Nonlinear Dyn (2018) 91:505–513
https://doi.org/10.1007/s11071-017-3885-3

ORIGINAL PAPER

Security analysis of the pseudo-random bit generator based on multi-modal maps

Dragan Lambić

Received: 2 July 2017 / Accepted: 16 October 2017 / Published online: 27 October 2017

Abstract In this paper, a security analysis of the pseudo-random bit generator based on multi-modal maps is made, which reveals existence of serious security problems. Depending on parameter k, there are certain number of weak keys causing improper functioning of this generator. Also, based on certain number of consecutive output bits, the initial values of this pseudo-random number generator (PRNG) can be obtained with attack which complexity is significantly less than estimated key space. Although the assumed safety of the example of cipher based on this PRNG (when $k = 3$) is estimated at 2^{159}, it is possible to carry out successful known-plaintext attack with the complexity less than 2^{128}. For above-mentioned reasons, analyzed PRNG cannot be considered safe for the use in cryptographic systems.

Keywords Chaos · Pseudo-random number generator · Cryptanalysis · Cryptography

1 Introduction

A pseudo-random number generator (PRNG) produces sequences which resembles to a sequence of true random numbers, but they are computed from an initial seed value by some algorithm, and therefore they are completely deterministic [1]. Recently, chaos have been widely used for secure communications and encryption [2], and chaotic systems are often used as the core component of PRNG [3]. A cryptographically secure PRNG, in addition to common requirements, should be highly unpredictable, and it should be computationally unfeasible to compute any preceding bits based on some known part of output sequence [1].

However, great number of existing PRNGs suffer from serious security problems, and therefore they are not safe for the use in cryptographic systems. A problem that is very often encountered in chaos-based PRNGs is problem with weak keys and equivalent keys. In great number of chaos-based PRNGs, the control parameters of the used chaotic systems are based on the secret key. In the case when the process of obtaining the control parameters from the secret key have certain shortcomings, there is a possibility that chaotic system used in PRNG could evolve in an non-chaotic way [4]. In this situation, PRNG cannot produce pseudo-random numbers required by the cryptosystem.

For example, in pseudo-random number generator based on pseudorandomly enhanced logistic map [5], initial condition and control parameter are calculated on the basis of secret key. However, approximately 2^{50} different secret keys produce the same initial condition and control parameter of enhanced logistic map, which makes this PRNG non-resistant to brute-force attack [6]. Above example indicate that there is a need for a thorough security analysis of key space of existing PRNGs.

D. Lambić (✉)
Faculty of Education, University of Novi Sad, Podgorička 4, Sombor, Serbia
e-mail: dragan.lambic@pef.uns.ac.rs

Another problem that is very often encountered in PRNG design is predictability. Although cryptographically secure PRNG should be highly unpredictable, it is often very easy to compute preceding bits based on some known part of output sequence. For example, Mersenne Twister is very fast PRNG which is considered unsafe for the use in cryptographic systems because only 624 iterations is required in order to determine all of the parameters needed to predict its next output bit [7]. For this reason, serious security analysis of predictability of existing PRNGs is needed in order to determine which PRNG could be used in cryptographic systems.

PRNGs are very important part of image encryption algorithms. Security of image encryption algorithms depends largely on quality and security of its underlying PRNG. For example, PRNG presented in paper [8] have equivalent secret keys which can be used for decryption of cipher-images of certain size [9,10]. Image encryption algorithm proposed in paper [11] use PRNG, however, presence of weak keys and equivalent keys was established [12]. Image encryption algorithm based on nonlinear chaotic algorithm is breakable through different attacks of variable complexity because chaotic map used for generation of pseudo-random numbers has a non-uniform distribution [13]. Chosen-ciphertext attack can be used in order to obtain the equivalent version of the secret key of an image encryption algorithm with compound chaotic stream cipher based on perturbation [14]. Image encryption algorithm based on chaos and line map is vulnerable to differential cryptanalysis which demonstrates that the security of the scheme depends only on the permutation key [15]. Besides these, there are many other examples of cryptanalyzed image encryption algorithms.

