_4x_6x_7 + x_1x_3x_4x_6 + x_1x_3x_4 + x_1x_3x_5x_6 + x_1x_3x_5 + \\
&x_1x_3x_6 + x_1x_3x_7 + x_1x_4x_5x_6x_7 + x_1x_4x_5x_7 + x_1x_5x_6 + \\
&x_1x_5x_7 + x_1x_5 + x_1x_6x_7 + x_1x_6 + x_1 + x_2x_3x_4x_5x_6 + \\
&x_2x_3x_4x_5x_7 + x_2x_3x_4x_5 + x_2x_3x_4x_6x_7 + x_2x_3x_4 + x_2x_3x_5x_7 + \\
&x_2x_3x_6x_7 + x_2x_3x_6 + x_2x_4x_5x_6 + x_2x_4x_5x_7 + x_2x_4x_5 + \\
&x_2x_4x_6x_7 + x_2x_4x_6 + x_2x_4x_7 + x_2x_4 + x_2x_5x_6x_7 + \\
&x_2x_6x_7 + x_2x_6 + x_2x_7 + x_3x_4x_5x_6x_7 + x_3x_4x_5 + x_3x_4x_6x_7 + \\
&x_3x_4x_6 + x_3x_4x_7 + x_3x_5x_6x_7 + x_3x_6x_7 + x_3x_6 + x_3x_7 + \\
&x_4x_5x_6 + x_4x_5 + x_5x_6x_7 + x_5x_6 + x_5 + x_6 + x_7.
\end{aligned}$$

### 2.1 Initialization Process

A 128-bit key $K = \{k_0, k_1, \ldots, k_{127}\}$ ($k_i \in \{0,1\}$) and an initialization vector $IV = \{iv_0, iv_1, \ldots, iv_{191}\}$ ($iv_i \in \{0,1\}$) are loaded into the registers $A$ and $B$ as follows:

$$a_i = k_i \ (0 \le i \le 127), \quad b_i = iv_i \ (0 \le i \le 191).$$

The initialization process updates the state 448 times without the keystream generation. It consists of the stage 1 (320 cycles) and the stage 2 (128 cycles). In the stage 1, the output of the nonlinear function $s_t$ is fed back to register $B^t$, i.e., $s_t$ is XORed with $b_{t+192}$.

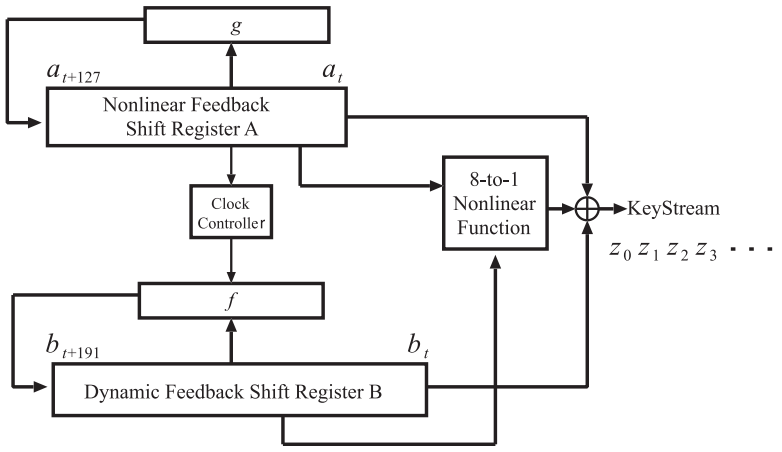


**Fig. 1** RAKAPOSHI Stream Cipher.

In the stage 2, the output of the nonlinear function $s_t$ is fed back to register $A^t$, i.e., $s_t$ is XORed with $a_{t+128}$. The usage of $s_t$ is only the difference between the stage 1 and the stage 2.

After the initialization process, the state $S^{448} = (\{A^{448}, B^{448}\})$ is obtained.

### 2.2 Keystream Generation

For $t \geq 448$, an internal state $S^t = (A^t, B^t)$ generates a keystream bit $z_{t-448}$ such that $z_{t-448} = b_t \oplus a_t \oplus s_t$ with updating the internal state. Note that the fixed key and IV pair must be changed after $2^{64}$ keystream bits are generated.

## 3. Slide Property of Stream Cipher

If two different keys always convert the same plaintexts into the same ciphertexts, such a key pair is called an equivalent key in terms of that these keys are functionally equivalent. Since stream ciphers additionally use IV for generating a keystream, equivalent Key-IV pairs can also be defined. Here, a ciphertext is obtained by XORing a plaintext with a keystream in stream ciphers. Thus, an equivalent Key-IV pair essentially means the pair generating the same keystreams. The existence of these pairs indicates that the effective $(K, IV)$ space is smaller than the expected value which is the sum of the $K$ and $IV$ size.

In stream ciphers, a variant of equivalent $(K, IV)$ called *slide Key-IV pairs* is also defined in Refs. [9], [21]. A slide Key-IV pair generates the same keystream but $w \cdot n$-bit shifted, where $w$ is the size of the word $z_t$ in the keystream, e.g., $w = 1$ for RAKAPOSHI. Though the existence of this pair does not directly affect the effective $(K, IV)$ space unlike the case of equivalent Key-IV pairs, it has the following interesting property.

