# Security Evaluation of Rakaposhi Stream Cipher

Mohammad Ali Orumiehchiha[1], Josef Pieprzyk[1], Elham Shakour[2] and Ron Steinfeld[3]

[1]Center for Advanced Computing, Algorithms and Cryptography, Department of Computing,
Faculty of Science, Macquarie University, Sydney, NSW 2109, Australia
(mohammad.orumiehchiha,josef.pieprzyk)@mq.edu.au
[2]Faculty of Mathematics, Amirkabir University,Tehran, Iran
elham.shakoor@gmail.com
[3]Clayton School of Information Technology
Monash University, Clayton VIC 3800, Australia
ron.steinfeld@monash.edu

**Abstract.** Rakaposhi is a synchronous stream cipher, which uses three main components a non-linear feedback shift register (NLFSR), a dynamic linear feedback shift register (DLFSR) and a non-linear filtering function ($NLF$). NLFSR consists of 128 bits and is initialised by the secret key $K$. DLFSR holds 192 bits and is initialised by an initial vector ($IV$). $NLF$ takes 8-bit inputs and returns a single output bit. The work identifies weaknesses and properties of the cipher. The main observation is that the initialisation procedure has the so-called sliding property. The property can be used to launch distinguishing and key recovery attacks. The distinguisher needs four observations of the related $(K, IV)$ pairs. The key recovery algorithm allows to discover the secret key $K$ after observing $2^9$ pairs of $(K, IV)$. In the proposed related-key attack, the number of related $(K, IV)$ pairs is $2^{(128+192)/4}$ pairs. The key recovery algorithm allows to discover the secret key $K$ after observing $2^9$ related $(K, IV)$ pairs.
Further the cipher is studied when the registers enter short cycles. When NLFSR is set to all ones, then the cipher degenerates to a linear feedback shift register with a non-linear filter. Consequently, the initial state (and Secret Key and $IV$) can be recovered with complexity $2^{63.87}$.
If DLFSR is set to all zeros, then $NLF$ reduces to a low non-linearity filter function. As the result, the cipher is insecure allowing the adversary to distinguish it from a random cipher after $2^{17}$ observations of keystream bits. There is also the key recovery algorithm that allows to find the secret key with complexity $2^{54}$.

**Keywords:** Rakaposhi Stream Cipher, Related Key Attack, Weak State, Cryptanalysis, Distinguishing Attack, Key Recovery Attack.

## 1 Introduction

Stream ciphers are symmetric cipher systems, which provide confidentiality in many applications ranging from mobile phone communication to private virtual networks. They may be implemented efficiently in software and hardware and are a preferred choice when dealing with an environment that has restricted computing resources (such as smart cards, RF tags). The inner work of stream ciphers is controlled normally by two parameters: an initialisation vector ($IV$) and a key ($K$). The initialisation vector is public and the key is secret.

There are many tools for analysis of stream ciphers. The most two prominent are the linear and differential attacks. There are also many more "exotic" tools for analysis. One such tool is the related key attack. This attack is especially dangerous when an adversary has access to many pairs of $(IV, K)$. This may happen if the adversary is able to experiment with a stream cipher device that has been left unattended (so-called midnight or lunchtime attacks). The adversary may modify the unknown secret key by forcing changes on few key bits, may try different $IV$s or may modify the clock procedure.

The Rakaposhi stream cipher was designed by Cid, Kiyomoto, and Kurihara in 2009 (see [1]). The cipher is based on a non-linear feedback shift register (NLFSR) and a dynamic linear shift register (DLFSR). The design was crafted to be suitable for lightweight implementations, where computing, power and time resources are in short supply. The cipher offers the 128-bit security and has been designed to complement the eStream portfolio for hardware-oriented stream ciphers. The designers of the cipher claim that Rakaposhi is an efficient synchronous stream cipher that resists all known attacks and they conjecture that it is also secure against other yet unknown attacks.

This work analyses the Rakaposhi cipher and shows its weaknesses. In particular, we

– examine the resistance of the cipher against the related key attack, where the adversary can access related pairs $(IV, K)$,
– study the security implications when NLFSR enters a short cycle,
– investigate the security level when DLFSR enters a short cycle.

### 1.1 Related Works

The related key attack is studied in the context the Rakaposhi cipher and its initialization procedure. Similar analysis tools can be found in [3, 4, 8] but in a different context. The related key attack can be seen as a member of the differential cryptanalysis toolbox. We use the slide attack published by Canniere et. al. in [2] to launch the related key attack. The second part of our work is influenced by the paper by Zhang and Wang [9], in which the authors study the security of the Grain stream cipher [5, 6]. While working on the paper, we have become aware of a paper by [7] that shows a similar analysis of the initialization procedure of Rakaposhi.