In [1] the pseudo-random bit generator based on multi-modal maps is proposed. This PRNG is proposed as a cryptographically secure PRNG which should be able to resist the chosen-plaintext attack. Unfortunately, security analysis revealed existence of serious security problems. In analyzed PRNG, initial values of chaotic map are used as a secret key. Security analysis revealed that many values of multi-modal map lead to non-chaotic behavior, causing the existence of a certain number of weak keys. Therefore, it is not safe to choose secret key in completely random manner.

Also, based on certain number of consecutive output bits, an attacker can recover the initial values of this PRNG with attack which complexity is significantly less than estimated key space. It is considered that some cipher or PRNG is broken if attacker can exploit perceived weaknesses with a complexity lower than a estimated key space [16]. Cipher or PRNG must have more than 2^{128} different secret keys in order to resist brute-force attack [17]. Although the example of cipher based on this PRNG (when $k = 3$) should be able to resist the chosen-plaintext attack, it is possible to carry out successful known-plaintext attack with the complexity less than 2^{128}.

The rest of this paper is organized as follows. In Sect. 2, the analyzed PRNG and gray-scale image cryptosystem based on this PRNG are described. Security analysis of analyzed PRNG and cipher based on this PRNG is presented in Sect. 3. In Sect. 4, the conclusions are drawn.

2 Description of the analyzed PRNG

In paper [1] PRNG is presented, which use k-modal map and a combination of its k-time series. This PRNG can be used for $k > 0$, and in the paper [1] example for $k = 3$ is presented and analyzed. In this paper, we will, for purposes of clarity, present only the case in which $k = 3$. For general case of the analyzed PRNG, please see reference [1].

An example of the k-modal map when $k = 3$ have 3 subintervals, and it is presented by the following equation:

$$f_\beta(x) = \beta \begin{cases} (\frac{1}{3} - x) \cdot x, & \text{for} x \in [0, \frac{1}{3}); \\ (\frac{2}{3} - x) \cdot (x - \frac{1}{3}), & \text{for} x \in [\frac{1}{3}, \frac{2}{3}); \\ (1 - x) \cdot (x - \frac{2}{3}), & \text{for} x \in [\frac{2}{3}, 1]; \end{cases} \tag{1}$$

where $\beta \in [0, 36]$. Depending on the value of this parameter the system can be unimodal, bi-modal, or tri-modal. In general case map have k subintervals and $\beta \in [0, 4 \cdot k^2]$.

The example of PRNG proposed in [1] for $k = 3$ is described by following steps.

Step 1: Set the value of $k > 0$. In this example $k = 3$.
Step 2: Compute the values of β_j, for $j = 1, \ldots, k$, by following formulas:
$\beta_1 = 4 \cdot k$;
$\beta_j = j \cdot \beta_1$; for $j \geq 2$.

For $k = 3$, $\beta_1 = 4 \cdot 3 = 12$, $\beta_2 = 2 \cdot 12 = 24$ and $\beta_3 = 3 \cdot 12 = 36$.

Step 3: For each $j \leq k$, split the domains of functions $f_{\beta_j}(x)$ into $2 \cdot j$ regions $\delta_1^j, \ldots, \delta_{2 \cdot j}^j$ which are separated by the values $a_1^j, \ldots a_{2 \cdot j - 1}^j$. For $k = 3$, there are three functions $f_{\beta_j}(x)$. Function $f_{\beta_1}(x)$ is defined in interval $x \in [0, \frac{1}{3})$ which is divided into two regions δ_1^1 and δ_2^1 split by $a_1^1 = \frac{1}{6}$. For each $x_{i+1}^1 = f_{\beta_1}(x_i^1)$ corresponding value of ζ_{i+1}^1 is obtained in following way. If $x_{i+1}^1 \in \delta_1^1$ $(x_{i+1}^1 < a_1^1)$, then $\zeta_{i+1}^1 = 0$. If $x_{i+1}^1 \in \delta_2^1$ $(x_{i+1}^1 > a_1^1)$ then $\zeta_{i+1}^1 = 1$.