Let $(K'_{(n)}, IV'_{(n)})$ be a $w \cdot n$-bit slide Key-IV pair of $(K, IV)$. In other words, $(K'_{(n)}, IV'_{(n)})$ generates the $w \cdot n$-bit shifted keystream with respect to that of $(K, IV)$ such that $z'_t = z_{t+n}$ for $0 < t$. Suppose that plaintexts $P = \{p_0, p_1, \ldots, p_L\}$ and $P' = \{p'_0, p'_1, \ldots, p'_L\}$ are encrypted with $(K, IV)$ and $(K'_{(n)}, IV'_{(n)})$, respectively. Then, ciphertexts $C = \{c_0, c_1, \ldots, c_L\}$ and $C' = \{c'_0, c'_1, \ldots, c'_L\}$ are as follows:

$$
\begin{aligned}
C &= \{c_0, c_1, \ldots, c_L\} = \{p_0 \oplus z_0, p_1 \oplus z_1, \ldots, p_L \oplus z_L\},\\
C' &= \{c'_0, c'_1, \ldots, c'_L\} = \{p'_0 \oplus z'_0, p'_1 \oplus z'_1, \ldots, p'_L \oplus z'_L\}\\
&= \{p'_0 \oplus z_n, p'_1 \oplus z_{n+1}, \ldots, p'_L \oplus z_{n+L}\}.
\end{aligned}
$$

If an attacker can get above ciphertexts which are generated from $w{\cdot}n$-bit slide Key-IV pairs, he can obtain information of plaintexts from only ciphertexts without the knowledge of keys by XORing $w \cdot n$-bit shifted $C$ to $C'$ as follows:

$$
\begin{aligned}
c_{n+t} \oplus c'_t &= p_{n+t} \oplus z_{n+t} \oplus p'_t \oplus z_{n+t}\\
&= p_{n+t} \oplus p'_t.
\end{aligned}
$$

At first glance, this assumption seems to be very strong. However, it corresponds to the related-key and chosen IV setting for some classes of stream ciphers [*2]. Besides, a slide Key-IV pair can be used not only for exposing the plaintext information but also for related-key key recovery attacks [9], [21]. Moreover, it may be utilized for speeding up the key search in the single key setting [9].

Therefore, the slide property is a very useful tool of analyses and evaluations of the security of stream ciphers.

[*2] It highly depends on structures and algorithms of target stream ciphers.

## 4. Slide Property of RAKAPOSHI

In this section, we analyze a slide property of RAKAPOSHI stream cipher.

### 4.1 Conditions of Slide Pairs

For RAKAPOSHI, a $(K, IV)$ pair has a corresponding $n$-bit *slide* pair $(K'_{(n)}, IV'_{(n)})$ that generates an $n$-bit shifted keystream for $0 \leq n < 320$, if these pairs satisfy the following conditions:

**Condition 1 :** $S^n(= \{A^n, B^n\}) = S'^0(= \{A'^0, B'^0\})$,
**Condition 2 :** $s^{320+i} = 0 \ \ (0 \leq i < n)$,
**Condition 3 :** $s^{448+i} = 0 \ \ (0 \leq i < n)$,

where $S'^t$ is a state generated from $(K'_{(n)}, IV'_{(n)})$ at time $t$. **Figure 2** illustrates these conditions for an $n$-bit slide pair.

Assume that the condition 1 holds, $S^{320}(= \{A^{320}, B^{320}\})$ and $S'^{320-n}(= \{A'^{320-n}, B'^{320-n}\})$ are identical, because $S^n$ and $S'^0$ are updated in the same manner during the stage 1.

However, $S^{320}$ and $S'^{320-n}$ are updated by different update processes in the stage 1 and 2, respectively. As mentioned in Section 2.1, the difference of these stages is only the usage of $s_t$, i.e., $s_t$ is XORed with $b_{t+192}$ in the stage 1 while it is XORed with $a_{t+128}$ in the stage 2. When the condition 2 holds, the relation of $s^{320+i} = s'^{320-n+i} = 0$ is obtained for $0 \leq i < n$. It allows us to omit these differences of the stage 1 and 2, and then $S^{320+n}(= \{A^{320+n}, B^{320+n}\}) = S'^{320}(= \{A'^{320}, B'^{320}\})$. After that, since these states are updated in the same manner during the stage 2, $S^{448}(= \{A^{448}, B^{448}\})$ and $S'^{448-n}(= \{A'^{448-n}, B'^{448-n}\})$ are surely identical.

In the keystream generation, $s_t$ is used for generating a keystream bits, and does not affect the state updating. Therefore, the condition 3 ensures that $S^{448+n}(= \{A^{448+n}, B^{448+n}\})$ and $S'^{448}(= \{A'^{448}, B'^{448}\})$ are identical. It means that $(K', IV')$ produces an $n$-bit sliding keystream with respect to $(K, IV)$. In other words, the following equations hold, $z_i = z'_{i-n}$ $(n \leq i < 2^{64})$.

### 4.2 Evaluation

Let us estimate how many $n$ -bit slide pairs exist in the RAKAPOSHI stream cipher. The condition 1 is expressed as

$$
\begin{aligned}
A^n &= \{k_n, k_{n+1}, \ldots, k_{127}, x_0, \ldots, x_{n-1}\}\\
&= \{k'_0, k'_1, \ldots, k'_{127}\} = A'^0,\\
B^n &= \{iv_n, iv_{n+1}, \ldots, iv_{191}, y_0, \ldots, y_{n-1}\}\\
&= \{iv'_0, iv'_1, \ldots, iv'_{191}\} = B'^0,
\end{aligned}
$$

where $x_t = g(a_t, a_{t+6}, a_{t+7}, a_{t+11}, a_{t+16}, a_{t+28}, a_{t+36}, a_{t+46}, a_{t+55}, a_{t+62})$ and $y_t = f(b_t, b_{t+14}, b_{t+37}, b_{t+41}, b_{t+49}, b_{t+51}, b_{t+93}, b_{t+107}, b_{t+120}, b_{t+134}, b_{t+136}, b_{t+155}, b_{t+158}, b_{t+176}, a_{t+41}, a_{t+89}) \oplus s_t$. Since the state size and the sum of $K$ and $IV$ size are the same, a $(K, IV)$ pair surely has one pair of $(K'_n, IV'_n)$ satisfying the condition 1 regardless of the value of $n$.