The rest of the paper is structured as follows. Section 2 describes briefly the Rakaposhi stream cipher. Section 3 presents the weaknesses of the cipher and investigate the security of the initialisation procedure under the related key attack. Section 4 discusses the security implications when one of the registers (either NLFSR or DLFSR) enters a short cycle. Section 5 concludes the work.

## 2 Description of Rakaposhi Stream Cipher

The Rakaposhi stream cipher consists of the following three building blocks (see Figure 1) :

– a 128-bit NLFSR also called the register $A$,
– a 192-bit DLFSR also called the register $B$,
– a non-linear function $NLF$

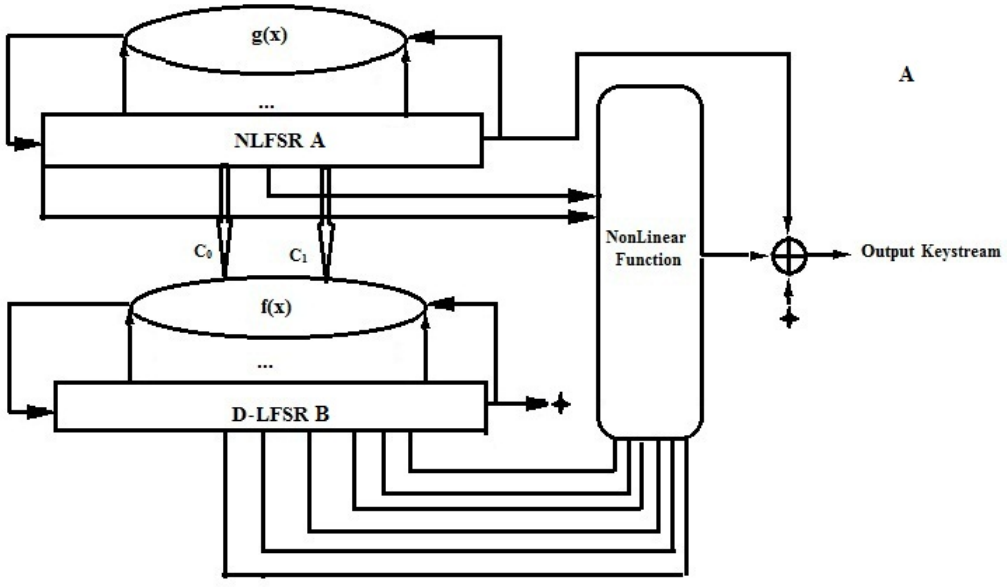


**Fig. 1.** Rakaposhi Stream Cipher

The NLSFR register $A$ is defined by its feedback function:

$$\begin{aligned} g(x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9) = {} & x_1x_3x_9 \oplus x_1x_7x_9 \oplus x_5x_8 \oplus x_2x_5 \oplus \\ & x_3x_8 \oplus x_2x_7 \oplus x_9 \oplus x_8 \oplus x_7 \oplus x_6 \oplus \\ & x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \oplus x_0 \oplus 1, \end{aligned}$$

where $a_{t+128} = g(a_t, a_{t+6}, a_{t+7}, a_{t+11}, a_{t+16}, a_{t+28}, a_{t+36}, a_{t+45}, a_{t+55}, a_{t+62})$ and $a_{t+i}$ is $i^{th}$ bit of the register $A$ at clock $t$.

The DLFSR register $B$ is controlled by two bits $(c_0, c_1)$ taken from the state of NLFSR. The bits choose one of four possible characteristic polynomials of DLFSR. The form of the polynomials is as follows:

$$\begin{aligned} f(x) = {} & x_{192} \oplus x_{176} \oplus c_0 x_{158} \oplus (1 \oplus c_0) x_{155} \oplus c_0 c_1 x_{136} \oplus \\ & c_0(1 \oplus c_1) x_{134} \oplus c_1(1 \oplus c_0) x_{120} \oplus (1 \oplus c_0)(1 \oplus c_1) x_{107} \oplus \\ & x_{93} \oplus x_{51} \oplus x_{49} \oplus x_{41} \oplus x_{37} \oplus x_{14} \oplus 1, \end{aligned} \tag{1}$$

where the bits $(c_0, c_1)$ are the $42^{th}$ and $90^{th}$ bits of the register $A$ at clock $t$, respectively. The recursive relation for DLFSR is as follows:

$$\begin{aligned} b_{t+192} = {} & b_t \oplus b_{t+14} \oplus b_{t+37} \oplus b_{t+41} \oplus b_{t+49} \oplus b_{t+51} \oplus \\ & b_{t+93} \oplus c_0 \cdot c_1 \cdot b_{t+107} \oplus c_0 \cdot c_1 \cdot b_{t+120} \oplus c_0 \cdot c_1 \cdot b_{t+134} \oplus c_0 \cdot c_1 \cdot b_{t+136} \oplus \\ & c_0 \cdot b_{t+155} \oplus c_0 \cdot b_{t+158} \oplus b_{t+176} \end{aligned} \tag{2}$$

where $\overline{c_i} = 1 \oplus c_i$ denotes the negation of $c_i$ and $b_{t+i}$ is the $i^{th}$ bit of $B$ at clock $t$.