Intervals for the other two functions are divided in a similar way. Function $f_{\beta_2}(x)$ is defined in interval $x \in [0, \frac{2}{3})$ which is divided into four regions $\delta_1^2, \delta_2^2, \delta_3^2, \delta_4^2$ separated by $a_1^2 = 0.1521$, $a_2^2 = 0.40174$, $a_3^2 = 0.5954$. If $x_i^2 \in (\delta_1^2 \cup \delta_3^2)$, then $\zeta_i^2 = 0$, otherwise $\zeta_i^2 = 1$. Function $f_{\beta_3}(x)$ is defined in interval $x \in [0, 1]$ which is divided into six regions $\delta_1^3, \ldots, \delta_6^3$ separated by $a_1^3 = 0.1465$, $a_2^3 = 0.396$, $a_3^3 = 0.639$, $a_4^3 = 0.8335$, $a_5^3 = 0.9572$. If $x_i^3 \in (\delta_1^3 \cup \delta_3^3 \cup \delta_5^3)$, then $\zeta_i^3 = 0$, and otherwise $\zeta_i^3 = 1$.

Step 4: A k chaotic sequences, based on initial values $x_0^1, \ldots, x_0^k$, are generated by $x_{i+1}^j = f_{\beta_j}(x_i^j)$. Each sequence x^j is used to produce binary sequence ζ^j by following procedure described in Step 3. When $k = 3$, initial values x_0^1, x_0^2, x_0^3 are used to obtain binary sequences $\zeta^1, \zeta^2, \zeta^3$. Finally, output bits of the analyzed PRNG Z are obtained by $Z_i = \zeta_i^1 \oplus \zeta_i^2 \oplus \zeta_i^3$.

2.1 Gray-scale image cryptosystem based on the analyzed PRNG

In paper [1], a stream cipher algorithm for gray-scale images is presented, in order to demonstrate the application and quality of the proposed PRNG. Secret key of this cipher consist of initial conditions $x_0^1, \ldots, x_0^k$ of the k-modal chaotic map. According to the IEEE floating-point standard, the computational precision of the 64-bit double-precision number is 2^{53}. Key space depends on the number of initial conditions k. In analyzed example, $k = 3$, so the key space is estimated at $2^{53} \cdot 2^{53} \cdot 2^{53} = 2^{159}$ [1].

This stream cipher is used for encryption of images represented with $N \times M$ pixels. First, one column of random values $R(l, 1) \in \{0, 1, \ldots, 255\}$ is added to the image where $l = 1, \ldots, N$. This image is called augmented image and its size is $N \times (M + 1)$ pixels. Denote by P_i bits of augmented image. Then cipher image bits are calculated by:

$$C_i = \begin{cases} P_i \oplus Z_i \oplus IV_i, & \text{for } \ 0 \leq i < 8; \\ P_i \oplus Z_i \oplus C_{i-8}, & \text{for } \ i \geq 8; \end{cases} \tag{2}$$

where C_i is the ith bit of cipher image, Z_i is the ith bit produced by analyzed PRNG and IV is an initialization vector of 8 bits.

3 Security analysis

All identified shortcomings of the analyzed PRNG will be shown on the example for $k = 3$ which is presented in detail in paper [1], but these shortcomings exist in the general case as well.

3.1 Weak keys

In analyzed PRNG, initial values of chaotic map are used as a secret key. It is very important that the secret key is selected in a random manner. Therefore, initial values $x_0^1, \ldots, x_0^k$ should be random in order to prevent attacks on the secret key. On the other hand, if initial values of chaotic map are not selected carefully, then it is possible that chaotic map evolves in an non-chaotic way. This problem is especially important in hardware realizations of chaotic maps, because chaotic behavior could be degraded or even lost due to variation in the values of the circuit elements such as capacitance or transconductance ratios [18,19].