On the other hand, conditions 2 and 3 depend on the value of $n$. The probability that a $(K, IV)$ pair satisfies the conditions 2

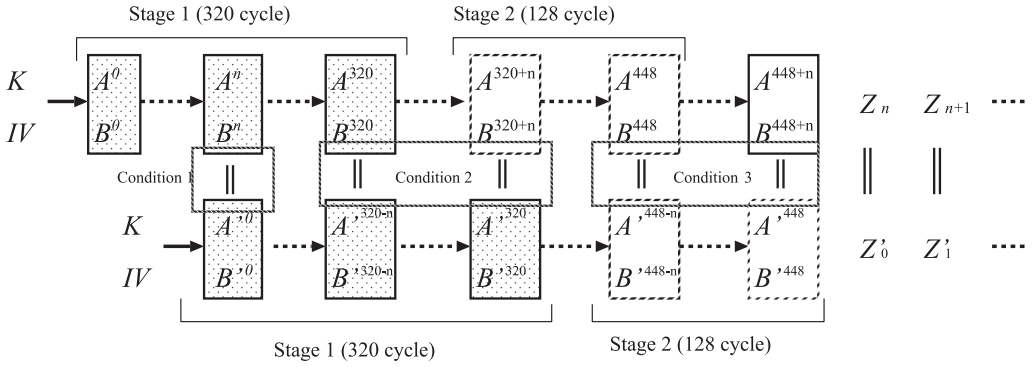


**Fig. 2** $n$-bit slide pair of RAKAPOSHI.

**Table 2** Examples of slide pairs.

| 1-bit slide pair | | | |
|---|---|---|---|
| $K$ | $IV$ | $K'_{(1)}$ | $IV'_{(1)}$ |
| 4bdf973abdd66263$_x$ | 49cba4aa656336eb$_x$ | 97bf2e757bacc4c7$_x$ | 93974954cac66dd7$_x$ |
| d4ef3bfb30609c57$_x$ | be0b3db8cc516480$_x$ | a9de77f660c138af$_x$ | 7c167b7198a2c901$_x$ |
| | 95b8910812c5c95b$_x$ | | 2b712210258b92b7$_x$ |
| keystream | | keystream | |
| $0010000110011010100110110110011011_2$ | | $0100011001101010011010110011011 1_2$ | |
| $1010000101111111001111001000010000_2$ | | $0100010111111100111100100001000 1_2$ | |
| 5-bit slide pair | | | |
| $K$ | $IV$ | $K'_{(5)}$ | $IV'_{(5)}$ |
| 5f3c0c948aa9262e$_x$ | 01660552b4169c5d$_x$ | e78192915524c5cf$_x$ | 2cc0aa5682d38bba$_x$ |
| 7e55d395a458fba2$_x$ | d584fedb576094f6$_x$ | caba72b48b1f745b$_x$ | b09fdb6aec129ed9$_x$ |
| | cb5cd06e132a3644$_x$ | | 6b9a0dc26546c889$_x$ |
| keystream | | keystream | |
| $1001011100000000011100110111010 11_2$ | | $1110000000011100110111010110110 0_2$ | |
| $0110001111111000001101100100111 1_2$ | | $0111111100000110110010011110011 0_2$ | |
| 10-bit slide pair | | | |
| $K$ | $IV$ | $K'_{(10)}$ | $IV'_{(10)}$ |
| d048119b66a37d84$_x$ | 8b75aad54c32a2b6$_x$ | 20466d9a8df61354$_x$ | d6ab5530ca8adbc4$_x$ |
| d51287aef2f796d1$_x$ | f118a4764dd0560a$_x$ | 4a1ebbcbde5b4738$_x$ | 6291d93741582a23$_x$ |
| | 88fc32827bc213cc$_x$ | | f0ca09ef084f334b$_x$ |
| keystream | | keystream | |
| $0010001010001010010001011001011 1_2$ | | $0010100100010110010111100111001 0_2$ | |
| $1001110010010010100111001101011 1_2$ | | $0100101001110011010111010100100 1_2$ | |

**Table 1** Experimental results of probability that a Key-IV pair has a $n$ bit slide pair.

| $n$ | Theoretical value | Experimental value |
|---|---|---|
| 1 | $2^{-2}$ | $2^{-2.0011}$ (4191041/16777216) |
| 2 | $2^{-4}$ | $2^{-4.0039}$ (1045695/16777216) |
| 3 | $2^{-6}$ | $2^{-6.0027}$ (261636/16777216) |
| 4 | $2^{-8}$ | $2^{-7.9942}$ (65797/16777216) |
| 5 | $2^{-10}$ | $2^{-9.9951}$ (16439/16777216) |
| 6 | $2^{-12}$ | $2^{-12.0084}$ (4072/16777216 |
| 7 | $2^{-14}$ | $2^{-14.0084}$ (1018/16777216) |
| 8 | $2^{-16}$ | $2^{-16.0284}$ (251/16777216) |
| 9 | $2^{-18}$ | $2^{-17.933}$ (67/16777216) |
| 10 | $2^{-20}$ | $2^{-19.830}$ (18/16777216) |

and 3 is $2^{-2n}(= 2^{-n} \times 2^{-n})$. Therefore, any $(K, IV)$ pair theoretically has an $n$-bit slide pair $(K'_{(n)}, IV'_{(n)})$ that generates an $n$-bit shifted keystream with a probability of $2^{-2n}$. We have confirmed the correctness of these theoretical values by testing $2^{24}$ random chosen $(K, IV)$ pairs for $n = 0, \ldots, 10$. **Table 1** shows experimental results of the probability that a Key-IV pair has an $n$ bit slide pair for $2^{24}$ randomly-chosen Key-IV pairs. It is confirmed that our theoretical values are correctly approximated. **Table 2** gives examples of 1, 5 and 10 bits slide pairs. In addition, we can say that a $(K, IV)$ pair having $(K'_{(n)}, IV'_{(n)})$ pairs also has $(K'_{(1)}, IV'_{(1)})$ $\ldots$ $(K'_{(n-1)}, IV'_{(n-1)})$ pairs.