Rakaposhi uses a non-linear filtering function $NLF : GF(2^8) \rightarrow GF(2)$, which is based on the AES S-Box. The function $NLF$ is a balanced Boolean function of its algebraic degree equal to 7. $NLF$ takes 8-bit inputs (2 bits from $A$ and 6 bits from $B$) according to the following relation

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

where the two bits $a_{t+67}, a_{t+127}$ are taken from $A$ and the other bits from $B$. Finally, the keystream output is generated by linear combination of the outputs of both registers $A$ and $B$ with the output of the $NLF$ function. The reader interested in more detail is referred to the original paper [1].

### 2.1 Initialisation Procedure

The goal of the initialisation procedure is to mix $IV$ and the secret key $K$. Assume that $IV = [iv_0, \cdots, iv_{191}]$ and $K = [k_0, \cdots, k_{127}]$. $IV$ and $K$ are loaded to NLFSR and DLFSR, respectively so

$$\begin{aligned} a_i &= k_i \quad \text{for} \quad 0 \leq i \leq 127 \\ b_j &= iv_j \quad \text{for} \quad 0 \leq j \leq 191, \end{aligned}$$

where the bits of registers $A$ and $B$ are $a_i$ and $b_j$, respectively. The registers $A$ and $B$ are then clocked 448 times without producing output keystream bits. This stage is divided into two phases:

**Phase 1:** the output of $NLF$ is linearly combined with the feedback of the register $B$ for the first 320 clocks.

**Phase 2:** the output of $NLF$ is linearly combined with the feedback of the register $A$ for the next 128 clocks.

After finishing Phase 2, the cipher starts producing keystream outputs.

## 3 Cryptanalysis of Rakaposhi Stream Cipher

Now, we show how we can launch the distinguishing and key recovery attacks on the Rakaposhi cipher. The attacks use a sliding property of the cipher. An interesting property of the proposed attacks is that their complexities are not affected by the number of clocks, which the cipher performs during the initialisation process. This means that the attacks works even if the number of clocks is increased.

### 3.1 Properties of Rakaposhi Cipher

We present some cryptographic properties of the Rakaposhi stream cipher that corroborate the proposed attacks.

1. The secret key and $IV$ are loaded in two registers $A$ and $B$, respectively. Consequently, at clock $t = 0$, $A$ contains $K$ and $B$ – $IV$.
2. The initialisation procedure applies the same primitives that are used during the keystream generation stage. This implies that the initialisation for the key and $IV$ is similar to the initialisation for the key and $IV$ when they are shifted by one position. We refer to this characteristics as the sliding property.
3. The register $A$ (NLFSR) has a short cycle of the length 1. When the state of $A$ becomes all ones, then $A$ stays in this state forever.
4. The register $B$ (DLFSR) has a short cycle of the length 1. When the state of $B$ becomes all zeros, then $B$ stays in this state forever.

The first two properties mean that the adversary may find related $(K, IV)$ pairs, which produce keystream outputs that are shifted. The third and fourth properties can be exploited by the adversary so they can distinguish the cipher from a truly random binary source and recover internal state of the cipher and finally corresponding secret key.

### 3.2 Related Key Attack on Rakaposhi

In our sliding attack we assume that we have two related pairs $(K, iv)$ and $(\widehat{K}, \widehat{iv})$. Consider the initialisation procedure for the two pairs. Let $K = (k_0, \cdots, k_{127})$ and $iv = (iv_0, ..., iv_{191})$ be loaded into the registers $A$ and $B$, respectively. Denote the states of the registers $A$ and $B$ at the clock $t$ by $A^t$ and $B^t$, respectively. The evolution of states over time is described below.

$$
\begin{array}{ll}
A^0 = [k_0, \cdots, k_{127}] & B^0 = [iv_0, \cdots, iv_{191}] \\
A^1 = [k_1, \cdots, k_{127}, a_{128}] & B^1 = [iv_1, \cdots, iv_{191}, b_{192}] \\
\vdots & \vdots \\
\xRightarrow{\textit{Phase 2 of Initialization}} A^{192} = [a_{192}, \cdots, a_{319}] & B^{192} = [b_{192}, \cdots, b_{383}] \\
\vdots & \vdots \\
\xRightarrow{\textit{Initialization finished}} A^{320} = [a_{320}, \cdots, a_{447}] & B^{320} = [b_{320}, \cdots, b_{511}] \\
\xRightarrow{\textit{Key Generation started}} A^{321} = [a_{321}, \cdots, a_{448}] & B^{321} = [b_{321}, \cdots, b_{512}] \\
A^{322} = [a_{322}, \cdots, a_{449}] & B^{322} = [b_{321}, \cdots, b_{512}]
\end{array}
$$