Chaotic k-modal map have certain number of fixed points $(f_\beta(x) = x)$ depending on values of k and β_j. When $k = 3$, function $f_{\beta_1}(x)$ have two fixed points 0 and $\frac{1}{4}$. If some x_i^1 is equal to one of these fixed points, then all subsequent states of $f_{\beta_1}(x)$ will be equal to each other and therefore all values of ζ_{i+s}^1, for $s > 0$ will have same value (zero or one). For that reason, instead of three functions, only two functions will have output bits based on chaotic behavior, which reduces security of the analyzed PRNG.

Due to the fact that the chaotic k-modal map is quadratic function, each of these two fixed points can be obtained on the basis of two values, for example

$f_{\beta_1}(0) = f_{\beta_1}(\frac{1}{3}) = 0$ and $f_{\beta_1}(\frac{1}{12}) = f_{\beta_1}(\frac{1}{4}) = \frac{1}{4}$. Each value $0, \frac{1}{3}, \frac{1}{12}, \frac{1}{4}$ can be obtained on the basis of two values (8 values in total), and so on. Therefore, if we consider I iterations required to obtain I output bits of analyzed PRNG, there is 2^I values of x_0^1 which lead to one of these two fixed points. However, due to constraints caused by discretization of real numbers, the vast majority of these values will be found in the same intervals of length $\frac{1}{2^{53}}$ which is why a greater number of real values will be represented with the same discretized values.

When we consider functions $f_{\beta_2}(x)$ and $f_{\beta_3}(x)$, the number of fixed points is even greater. Function $f_{\beta_2}(x)$ is piecewise function composed of two quadratic functions and have four fixed points $0, \frac{7}{24}, \frac{69-\sqrt{153}}{144}, \frac{69+\sqrt{153}}{144}$ and each of these fixed points can be obtained on the basis of four values, except for 0. Therefore, there is approximately 4^I values of x_0^2 which lead to one of these fixed points in I iterations. Function $f_{\beta_3}(x)$ is piecewise function composed of three quadratic functions and have six fixed points $0, \frac{11}{36}, \frac{35-\sqrt{73}}{72}, \frac{35+\sqrt{73}}{72}, \frac{54}{72}, \frac{64}{72}$ and each of these fixed points can be obtained on the basis of six values, except for 0. Therefore, there is approximately 6^I values of x_0^3 which lead to some of these fixed points in I iterations.

If all three initial conditions x_0^1, x_0^2, x_0^3 are equal to some appropriate fixed point, then the analyzed PRNG would generate a sequence in which all bits would be equal to each other. There are $2 \cdot 4 \cdot 6 = 48$ such weak keys. It is not such a big number compared to the total key space of 2^{159} different keys. However, the question is how many keys will lead to a situation where all three values x_i^1, x_i^2, x_i^3 are equal to some of the fixed points. Great number of real values x_0^1, x_0^2, x_0^3 lead to some fixed point; however, the vast majority of these values is lost in the process of discretization. On the other hand, there are some examples of quadratic functions in literature, in which 89% of initial points lead to fixed points [20].

Piecewise chaotic map could guarantee chaotic behavior under the condition that certain guidelines on selecting the initial conditions are followed [21]. However, if initial conditions, which are used as secret key, are chosen by method which is not completely random, then the secret key is susceptible to cryptographic attacks. Because certain number of weak keys exists, analyzed PRNG [1] should be used with caution.