For $n = 1$, a $(K, IV)$ pair has $(K'_{(1)}, IV'_{(1)})$ that generates a only 1-bit shifted keystream with a probability of $2^{-2}$, which is a greatly high probability compared to an ideal stream cipher that generates a random keystream by $(K, IV)$. If an attacker can access to stream ciphers using such a slide pair, it is easy to distinguish keystreams from random streams.

### 4.3 Partial Slide Property of RAKAPOSHI

Here, we introduce a variant of the slide property. Recall that conditions 1-3 in Section 4.1 ensure that a $(K, IV)$ pair has an $n$-bit slide pair $(K'_{(n)}, IV'_{(n)})$ that produces an $n$-bit sliding keystream of $(K, IV)$. If the condition 3 does not hold, it is not ensured that $a_{448+128+i}$ and $a'_{448+128+i+n}$ are identical for $0 \le i < n$, due to the difference of usage of $s$. However, these differences do no affect generations of $z_{n+1}(= z'_1), \ldots, z_{60}(= z'_{60-n})$. Thus, if only the conditions 1 and 2 hold, we can obtain the keystream pairs in which $\{z_{n+1}, \ldots, z_{60}\}$ and $\{z'_1, \ldots, z'_{60-n}\}$ are identical. We call such a pair an $n$-bit *partial slide pair*.

Therefore, a $(K, IV)$ pair has an $n$-bit partial slide pair $(K''_{(n)}, IV''_{(n)})$ that generates an $n$-bit partial sliding keystream with

a probability of $2^{-n}$, that occurs with a higher probability than the basic slide property. For $n = 1$, a $(K, IV)$ pair has a 1-bit partial slide pair $(K''_{(1)}, IV''_{(1)})$ with a probability of $2^{-1}$, where 59 bits of $\{z_2, \ldots, z_{60}\}$ and $\{z'_1, \ldots, z'_{59}\}$ are identical. If an attacker can access to stream ciphers using such a partial slide pair, it is easy to distinguish keystreams from random streams. The success probability is $1 - 2^{-59}$, because the 59 bits conditions of the keystream coincidentally hold with a probability of $2^{-59}$ independently from the event of the partial slide pair.

## 5. Related-Key Attack on RAKAPOSHI

In this section, we give a technique for exploiting the slide property of RAKAPOSHI in order to construct a related-key key recovery attack on RAKAPOSHI. To begin with, we explain a method for determining a part of the key bits by utilizing the 1-bit slide property. After that, we generalize it and propose a related-key attack on RAKAPOSHI based on the $n$-bit slide property. Moreover we improve it by using the partial slide property.

### 5.1 Related-Key Attacks Using an 1-bit Slide Pair

Define the related key $K^*_{(1)}$ of this attack as [*3]

$$K^*_{(1)} = \{k^*_0, k^*_1, \ldots, k^*_{127}\} = \{k_1, k_2, \ldots, k_{127}, x_0\}$$

where

$$\begin{aligned} x_0 &= g(a_0, a_6, a_7, a_{11}, a_{16}, a_{28}, a_{36}, a_{46}, a_{55}, a_{62}), \\ &= g(k_0, k_6, k_7, k_{11}, k_{16}, k_{28}, k_{36}, k_{46}, k_{55}, k_{62}). \end{aligned}$$

Since $x_0$ includes only key bits and does not depends on the value of $IV$, a related key $K^*$ is determined if $K$ is given. In the related-key setting, an attacker knows that a pair of $(K, K^*_{(1)})$ holds this relation, though actual values of those are unknown.

This attack uses chosen IV pairs $(IV, IV^*_{(1)})$ satisfying the following relation,

$$\begin{aligned} IV &= \{iv_1, iv_2, \ldots, iv_{191}, y_0\} \\ &= \{iv^*_0, iv^*_1, \ldots, iv^*_{191}\} = IV^*_{(1)}, \end{aligned}$$

where

$$\begin{aligned} y_0 &= f(b_0, b_{14}, b_{37}, b_{41}, b_{49}, b_{51}, b_{93}, b_{107}, b_{120}, b_{134}, \\ &\quad b_{136}, b_{155}, b_{158}, b_{176}, a_{41}, a_{89}) \\ &\quad \oplus v(a_{67}, a_{127}, b_{23}, b_{53}, b_{77}, b_{81}, b_{103}, b_{128}) \\ &= f(iv_0, iv_{14}, iv_{37}, iv_{41}, iv_{49}, iv_{51}, iv_{93}, iv_{107}, iv_{120}, iv_{134}, \\ &\quad iv_{136}, iv_{155}, iv_{158}, iv_{176}, k_{41}, k_{89}) \\ &\quad \oplus v(k_{67}, k_{127}, iv_{23}, iv_{53}, iv_{77}, iv_{81}, iv_{103}, iv_{128}). \end{aligned}$$

In the chosen-IV setting, an attacker is able to choose the values of $IV$ freely. Given $IV$, we can determined $IV^*_{(1)}$ except $iv^*_{191}$ $(= y_0)$, because $y_0$ includes four key bits, $\{k_{41}, k_{89}, k_{67}, k_{127}\}$, which are secret values even in the related-key setting.