The keystream output bits $z_i$; $i \geq 0$, are computed as follows:

$$
\begin{aligned}
z_0 &= a_{321} \oplus b_{321} \oplus NLF(a_{320+67}, a_{320+127}, b_{320+23}, b_{320+53}, b_{320+77}, b_{320+81}, b_{320+103}, b_{320+128}) \\
z_1 &= a_{322} \oplus b_{322} \oplus NLF(a_{321+67}, a_{321+127}, b_{321+23}, b_{321+53}, b_{321+77}, b_{321+81}, b_{321+103}, b_{321+128}) \\
&\vdots
\end{aligned}
$$

The relation between keystreams generated by the cipher when initialised by the related pairs is described by the following theorem.

**Theorem 1.** *Given two pairs $(K, iv)$ and $(\widehat{K}, \widehat{iv})$, where $K = (k_0, \cdots, k_{127})$ and $iv = (iv_0, ..., iv_{191})$. If the pair $(\widehat{K}, \widehat{iv})$ satisfies the following equations*

$$
\begin{cases}
\widehat{k_i} = k_{i+1} & 0 \leq i \leq 126, \quad \hat{k}_{127} = a_{128} \\
\widehat{iv_i} = iv_{i+1} & 0 \leq j \leq 190, \ \widehat{iv}_{191} = b_{192}
\end{cases} \tag{3}
$$

*then the keystream output bits $\widehat{z}_i = z_{i+1}$ for $i \geq 0$ with probability $2^{-2}$.*

*Proof.* By satisfying Equation (3), the internal states of $[A^{191}, B^{191}]$ are equal to $[\widehat{A}^{190}, \widehat{B}^{190}]$. But at the next clock, the states may not be identical because the state $[\widehat{A}^{191}, \widehat{B}^{191}]$ is still at the first step while $[A^{192}, B^{192}]$ is running at the second step. If $\hat{b}^{383} = b^{382}$, which occurs with probability 1/2, then

$$[\widehat{A}^{191}, \widehat{B}^{191}] = [A^{192}, B^{192}].$$

The same argument is valid for the states $[A^{320}, B^{320}]$ and $[\widehat{A}^{319}, \widehat{B}^{319}]$. The states are identical, when $\hat{a}^{446} = b^{447}$, which also happens with probability 1/2. Consequently, $\widehat{z}_i = z_{i+1}$ for $i \geq 0$ with total probability 1/4. □

Table 1 presents some $(K, IV)$ pairs, which produce shifted identical key stream outputs. According to Theorem 1, the adversary can use this weakness to generate the same but $l-$bit shifted keystream outputs by defining related $(K, IV)$ pairs with probability $2^{-2 \cdot l}$. The discovered weakness allows the

**Table 1.** Shifted identical key stream outputs corresponding two related $(K, IV)$ pairs

| | Key | IV | Output bits |
|---|---|---|---|
| 1 | 10011011110011101010000 | 01000111100000010100010001101011 0 | 00000110000100011100111 |
| | 10000000111010011000000 | 00000000001000000110111000011111 0 | 11000001010001010100101 |
| | 00011000101100010011111 | 01110101011000011111000011001100 1 | 10010100001101111001101 |
| | 01110111010000010010011 | 11000101110110110000010100110110 0 | 10101010111010000001101 |
| | 00000001001100110010011 | 01000111000000010010010010111100 1 | 11001110010001000111101 |
| | 010011010111100111 | 10101101010110001011001010 1 | 011110000011110101 |
| | 00110111100111010100011 | 10001111000000101000100011010110 0 | 00001100001000111001111 |
| | 00000001110100110000000 | 00000000010000001101110000111110 0 | 10000010100010101001101 |
| | 00110001011000100111100 | 11101010110000111110000110011001 1 | 00101000011011110011101 |
| | 11101110100000100100100 | 10001011101101100000101001101100 0 | 01010101101000000011011 |
| | 00000010011001100100100 | 10001110000000100100100101111001 1 | 10011100100010001111000 |
| | 100110101111001111 | 01011010101100010100101011 1 | 111100000111101010 |
| 2 | 00000000010111111011000 | 00111000101110001110111000101100 0 | 01110100100110000001111 |
| | 11101100111001000101111 | 01000100011110000110010101001011 1 | 00110010110000101110100 |
| | 01110110110011110010100 | 01011001000101000000110100110001 1 | 11111001100001111011110 |
| | 11011101100001110010000 | 01001001101101110001110011010101 1 | 10011110000100100110100 |
| | 11011000001110010110100 | 11111001010000110011100111011000 0 | 11100100000111001010000 |
| | 100111100110110000 | 00000111100100011001011010 1 | 100010110111101111 |
| | 00000000101111111011001 | 01110001011100011101110001011000 0 | 11101001001100000001110 |
| | 11011001110010001011100 | 10001000111100001100101010010111 0 | 01100101100001011101101 |
| | 11101101100111100101010 | 10110010001010000001101001100011 0 | 11110011000011111011101 |
| | 10111011000011100100010 | 10010011011011100011100110010111 0 | 00111100001001001101010 |
| | 10110000011100101101010 | 11110010100001100111001110111000 0 | 11001000001110010100010 |
| | 001111001101100001 | 00001111001000110010110101011 1 | 000101101111011111 |