3.2 Cryptanalysis of the analyzed PRNG

Basic property of a cryptographically secure PRNG is unpredictability [1]. PRNG is considered unpredictable if an attacker cannot predict value of next output bit with a probability greater than 50%, based on n consecutive output bits of some PRNG. Also, if PRNG is cryptographically secure, it is computationally unfeasible to compute any preceding bits, based on n consecutive output bits. In this section, we will show that initial values of chaotic k-modal map can be obtained with attacks which complexity is significantly less than estimated key space.

First, we will present attack 1. Assume that the attacker knows the values of the first n output bits of the analyzed PRNG $Z_0, \dots Z_{n-1}$. First, attacker should guess values of all ζ_i^j for $i < n$ and $j \le k$. For each i there are 2^k possibilities. However, because attacker know value of Z_i and $Z_i = \zeta_i^1 \oplus \dots \oplus \zeta_i^k$, only 2^{k-1} guesses are needed in order to obtain values of ζ_i^j for some particular i. For example, when $k = 3$ and attacker try to guess values of $(\zeta_i^1, \zeta_i^2, \zeta_i^3)$, there are 8 possibilities (0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1). However, based on known value Z_i, the number of possible values is reduced to 4. For example, if $Z_i = 0$, then $(\zeta_i^1, \zeta_i^2, \zeta_i^3)$ can have only some of the values (0,0,0), (0,1,1), (1,0,1), (1,1,0). Therefore, instead of $(2^k)^n$ guesses, only $(2^{k-1})^n$ guesses are needed in order to obtain all ζ_i^j.

When the values $\zeta_0^j, \dots, \zeta_{n-1}^j$ are known, by using reverse interval mapping [22], attacker can narrow the interval in which the x_0^j is located for approximately 2^n times (up to a precision limit). Example of this process will be shown on function $f_{\beta_1}(x)$ when $k = 3$. According to Eq. 1, $f_{\beta_1}(x) = \beta_1 \cdot (\frac{1}{3} - x) \cdot x = -12x^2 + 4x$ which is defined in interval $x \in [0, \frac{1}{3})$ divided into two regions δ_1^1 and δ_2^1 split by $a_1^1 = \frac{1}{6}$. Assume that the attacker knows the first 3 values of ζ_i^1 which will in this example be $\zeta_0^1 = 0$, $\zeta_1^1 = 0$ and $\zeta_2^1 = 1$. Therefore, $x_0^1 \in \delta_1^1$, $x_1^1 \in \delta_1^1$ and $x_2^1 \in \delta_2^1$. Based on $x_1^1 = f_{\beta_1}(x_0^1) \in \delta_1^1$, attacker can determine in which part of interval $\delta_1^1 = [0, \frac{1}{6})$ is x_0^1 located by solving equation $\frac{1}{6} = f_{\beta_1}(x)$. There are two solutions of this quadratic equation $\frac{4+2\sqrt{2}}{24}$ and $\frac{4-2\sqrt{2}}{24}$, and because only $\frac{4-2\sqrt{2}}{24} \in [0, \frac{1}{6})$, attacker now knows that x_0^1 is located in interval $[0, \frac{4-2\sqrt{2}}{24})$.

Based on $x_2^1 = f_{\beta_1}(f_{\beta_1}(x_0^1)) \in \delta_2^1$, attacker can further reduce the interval in which x_0^1 is located by solving equation $\frac{4-2\sqrt{2}}{24} = f_{\beta_1}(x)$. Only one solution of this quadratic equation $\frac{4-\sqrt{8+4\sqrt{2}}}{24} \in [0, \frac{4-2\sqrt{2}}{24})$ so attacker now knows that x_0^1 is located in interval $(\frac{4-\sqrt{8+4\sqrt{2}}}{24}, \frac{4-2\sqrt{2}}{24})$. In this way, on the basis of only three known values of ζ_i^1, search interval for x_0^1 was reduced from $[0, \frac{1}{3})$ to $(\frac{4-\sqrt{8+4\sqrt{2}}}{24}, \frac{4-2\sqrt{2}}{24})$ and therefore search for x_0^1 is about 9 times shorter. A similar procedure can be applied on functions $f_{\beta_2}(x)$ and $f_{\beta_3}(x)$ in order to narrow the intervals in which x_0^2 and x_0^3 are located.