If the value of $iv^*_{191}$ is correctly guessed, $(K, IV)$ and $(K^*_{(1)}, IV^*_{(1)})$ satisfy the condition 1 regarding the 1-bit slide pair. Then, $(K^*_{(1)}, IV^*_{(1)})$ generates a 1-bit shifted keystream of $(K, IV)$ with a probability of $2^{-2}$. Since the probability that $iv^*_{191}$ is correctly guessed is $2^{-1}$, we expect to obtain one a 1-bit sliding keystream pair after testing $2^3$ $(IV, IV^*_{(1)})$ pairs. Once such a $(IV, IV^*_{(1)})$ pair is found, we can confirm that $iv^*_{191}(= y_0)$ is correctly guessed. Then, a 1-bit equation of $y_0$, which includes 4 key bits of $\{k_{41}, k_{89}, k_{67}, k_{127}\}$, is obtained. Using four equations, $\{k_{41}, k_{89}, k_{67}, k_{127}\}$ can be determined with the high probability.

The details of the attack procedure are given as follows:

(1) Choose one pair of $(IV, IV^*_{(1)})$, where $iv^*_{191}$ is guessed.
(2) Obtain two keystreams of $(K, IV)$ and $(K^*_{(1)}, IV^*_{(1)})$.
(3) If these keystreams are the 1-bit sliding pair, then store the 1-bit equation of $\{k_{41}, k_{89}, k_{67}, k_{127}\}$ corresponding to $iv^*_{191}$.
(4) Repeat steps 1-3 until 4 equations are obtained.
(5) Determine the key bits of $\{k_{41}, k_{89}, k_{67}, k_{127}\}$ by using four equations.
(6) Obtain the other 124 bits of the key in the brute force manner.

One equation can be obtained with a probability of $2^{-3}$. Thus, it is expected to repeat steps 1-4 $2^5$ $(= 4 \times 2^3)$ times. The time complexity of the steps 1-4 is $2^6$ $(= 2 \times 2^5)$ initialization processes, and the number of required chosen IVs is $2^6$. In step 5, we search $\{k_{41}, k_{89}, k_{67}, k_{127}\}$ by checking obtained 1-bit equations. Specifically, we guess the values of $\{k_{41}, k_{89}, k_{67}, k_{127}\}$, and check whether these values satisfy all four equations. After testing $2^4$ patterns of $\{k_{41}, k_{89}, k_{67}, k_{127}\}$, the one set that holds the equations expects to be found. Thus, the time complexity of the step 5 is estimated as $2^4$ estimations of the equations, which is obviously less than $2^4$ initialization processes. Therefore, the whole time complexity is estimated as $2^{124}(\approx 2^4 + 2^6 + 2^{124})$ initialization processes. This related-key attack recovers a key with a time complexity of $2^{124}$, $2^6$ chosen IVs and one related key.

### 5.2 Related-Key Attacks Using an $n$-bit Slide Pair

We extend the attack exploiting a 1-bit slide pair to an attack based on the $n$-bit slide pair. The related key $K^*_{(n)}$ and the chosen $IV$ pair are defined as,

$$\begin{aligned} K^*_{(n)} &= \{k^*_0, k^*_1, \ldots, k^*_{127}\} = \{k_n, k_{n+1}, \ldots, k_{127}, x_0, \ldots, x_{n-1}\}, \\ IV &= \{iv_n, iv_{n+1}, \ldots, iv_{191}, y_0, \ldots, y_{n-1}\} \\ &= \{iv^*_0, iv^*_1, \ldots, iv^*_{191}\} = IV^*_{(n)}, \end{aligned}$$

assuming that the values of $n$ are less than 127. **Table 3** shows involved key bits of each $y_t$ for $0 \le t \le 12$.

If the values of $\{y_0, \ldots, y_{n-1}\}$ are correctly guessed with a prob-

**Table 3** Included key bits in each $y_t$.

| $y_t$ | Included key bits |
|---|---|
| $y_0$ | 41, 67, 89, 127 |
| $y_1$ | 42, 68, 90, (0, 6, 7, 11, 16, 28, 36, 45, 55, 62) |
| $y_2$ | 43, 69, 91, (1, 7, 8, 12, 17, 29, 37, 46, 56, 63) |
| $y_3$ | 44, 70, 92, (2, 8, 9, 13, 18, 30, 38, 47, 57, 64) |
| $y_4$ | 45, 71, 93, (3, 9, 10, 14, 19, 31, 39, 48, 58, 65) |
| $y_5$ | 46, 72, 94, (4, 10, 11, 15, 20, 32, 40, 49, 59, 66) |
| $y_6$ | 47, 73, 95, (5, 11, 12, 16, 21, 33, 41, 50, 60, 67) |
| $y_7$ | 48, 74, 96, (6, 12, 13, 17, 22, 34, 42, 51, 61, 68) |
| $y_8$ | 49, 75, 97, (7, 13, 14, 18, 23, 35, 43, 52, 62, 69) |
| $y_9$ | 50, 76, 98, (8, 14, 15, 19, 24, 36, 44, 53, 63, 70) |
| $y_{10}$ | 51, 77, 99, (9, 15, 16, 20, 25, 37, 45, 54, 64, 71) |
| $y_{11}$ | 52, 78, 100, (10, 16, 18, 21, 26, 38, 46, 55, 65, 72) |
| $y_{12}$ | 53, 79, 101, (11, 17, 19, 22, 27, 39, 47, 56, 66, 73) |

[*3] This type of related keys has been utilized in attacks of Grain family [9], [21].

ability of $2^{-n}$, $(K^*_{(n)}, IV^*_{(n)})$ generates an $n$-bit sliding keystream of $(K, IV)$ with a probability of $2^{-2n}$. Once we find such pairs, $n$ equations regarding each value of $\{y_0, \ldots, y_{n-1}\}$ are obtained. If $y_t$ includes $m$ bits of the key, $m$ independent equations of $y_t$ are needed for determining $m$ bits of the key.