adversary to distinguish the cipher from a random bit generator. Assume that the adversary can apply related $(K, IV)$ pairs, which does not know the exact values of secret key. Then, after applying $m$ ($\gg 4$) different ( randomly generated ) related $(K, IV)$ pairs, on average $m/4$ of generated key stream outputs have identical sequences with just one bit shift.

### 3.3 Recovery of Secret Keys

Now, we propose a key recovery attack that exploits the sliding property of pairs $(K, IV)$. We show an algorithm that allows to recover the 128-bit key after $O(2^9)$ initialisation operations. The algorithm needs 512 related $(K, IV)$ pairs and it can find the secret key with probability one.

Assume that both $(K, IV)$ and $(\widehat{K}, \widehat{IV})$ generate almost identical keystream bits, where the second keystream is a shifted by one bit copy of the first keystream. At clock $t = 321$, the first generated bit

is $b_{192}$, which is equal to:

$$\begin{aligned} b_{192} &= b_0 \oplus b_{14} \oplus b_{37} \oplus b_{41} \oplus b_{49} \oplus b_{51} \oplus b_{93} \oplus (1 \oplus c_0)(1 \oplus c_1)b_{107} \\ &\quad \oplus (1 \oplus c_0)c_1 b_{120} \oplus c_0(1 \oplus c_1)b_{134} \oplus c_0 c_1 b_{136} \oplus (1 \oplus c_0)b_{155} \oplus c_0 b_{158} \oplus b_{176} \\ &\quad \oplus NLF(a_{67}, a_{127}, b_{23}, b_{53}, b_{77}, b_{81}, b_{103}, b_{128}), \end{aligned}$$

where $c_0 = a_{41}$ and $c_1 = a_{89}$. Since the contents of $b_i$ $(0 \leq i \leq 191)$ are known and can be chosen by the adversary, then Equation (4) is a non-linear relation based on only 4 unknown variables $a_{41}, a_{89}, a_{67}, a_{127}$. We now take a closer look at the equation (4):