When $k = 3$, there are three initial values x_0^1, x_0^2, x_0^3 for which key space is estimated at $2^{53} \cdot 2^{53} \cdot 2^{53} = 2^{159}$ [1]. By knowing n consecutive output bits of analyzed PRNG, attacker needs only $(2^2)^n$ guesses in order to find all ζ_i^j for $i < n$ and $j \leq 3$ which enables reduction in search interval for each x_0^j for approximately $2^n < 2^{53}$ times. Reduction in search interval for some x_0^j by more than 2^{53} times is not possible, because precision of 2^{53} is used. Therefore, if values of 53 consecutive output bits of analyzed PRNG are known, complexity of only 2^{106} is needed in order to recover initial values x_0^1, x_0^2, x_0^3 (or x_i^1, x_i^2, x_i^3). If value of some x_i^j is recovered, it is very easy to calculate value of x_0^j. Complexity of 2^{106} is drastically less than 2^{159} reported in [1] or 2^{128} recommended in [17], so example of analyzed PRNG (when $k = 3$) cannot be considered safe for the use in cryptographic systems.

Also, there is another approach that can exploit the previously described weakness of the analyzed PRNG, which will be called attack 2. Assume that the attacker knows the values of the first n output bits of the analyzed PRNG $Z_0, \ldots Z_{n-1}$. In this version of the attack, attacker should guess values of all x_0^j for $1 < j \leq k$. When $k = 3$, attacker only need to guess values of x_0^2, x_0^3. Based on x_0^2, x_0^3 attacker can produce binary sequences ζ_i^2, ζ_i^3 (for $i < n$) by following procedure described in Step 3 of the analyzed PRNG. After that, attacker can calculate values of sequence ζ_i^1 by using $\zeta_i^1 = Z_i \oplus \zeta_i^2 \oplus \zeta_i^3$. When the values ζ_i^1 are known, by using reverse interval mapping procedure described in attack 1, attacker can narrow the interval in which the x_0^1 is located for approximately 2^n times. If $n = 53$, then the complexity of only 2^{106} is needed in order to recover initial values x_0^1, x_0^2, x_0^3, because attacker need to guess only two values x_0^2, x_0^3 in order to calculate value of x_0^1.

3.3 Example 1 of cryptanalysis of the analyzed PRNG

In this section example of the attack 1 will be described. Assume that user of the analyzed PRNG randomly selected initial values $x_0^1 = 0.149284547945563$, $x_0^2 = 0.519067023060468$, $x_0^3 = 0.881509362057897$ which will be considered as a secret key. Based on this initial values, user generate sequence of 53 bits $Z =$ 001101011000 01110001111011010011011001101010011110110 by using analyzed PRNG for $k = 3$. Assume that the attacker knows the values of the first 53 output bits Z, but do not know initial values x_0^1, x_0^2, x_0^3.

First, attacker should guess values of $\zeta_i^1, \zeta_i^2, \zeta_i^3$ for all $i < 53$. Because $Z_0 = 0$, $(\zeta_0^1, \zeta_0^2, \zeta_0^3)$ can have only some of the four values (0,0,0), (0,1,1), (1,0,1), (1,1,0). Also, because $Z_2 = 1$, $(\zeta_2^1, \zeta_2^2, \zeta_2^3)$ can have only some of the four values (0,0,1), (0,1,0), (1,0,0), (1,1,1). Therefore, instead of 2^{159} guesses, only 2^{106} guesses are needed in order to obtain all $\zeta_i^1, \zeta_i^2, \zeta_i^3$ for $i < 53$. Guessed values of $\zeta_i^1, \zeta_i^2, \zeta_i^3$ are shown in Table 1.