As an example, let us consider the attack using a 4-bit slide pair. $\{y_0, \ldots, y_3\}$ includes 4, 13, 13 and 13 key bits, respectively, and in total these involve independent 41 key bits. If 13 independent equations regarding each $y$ are obtained [*4], we can determine key bits included in each equation. It implies that this attack requires 13 pairs of $(IV, IV^*_{(4)})$ causing a 4-bit sliding keystream. These pairs are obtained with a time complexity of $2^{17}$ $(= 13 \times 2 \times 2^{3\cdot4})$ and $2^{17}$ $(= 13 \times 2 \times 2^{3\cdot4})$ chosen IVs. Then, 41 bits of the key can be determined with a time complexity of $2^{15}(= 2^4 + 2^{13} + 2^{13} + 2^{13})$ by exhaustively checking obtained equations. Therefore, the whole time complexity is estimated as $2^{87}(= 2^{87} + 2^{15} + 2^{17})$ initialization processes. This related-key attack recovers the key with a time complexity of $2^{87}$ and $2^{17}$ chosen IVs.

Using 8-bit slide pairs, each value of $\{y_0, \ldots, y_7\}$ includes 13 bits of the key except $y_0$ and in total these involve 75 independent key bits. 13 pairs of $(IV, IV^*_{(8)})$ causing a 8-bit sliding keystream are obtained with a time complexity of $2^{29}$ $(= 13 \times 2 \times 2^{3\cdot8})$ and $2^{29}$ $(= 13 \times 2 \times 2^{3\cdot8})$ chosen IVs. Then, in total 75 bits of the key are determined with a time complexity of $2^{16}(= 2^4 + 2^{13} \times 7)$ by exhaustively checking the obtained equations. Therefore, the whole time complexity is estimated as $2^{54}(= 2^{53} + 2^{16} + 2^{29})$ initialization processes. This related-key attack recovers the key with a time complexity of $2^{54}$ and $2^{29}$ chosen IVs.

Using 11-bit slide pairs, each value of $\{y_0, \ldots, y_{10}\}$ includes 13 bits of the key except $y_0$ and in total these involve independent 88 key bits. 13 pairs of $(IV, IV^*_{(11)})$ causing a 11-bit sliding keystream are obtained with a time complexity of $2^{38}$ $(= 13 \times 2 \times 2^{3\cdot11})$ and $2^{38}$ $(= 13 \times 2 \times 2^{3\cdot11})$ chosen IVs. Then, in total 88 bits of the key are determined with a time complexity of $2^{17}(= 2^4 + 2^{13} \times 10)$ by exhaustively checking obtained equations. Therefore, the whole time complexity is estimated as $2^{41}(= 2^{40} + 2^{17} + 2^{38})$ initialization processes. This related-key attack recovers the key with a time complexity of $2^{41}$ and $2^{38}$ chosen IVs. This attack is optimized for the time complexity.

### 5.3 Related-Key Attacks Using an $n$-bit Partial Slide Pair

This attack is further improved by using an $n$-bit partial slide pair in which $\{z_{n+1}, \ldots, z_{60}\}$ and $\{z'_1, \ldots, z'_{60-n}\}$ of the keystreams are identical. Using 13 bits partial slide pairs, i.e., 47 bits of $\{z_{14}, \ldots, z_{60}\}$ and $\{z'_1, \ldots, z'_{47}\}$ are identical, each value of $\{y_0, \ldots, y_{12}\}$ includes 13 bits of the key except $y_0$ and in total these involve 96 independent key bits. 13 pairs of $(IV, IV^*_{(13)})$ causing a 13-bit partial sliding keystream are obtained with a time complexity of $2^{30}$ $(= 13 \times 2 \times 2^{2\cdot13})$ and $2^{30}$ $(= 13 \times 2 \times 2^{2\cdot13})$ chosen IVs. Then, in total 96 bits of the key are determined with a time complexity of $2^{18}(= 2^4 + 2^{13} \times 12)$ by exhaustively checking the obtained equations. Therefore, the whole time complexity is estimated as $2^{33}(= 2^{32} + 2^{18} + 2^{30})$ initialization processes. This related-key attack recovers the key with a time complexity of $2^{33}$ and $2^{30}$ chosen IVs. The success probability is estimated as $(1 - 2^{-47})^{13} \approx 1$.

This technique also allows to improve the attacks based on 1, 4, 8 and 11 bits slide pairs with respect to the data complexity. **Table 4** shows the summary of our results. This result reveals that RAKAPOSHI is practically vulnerable to the related-key attack based on the slide property.

**Table 4** Summary of our related-key attacks.

| Used Slide pair | Time Complexity | Chosen IVs |
|---|---|---|
| 1 bit | $2^{124}$ | $2^6$ |
| 4 bit | $2^{87}$ | $2^{17}$ |
| 8 bit | $2^{54}$ | $2^{29}$ |
| 11 bit | $2^{41}$ | $2^{38}$ |
| 1 bit (partial) | $2^{124}$ | $2^6$ |
| 4 bit (partial) | $2^{87}$ | $2^{13}$ |
| 8 bit (partial) | $2^{54}$ | $2^{21}$ |
| 11 bit (partial) | $2^{41}$ | $2^{27}$ |
| 13 bit (partial) | $2^{33}$ | $2^{30}$ |

## 6. Speed-up Keysearch on RAKAPOSHI

In this section, we give a method for speeding up a keysearch in the single key setting.

In order to improve the naive brute force attack, we exploit partial slide pairs. In particular, we utilize the observation that if the condition 2 regarding $n$-bit partial slide pairs holds, we can check $n$ keys without recalculations of the initialization process.

Assume that an attacker aims to find $K^{target}$ in the brute-force style search, i.e., test all keys with the keystream of $(K^{target}, IV^{target})$. Let us consider that a candidate pair of $(K, IV)$ is set for the test. In the initialization process of $(K, IV)$, if $s^{320+i} = 0$ (condition 2) holds for $0 \le i < n$, then $(K, IV)$ surely has $1, \ldots, n$ bits partial slide pairs such that $\{(K_{(1)}, IV_{(1)}), \ldots, (K_{(n)}, IV_{(n)})\} = \{(A^1, B^1), \ldots, (A^n, B^n)\}$.