$$\begin{aligned} b_{192} &= b_0 \oplus b_{14} \oplus b_{37} \oplus b_{41} \oplus b_{49} \oplus b_{51} \oplus b_{93} \oplus (1 \oplus a_{41})(1 \oplus a_{89})b_{107} \\ &\oplus (1 \oplus a_{41})a_{89}b_{120} \oplus a_{41}(1 \oplus a_{89})b_{134} \oplus a_{41}a_{89}b_{136} \oplus (1 \oplus a_{41})b_{155} \oplus a_{41}b_{158} \oplus b_{176} \\ &\oplus a_{67}a_{127}b_{23}b_{53}b_{77}b_{81}b_{103} \oplus a_{67}a_{127}b_{23}b_{53}b_{77}b_{81} \oplus a_{67}a_{127}b_{23}b_{53}b_{77}b_{103} \\ &\oplus a_{67}a_{127}b_{23}b_{53}b_{81}b_{103}b_{128} \oplus a_{67}a_{127}b_{23}b_{53}b_{81}b_{103} \oplus a_{67}a_{127}b_{23}b_{53}b_{81}b_{128} \\ &\oplus a_{67}a_{127}b_{23}b_{53}b_{81} \oplus a_{67}a_{127}b_{23}b_{53}b_{103}b_{128} \oplus a_{67}a_{127}b_{23}b_{77}b_{81}b_{103} \\ &\oplus a_{67}a_{127}b_{23}b_{77} \oplus a_{67}a_{127}b_{23}b_{81}b_{103} \oplus a_{67}a_{127}b_{23}b_{81}b_{128} \oplus a_{67}a_{127}b_{23}b_{128} \\ &\oplus a_{67}a_{127}b_{23} \oplus a_{67}a_{127}b_{53}b_{77}b_{81}b_{103}b_{128} \oplus a_{67}a_{127}b_{53}b_{77}b_{81}b_{128} \\ &\oplus a_{67}a_{127}b_{53}b_{77}b_{81} \oplus a_{67}a_{127}b_{53}b_{77}b_{128} \oplus a_{67}a_{127}b_{53}b_{77} \oplus a_{67}a_{127}b_{53}b_{103} \\ &\oplus a_{67}a_{127}b_{77}b_{81}b_{103}b_{128} \oplus a_{67}a_{127}b_{77}b_{81}b_{103} \oplus a_{67}a_{127}b_{77}b_{81}b_{128} \oplus a_{67}a_{127}b_{77}b_{103}b_{128} \\ &\oplus a_{67}a_{127}b_{77}b_{128} \oplus a_{67}a_{127}b_{81}b_{103}b_{128} \oplus a_{67}a_{127}b_{81}b_{103} \oplus a_{67}a_{127}b_{81} \oplus a_{67}a_{127}b_{103} \\ &\oplus a_{67}a_{127} \oplus a_{67}b_{23}b_{53}b_{77}b_{81}b_{103} \oplus a_{67}b_{23}b_{53}b_{77}b_{81}b_{128} \oplus a_{67}b_{23}b_{53}b_{77} \oplus a_{67}b_{23}b_{53}b_{81}b_{103}b_{128} \\ &\oplus a_{67}b_{23}b_{53}b_{81}b_{103} \oplus a_{67}b_{23}b_{53}b_{81}b_{128} \oplus a_{67}b_{23}b_{53}b_{103} \oplus a_{67}b_{23}b_{77}b_{81}b_{103}b_{128} \oplus a_{67}b_{23}b_{81}b_{103} \\ &\oplus a_{67}b_{23}b_{81} \oplus a_{67}b_{23}b_{103}b_{128} \oplus a_{67}b_{23}b_{128} \oplus a_{67}b_{53}b_{77}b_{81}b_{103}b_{128} \oplus a_{67}b_{53}b_{77}b_{81}b_{103} \\ &\oplus a_{67}b_{53}b_{77}b_{81}b_{128} \oplus a_{67}b_{53}b_{77}b_{81} \oplus a_{67}b_{53}b_{77}b_{128} \oplus a_{67}b_{53}b_{81}b_{103}b_{128} \oplus a_{67}b_{53}b_{81} \\ &\oplus a_{67}b_{53}b_{103} \oplus a_{67}b_{53} \oplus a_{67}b_{77}b_{81}b_{103} \oplus a_{67}b_{77}b_{103}b_{128} \oplus a_{67}b_{81}b_{103} \\ &\oplus a_{67}b_{103} \oplus a_{67} \oplus a_{127}b_{23}b_{53}b_{77} \oplus a_{127}b_{23}b_{53}b_{81}b_{103} \oplus a_{127}b_{23}b_{53}b_{81}b_{128} \\ &\oplus a_{127}b_{23}b_{53}b_{81} \oplus a_{127}b_{23}b_{53} \oplus a_{127}b_{23}b_{77}b_{81}b_{103} \oplus a_{127}b_{23}b_{77}b_{103} \oplus a_{127}b_{23}b_{77} \\ &\oplus a_{127}b_{23}b_{81} \oplus a_{127}b_{23} \oplus a_{127}b_{53}b_{77}b_{81}b_{103}b_{128} \oplus a_{127}b_{53}b_{77}b_{81}b_{128} \oplus a_{127}b_{53}b_{77}b_{103}b_{128} \\ &\oplus a_{127}b_{53}b_{77}b_{103} \oplus a_{127}b_{53}b_{77} \oplus a_{127}b_{53}b_{81}b_{103} \oplus a_{127}b_{53}b_{81} \oplus a_{127}b_{53}b_{103} \\ &\oplus a_{127}b_{53}b_{128} \oplus a_{127}b_{77}b_{81}b_{103}b_{128} \oplus a_{127}b_{77}b_{81}b_{128} \oplus a_{127}b_{81}b_{103} \oplus a_{127}b_{81}b_{128} \oplus a_{127}b_{81} \\ &\oplus a_{127}b_{103}b_{128} \oplus a_{127}b_{103} \oplus a_{127} \oplus NLF'(b_{23}, b_{53}, b_{77}, b_{81}, b_{103}, b_{128}) \end{aligned} \tag{4}$$

where $NLF'$ is a Boolean function including all monomials of $NLF$, in which variables $a_{67}, a_{127}$ do not exist. Note that the adversary does not need to solve the equation. Instead, the adversary can recover four bits of the secret key by choosing appropriate bits for $IV$s. For example, if

$$\begin{cases} b_i = 0 & i \in \Phi \\ b_{158} = 1 \end{cases}$$

where $\Phi = \{0, 14, 37, 41, 49, 51, 93, 107, 120, 134, 136, 155, 176, 23, 53, 77, 81, 103, 128\}$, then $b_{192} = a_{41}$. Consequently, $\widehat{iv}_{191} = k_{41}$. In this way, the adversary is able to retrieve other four secret key bits. The number of the required related pairs $(K, IV)$ is 4. On the average, to find the valid pairs, the adversary needs 16 pairs. In other words, to retrieve 4 secret key bits, the adversary should run the initialisation algorithm 16 times for the related $(K, IV)$ pairs. Now, the adversary can keep going and continue the attack finding consecutive 4-bit parts of the secret key. Finally, to determine the whole 128-bit secret key, the adversary needs to apply $512 = 32 \times 16$ related $(K, IV)$ pairs on the average.