Based on the values of $\zeta_i^1, \zeta_i^2, \zeta_i^3$, attacker can narrow the interval $[a_i, b_i]$ in which the x_i^j is located by using reverse interval mapping. This process will be shown in detail for x_i^1 which are calculated by function $f_{\beta_1}(x) = \beta_1 \cdot (\frac{1}{3} - x) \cdot x = -12x^2 + 4x$. Function $f_{\beta_1}(x)$ is defined in interval $x \in [0, \frac{1}{3})$ divided into two regions δ_1^1 and δ_2^1 split by $a_1^1 = \frac{1}{6}$.

Based on values of ζ_i^1 shown in Table 2, we can determine in which region of interval $[0, \frac{1}{3})$ is each x_i^1 located. We will begin with x_{52}^1 which is located in interval $[0, 0.166666666666667]$ ($\frac{1}{6}$ is represented with 0.166666666666667 when precision of 2^{53} is used) because $\zeta_{52}^1 = 0$. Bounds of interval in which next value x_{51}^1 is located can be obtained by using reverse interval mapping. If $\zeta_i^1 = 0$, we can calculate a_i by solving equation $a_{i+1} = f_{\beta_1}(a_i)$ and calculate b_i by solving equation $b_{i+1} = f_{\beta_1}(b_i)$. If $\zeta_i^1 = 1$, we can calculate a_i by solving equation $b_{i+1} = f_{\beta_1}(a_i)$ and calculate b_i by solving equation $a_{i+1} = f_{\beta_1}(b_i)$.

Value of $a_{52} = 0$ and because $\zeta_{51}^1 = 0$, value of a_{51} is calculated by solving equation $0 = f_{\beta_1}(a_{51}) = -12(a_{51})^2 + 4 \cdot a_{51}$. There are two solutions of this quadratic equation $a_{51} = 0$ and $a_{51} = \frac{1}{3}$. Because of $\zeta_{51}^1 = 0$, attacker know that $x_{51}^1 \in$

Table 1 Values of $\zeta_i^1, \zeta_i^2, \zeta_i^3$

i	Z_i	ζ_i^1	ζ_i^2	ζ_i^3
0	0	0	0	0
1	0	1	1	0
2	1	0	0	1
3	1	0	0	1
4	0	1	0	1
5	1	1	0	0
6	0	0	0	0
7	1	0	1	0
8	1	1	0	0
9	0	1	0	1
10	0	0	0	0
11	0	0	1	1
12	0	0	1	1
13	1	0	1	0
14	1	1	0	0
15	1	0	0	1
16	0	0	0	0
17	0	1	0	1
18	0	0	1	1
19	1	1	0	0
20	1	1	0	0
21	1	1	1	1
22	1	0	1	0
23	0	1	0	1
24	1	1	1	1
25	1	1	1	1
26	0	0	1	1
27	1	1	1	1
28	0	0	0	0
29	0	0	0	0
30	1	0	1	0
31	1	1	0	0
32	0	1	1	0
33	1	0	1	0
34	1	0	1	0
35	0	1	1	0
36	0	1	0	1
37	1	0	0	1
38	1	1	0	0
39	0	0	1	1
40	1	0	1	0
41	0	0	1	1
42	1	0	1	0
43	0	1	0	1
44	0	1	1	0
45	1	1	0	0
46	1	1	0	0
47	1	0	0	1
48	1	0	1	0
49	0	1	1	0
50	1	1	0	0
51	1	0	0	1
52	0	0	1	1

δ_1^1 and therefore only $0 \in [0, \frac{1}{6}]$ will be used as bound a_{51}. Value of $b_{52} = 0.166666666666667$ so value of b_{51} is calculated by solving equation $0.166666666666667 = -12(b_{51})^2 + 4 \cdot b_{51}$. Only one solution $b_{51} = 0.048815536468909$ of this quadratic equation is from interval δ_1^1 so attacker knows that x_{51} is located in interval [0, 0.048815536468909]. Bounds of interval in which x_i^1 is located is calculated in same manner based on values of $\zeta_i^1, a_{i+1}, b_{i+1}$ for all i from 50 down to 0. All values of a_i, b_i for $i < 53$ are shown in Table 2.