Then we can simultaneously verify $n$ keys with only an initialization call of $(K, IV)$ by using additional keystreams of $\{(K^{target}, IV_{(1)}), \ldots, (K^{target}, IV_{(n)})\}$. Note that $n$ bits of $IV_{(n)}$, namely $y_0, \ldots, y_n$, are uncontrollable and cannot be fixed, while the other $(192 - n)$ bits of $IV_{(n)}$ are determined from $IV^{target}$. Thus, this attack requires a set of keystreams generated from $1 + 2^1 + 2^2 \ldots + 2^n$ chosen IVs.

The detailed algorithm is as follows:

(1) Set $K = 0$ and $IV = IV^{target}$
(2) Perform the initialization process and generate keystream bits $(z_0 \ldots z_{60})$.
(3) For $t = 0$ to [smallest $0 \le n < 60$ for which $s^{320+n+1} = 1$], check $(z_t \ldots z_{60})$ matches the keystream of $(K^{Target}, IV_{(t)})$.
  - if matching, output $K^{target} = A^t$
(4) Update $K = A^{t+1}$, and Return to step 2 only if $K \neq 0$.

As estimated in Ref. [9], this algorithm will eventually reach $K = 0$ again, because $K$ is updated in the invertible way. Then, it is expected that this code checks $2^{127}$ key values. The expected number of checked values of $K$ in the step 3 for each loop of steps 2-4 is 2 $(\approx 1 + 1 \cdot 1/4 + 2 \cdot 1/8 + \ldots)$. Thus, the complexity of this algorithm is estimated as $2^{126}$ initialization processes of the step

[*4] For $y_0$, 4 independent equations are enough.

2. If we cannot find the target key, the algorithm can be repeated with different starting values which have a different cycle.

In order to estimate the actual cost of the attack, we consider the case where $1-10$ bits partial slide pairs are used in this algorithm. Since the expected number of checked values of $K$ in the step 3 is 2 ($\approx 1 + 1 \cdot 1/4 + 2 \cdot 1/8 + \ldots + 10 \cdot 1/1024$), the time complexity for searching $2^{127}$ key values is $2^{126}$. After that, to cover all the key space, we will check another cycle with a same complexity. The whole complexity is given as $2^{127}$ initialization processes. The number of the set of IV used for the attack is $2^{11}$ ($\approx 1 + 2 + 2^2 + \ldots + 2^{10}$).

Therefore, we can speed up the keysearch by a factor of two. In the Grain v1 attack [9], this type of attack seems applicable in the case where $IV$ are all 1. Unlike the attack on Grain v1, our attack on RAKAPOSHI can be done for any $IV$ while a set of chosen IVs are needed. This shows that RAKAPOSHI has a 127 bits security instead of 128 bits. However, this attack is a marginal improvement compared to the brute force attack. Thus, we do not claim this to be a real attack based on an algorithmic weakness.

## 7. Conclusion

This paper has investigated slide properties of RAKAPOSHI stream ciphers. First, we have shown that for RAKAPOSHI, any Key-IV pair has a corresponding *slide* Key-IV pair that generates an $n$-bit shifted keystream with a probability of $2^{-2n}$. Then, we showed that this property is able to be converted into key recovery attacks on RAKAPOSHI. In the related-key setting, our attack based on the slide property can recover a 128-bit key with a time complexity of $2^{33}$ and $2^{30}$ chosen IVs. In addition, a method for speeding up the brute force attack by a factor of 2 can be constructed in the single key setting.

These results mainly exploit the self-similarity of the state update function of RAKAPOSHI. If the self-similarity is destroyed, this type of attack can be avoided. For example, inserting round constants or a counter value at each step is effective for preventing the attack presented in this paper.

**Acknowledgments** This work was supported in part by Grant-in-Aid for Scientific Research (C) (KAKENHI 23560455) for Japan Society for the Promotion of Science.