## 4 Weak $(K, IV)$ Pairs

In this section we study the security implications of short cycles of the two registers $A$ and $B$. Note that the initialisation procedure takes $K$ and $IV$ loads them to $A$ and $B$, respectively and then the

cipher is clocked 448 times. At the end of the initialisation, the cipher can be set in the following weak states:

- the register $A$ contains all ones and the state loops forever. To identify the collection of pairs $(K, IV)$ that leads to this state of $A$, it is enough to set $A = \mathbf{1}$ and to set $B$ to an arbitrary 192-bit vector and clock backwards. This process will generate $2^{192}$ pairs $(K, IV)$ that leads the initialisation to weak states.
- the register $B$ contains all zeros and the state loops forever. Again, to identify the collection of pairs $(K, IV)$ that leads to this state of $B$, it is enough to set $B = \mathbf{0}$ and to set $A$ to an arbitrary 128-bit vector and clock backwards. This process will generate $2^{128}$ pairs $(K, IV)$ that leads the initialization to weak states.
- both registers $A = \mathbf{1}$ and $B = \mathbf{0}$. There is a single pair of $(K, IV)$ only. To identify it, set registers appropriately and clock backwards. This case is not very interesting as it can be easily identified.

### 4.1 Weak $(K, IV)$ Pairs Leading to $A = 1$

It may happen that after the initialisation, the pair $(K, IV)$ leads to $A = \mathbf{1}$. An immediate consequence of this is that the register $A$ contains all ones and it stays in this state for all clocks. The adversary is able to identify this case and they are also able to recover the weak pair $(K, IV)$ that has led to $A = \mathbf{1}$. Clearly, if the adversary knows $IV$, then the task of finding $K$ is easier.

Note that the cipher with the register $A$ in the state of all ones is equivalent to a 192-bit LFSR whose outputs are filtered by a non-linear Boolean function $h$ with an 6-bit input. The function $h$ is the non-linear function $NLF$ with two bits set to ones (those that are coming from $A$). The function is a balanced function from $h : GF(2^6) \to GF(2)$ of degree 5 and non-linearity 20 and it is given below.

$$\begin{aligned} h(x_1, x_2, x_3, x_4, x_5, x_6) = {} & 1 \oplus x_1 \oplus x_1x_2 \oplus x_3 \oplus x_1x_3 \oplus x_1x_4 \oplus x_3x_4 \oplus x_2x_3x_4 \oplus x_5 \\ & \oplus x_1x_2x_5 \oplus x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_3x_4x_5 \oplus x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \\ & \oplus x_1x_6 \oplus x_2x_6 \oplus x_1x_3x_6 \oplus x_1x_2x_3x_6 \oplus x_4x_6 \oplus x_1x_4x_6 \oplus x_1x_2x_4x_6 \\ & \oplus x_3x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_2x_3x_4x_6 \oplus x_1x_2x_3x_4x_6 \oplus x_5x_6 \oplus x_2x_3x_5x_6 \\ & \oplus x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_1x_2x_4x_5x_6 \oplus x_2x_3x_4x_5x_6 \end{aligned}$$

The function can be approximated by a linear Boolean function $1 \oplus x_1 \oplus x_1 \oplus x_6$ with probability:

$$Pr(h = (1 + x_1 + x_2 + x_6)) = \frac{44}{64} = 0.6875 = 0.5 + 2^{-2.415}$$

The algebraic immunity of the function is 3 and the number of the annihilators is 10. To recover the contents of the register $B$, we may apply a basic algebraic attack the needs $2^{22.75}$ observations of the keystream bits and whose complexity is $2^{63.87}$. Once the adversary knows the contents of $B$ at the end of the initialization, they can clock backwards to recover the weak pair $(K, IV)$.

### 4.2 Weak $(K, IV)$ Pairs Leading to $B = 0$

The second class of weak $(K, IV)$ pairs leads to the state with $B = \mathbf{0}$. In this case the register $B$ stays in the zero state for all clocks. Consequently, all the outputs of DLFSR are zeros, which is equivalent to removal of the register $B$ from the cipher. The goal of the adversary is to recover the pair $(K, IV)$. Now we show that the adversary is able to recover the initial state (and the secret key by clocking NLFSR backwards) faster than in $2^{54}$ steps.