### References

[1] ISO/IEC 18033-4: Amendment 1 - Information technology - security techniques - Encryption algorithms - Part 4: Stream ciphers, JTC 1/SC 27 (IT Security Tech.) (2011). available from ⟨http://www.iso.org⟩.
[2] Arnault, F. and Berger, T.P.: F-FCSR: Design of a New Class of Stream Ciphers, Gilbert, H. and Handschuh, H. (Eds.), *FSE*, Vol.3557 of LNCS, pp.83–97, Springer (2005).
[3] Aumasson, J-P., Henzen, L., Meier, W. and Naya-Plasencia, M.: Quark: A Lightweight Hash, Mangard, S. and Standaert, F-X. (Eds.), *CHES*, Vol.6225 of LNCS, pp.1–15, Springer (2010).
[4] Babbage, S. and Dodd, M.: The MICKEY Stream Ciphers, *New Stream Cipher Designs - The eSTREAM Finalists*, pp.191–209 (2008).
[5] Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K. and Verbauwhede, I.: SPONGENT: A Lightweight Hash Function, *Cryptographic Hardware and Embedded Systems - CHES 2011*, pp.312–325 (2011).
[6] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y. and Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher, Paillier, P. and Verbauwhede, I. (Eds.), *CHES*, Vol.4727 of LNCS, pp.450–466, Springer (2007).
[7] Cannière, C.D. and Preneel, B.: Trivium, *New Stream Cipher Designs - The eSTREAM Finalists*, pp.244–266 (2008).
[8] De Cannière, C., Dunkelman, O. and Knezevic, M.: KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers, Clavier, C. and Gaj, K. (Eds.), *CHES*, Vol.5747 of LNCS, pp.272–288, Springer (2009).
[9] De Cannière, C., Küçük, Ö. and Preneel, B.: Analysis of Grain's Initialization Algorithm, Vaudenay, S. (Ed.), *AFRICACRYPT*, Vol.5023 of LNCS, pp.276–289, Springer (2008).
[10] Cid, C., Kiyomoto, S. and Kurihara, J.: The RAKAPOSHI Stream Cipher, Qing, S., Mitchell, C.J. and Wang, G. (Eds.), *ICICS*, Vol.5927 of LNCS, pp.32–46, Springer (2009).
[11] Ding, L. and Guan, J.: Cryptanalysis of RAKAPOSHI Stream Cipher, Cryptology ePrint Archive, Report 2012/696 (2012).
[12] Dinur, I., Güneysu, T., Paar, C., Shamir, A. and Zimmermann, R.: An Experimentally Verified Attack on Full Grain-128 Using Dedicated Reconfigurable Hardware, Lee, D.H. and Wang, X. (Eds.), *ASIACRYPT*, Vol.7073 of LNCS, pp.327–343, Springer (2011).
[13] The eSTREAM Project. available from ⟨http://www.ecrypt.eu.org/stream⟩.
[14] Guo, J., Peyrin, T. and Poschmann, A.: The PHOTON Family of Lightweight Hash Functions, Rogaway, P. (Ed.), *CRYPTO*, Vol.6841 of LNCS, pp.222–239, Springer (2011).
[15] Guo, J., Peyrin, T., Poschmann, A. and Robshaw, M.J.B.: The LED Block Cipher, *Cryptographic Hardware and Embedded Systems - CHES 2011*, pp.326–341 (2011).
[16] Hell, M. and Johansson, T.: Breaking the Stream Ciphers F-FCSR-H and F-FCSR-16 in Real Time, *J. Cryptology*, Vol.24, No.3, pp.427–445 (2011).
[17] Hell, M., Johansson, T., Maximov, A. and Meier, W.: The Grain Family of Stream Ciphers, *New Stream Cipher Designs - The eSTREAM Finalists*, pp.179–190 (2008).
[18] Imai, H. and Yamagishi, A.: CRYPTREC (Japanese Cryptographic Algorithm Evaluation Project), van Tilborg, H.C.A. and Jajodia, S. (Eds.), *Encyclopedia of Cryptography and Security* (*2nd ed.*), pp.285–288, Springer (2011).
[19] Isobe, T., Ohigashi, T. and Morii, M.: Slide Cryptanalysis of Lightweight Stream Cipher RAKAPOSHI, Hanaoka, G. and Yamauch, T. (Eds.), *IWSEC*, Vol.7631 of LNCS, pp.138–155, Springer (2012).
[20] Kiyomoto, S., Tanaka, T. and Sakurai, K.: K2: A Stream Cipher Algorithm using Dynamic Feedback Control, Hernando, J., Fernández-Medina, E. and Malek, M. (Eds.), *SECRYPT*, pp.204–213, INSTICC Press (2007).
[21] Lee, Y., Jeong, K., Sung, J. and Hong, S.: Related-Key Chosen IV Attacks on Grain-v1 and Grain-128, Mu, Y., Susilo, W. and Seberry, J. (Eds.), *ACISP*, Vol.5107 of LNCS, pp.321–335, Springer (2008).
[22] Orumiehchiha, M.A., Pieprzyk, J., Shakour, E. and Steinfeld, R.: Security Evaluation of Rakaposhi Stream Cipher, Cryptology ePrint Archive, Report 2012/656 (2012).
[23] Preneel, B. and Takagi, T. (Eds.): *Cryptographic Hardware and Embedded Systems - CHES 2011*, *Proc. 13th International Workshop, Nara, Japan*, Vol.6917 of LNCS, Springer (2011).
[24] Robshaw, M.J.B. and Billet, O. (Eds.): *New Stream Cipher Designs - The eSTREAM Finalists*, Vol.4986 of LNCS, Springer (2008).
[25] Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T. and Shirai, T.: Piccolo: An Ultra-Lightweight Blockcipher, *Cryptographic Hardware and Embedded Systems - CHES 2011*, pp.342–357 (2011).

**Takanori Isobe** received his B.E. and M.E. degrees from Kobe University, Japan, in 2006 and 2008, respectively. He joined Sony Corporation in 2008. His current research interests include cryptography. He receive the SCIS Paper Award from ISEC group of IEICE in 2008, and Best Paper Award of Fast Software Encryption (FSE) 2011 from the International Association for Cryptologic Research (IACR).

**Toshihiro Ohigashi** received his B.E. and M.E. degrees from University of Tokushima, Japan, and Ph.D. degree from Kobe University in 2002, 2004, and 2008, respectively. Since 2008, he has been an Assistant Professor in Information Media Center, Hiroshima University. His current research interests include cryptography, information security, and network protocol. He received the SCIS 20th Anniversary Award from ISEC group of IEICE in 2003. He is a member of IPSJ.

**Masakatu Morii** received his B.E. degree in electrical engineering and the M.E. degree in electronics engineering from Saga University, Saga, Japan, and D.E. degree in communication engineering from Osaka University, Osaka, Japan, in 1983, 1985, and 1989, respectively. From 1989 to 1990 he was an Instructor in the Department of Electronics and Information Science, Kyoto Institute of Technology, Japan. From 1990 to 1995 he was an Associate Professor at the Department of Computer Science, Faculty of Engineering at Ehime University, Japan. From 1995 to 2005 he was a Professor at the Department of Intelligent Systems and Information Science, Faculty of Engineering at the University of Tokushima, Japan. Since 2005, he has been a Professor at the Department of Electrical and Electronics Engineering, Faculty of Engineering at Kobe University, Japan. His research interests are in error correcting codes, cryptography, discrete mathematics and computer networks and information security. He is a member of IEEE.