Note that the $NLF$ function is now used with its 6 bits coming from the register $B$ set to zero. Consequently, the keystream output function that is a linear combination of the least significant bit of the register $A$ with the output of the $NLF$ function. The keystream output function is denoted by $\ell : \{0, 1\}^3 \to \{0, 1\}$ and is of the following form:

$$l(x_1, x_2, x_3) = x_1 \oplus x_2 \oplus x_1x_2 \oplus x_3.$$

The function $\ell$ is a non-linear balanced Boolean function of degree 2. One of the best approximations of $\ell$ is the linear function $x_3$. It is easy to check that

$$Pr(l = x_3) = \frac{6}{8} = 0.75 = 0.5 + 2^{-2} \tag{5}$$

**Distinguishing Attack** If $B = \mathbf{0}$, then the adversary may distinguish the generated keystream bits from a random bit generator. Consider the keystream output bits at clocks $t+0$, $t+6$, $t+7$, $t+11$, $t+16$, $t+28$, $t+36$, $t+45$, $t+55$, $t+62$. If we use the approximation (see Equation (5)) then we can write

$$Pr(z_{t+128} = g(z_{t+0}, z_{t+6}, z_{t+7}, z_{t+11}, z_{t+16}, z_{t+28}, z_{t+36}, z_{t+45}, z_{t+55}, z_{t+62})) \approx 0.502 \tag{6}$$

This means that the adversary requires around $2^{17}$ observations of the keystream output bits to tell apart the cipher from a random bit generator with negligible error probability.

**Recovery Attack** To recover the pair $(K, IV)$, the adversary may use the linear approximation of $\ell$ and try to guess the contents of $A$. The probability of the correct guess for the state is $(0.75)^{128} = 2^{-53.12}$, which is so far from the probability $2^{-128}$. In other words, the cipher has at most 54 bits security.

## 5 Conclusions

In this paper, we analysed the initialisation algorithm of the Rakaposhi stream cipher. We started from observations about cryptographic weak points of the cipher. We discovered the so-called sliding property of the pairs $(K, IV)$. This property can be exploited by launching distingushing and key recovery attacks. We showed that there is a distinguishing attack that needs four related $(K, IV)$ pairs only. Our key recovery attack recovers all bits of the secret key $K$ after observing $2^9$ related $(K, IV)$ pairs.

In the second part of the work, we studied the security of Rakaposhi when either the register $A$ or $B$ enters a short cycle at the end of the initialisation procedure. When the register $A$ loops in the all-ones state, then the adversary is able to recover the pair $(K, IV)$. Rakaposhi in this case degenerates to a LFSR cipher with a non-linear filter function. It is shown that the initial state of the register $B$ can be discovered by an algorithm of time complexity $2^{63.87}$.

If the register $B$ enters the zero state at the end the initialisation procedure, then we showed two efficient algorithms: one to distinguish Rakaposhi from a random bit generator and the other to recover the pair $(K, IV)$. The distinguisher needs $2^{17}$ keystream bit observations. The key recovery algorithm requires around $2^{54}$ operations. Note that this cryptographic weakness can be explored by the adversary when they have access to the cipher device and are allowed to play with the device by running it for different $IV$s.

## References


1. C. Cid, S. Kiyomoto, and J. Kurihara, *The rakaposhi stream cipher*, in Proceedings of the 11th international conference on Information and Communications Security, ICICS'09, Berlin, Heidelberg, 2009, Springer-Verlag, pp. 32–46.
2. C. De Cannière, O. Küçük, and B. Preneel, *Analysis of grain's initialization algorithm*, in Proceedings of the Cryptology in Africa 1st international conference on Progress in cryptology, AFRICACRYPT'08, Berlin, Heidelberg, 2008, Springer-Verlag, pp. 276–289.
3. H. Englund, T. Johansson, and M. Snmez Turan, *A framework for chosen iv statistical analysis of stream ciphers*, in Progress in Cryptology INDOCRYPT 2007, K. Srinathan, C. Rangan, and M. Yung, eds., vol. 4859 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2007, pp. 268–281.
4. E. Filiol, *A new statistical testing for symmetric ciphers and hash functions*, in Proceedings of the 4th International Conference on Information and Communications Security, ICICS '02, London, UK, UK, 2002, Springer-Verlag, pp. 342–353.

5. M. Hell, T. Johansson, and W. Meier, *Grain - a stream cipher for constrained environments*, ECRYPT Stream Cipher Project.
6. ———, *Grain ; a stream cipher for constrained environments*, Int. J. Wire. Mob. Comput., 2 (2007), pp. 86–93.
7. T. Isobe, T. Ohigashi, and M. Morii, *Slide cryptanalysis of lightweight stream cipher rakaposhi*, in Advances in Information and Computer Security, G. Hanaoka and T. Yamauchi, eds., vol. 7631 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 138–155.
8. M. juhani O. Saarinen, *Chosen-iv statistical attacks on estream stream ciphers*, in eSTREAM, ECRYPT Stream Cipher Project, Report 2006/013, 2006, pp. 5–19.
9. H. Zhang and X. Wang, *Cryptanalysis of stream cipher grain family*, in Cryptology ePrint Archive, Report 2009/109, 2009.