In Table 2, we can see that the interval $[a_i, b_i]$ is shorter for smaller i, for all $i > 5$. For $i = 5$ we have $a_5 = b_5$ which means that attacker found exact value of x_5^1 represented by precision of 2^{53}. Therefore, attacker can easily calculate all remaining values of x_i^1 and obtain initial secret value $x_0^1 = 0.149284547945563$. A similar procedure is applied on functions $f_{\beta_2}(x)$ and $f_{\beta_3}(x)$ in order to obtain secret values $x_0^2 = 0.519067023060468$, $x_0^3 = 0.881509362057897$. Therefore, when values of 53 consecutive output bits of analyzed PRNG are known, complexity of only 2^{106} is needed in order to recover initial values x_0^1, x_0^2, x_0^3.

3.4 Example 2 of cryptanalysis of the analyzed PRNG

In this section, example of the attack 2 will be described. Parameters of the analyzed PRNG will be the same as in previous example, $k = 3$, $x_0^1 = 0.149284547945563$, $x_0^2 = 0.519067023060468$, $x_0^3 = 0.881509362057897$. Consequently, user generates same sequence of 53 bits $Z = 0011010110000111000$

Table 2 Reverse interval mapping for x_0^1

i	ζ_i^1	a_i	b_i
0	0	0.149284547945563	0.149284547945563
1	1	0.329707676718519	0.329707676718519
2	0	0.0143448818285955	0.0143448818285955
3	0	0.0549102196982657	0.0549102196982657
4	1	0.183459292065321	0.183459292065321
5	1	0.329949426119979	0.329949426119979
6	0	0.013398218917072	0.013398218917078
7	0	0.05143872842649	0.051438728426512
8	1	0.17400360032035	0.17400360032041
9	1	0.33268736618806	0.33268736618807
10	0	0.0025788612984296	0.0025788612984403
11	0	0.01023563888656	0.010235638886602
12	0	0.039685335905247	0.039685335905404
13	0	0.13984223299004	0.13984223299052
14	1	0.32469873042847	0.32469873042878
15	0	0.033643735210394	0.03364373521154
16	0	0.12099212981469	0.12099212981835
17	1	0.30829937353362	0.30829937353764
18	0	0.092615449466147	0.092615449479803
19	1	0.26753034010679	0.26753034013106
20	1	0.21125156583656	0.2112515658953
21	1	0.30947957446261	0.30947957452546
22	0	0.088587013520321	0.088587013735732
23	1	0.26017614650789	0.26017614691155
24	1	0.22840505858532	0.22840505949122
25	1	0.28759378355063	0.28759378489292
26	0	0.15785291826326	0.15785292215892
27	1	0.33240114740231	0.33240114822636
28	0	0.0037183127990086	0.0037183160767644
29	0	0.014707340995179	0.014707353813697
30	0	0.056233693430936	0.056233740180374
31	1	0.18698803440113	0.18698815830541
32	1	0.32837777706425	0.32837783749414
33	0	0.019527300088638	0.019527534621082
34	0	0.073533414969531	0.073534243183796
35	1	0.22924770247315	0.22924955368782
36	1	0.2863339203585	0.28633670082198
37	0	0.16148232843653	0.16149031405959
38	1	0.33301080497872	0.33301179781758
39	0	0.0012849014419571	0.0012888651239656
40	0	0.0051197941072419	0.005135526416169
41	0	0.020164628928561	0.020225622085822
42	0	0.075779168596326	0.075993578878185
43	1	0.2342068856675	0.23467402714416
44	1	0.27783332038394	0.27859315914117
45	1	0.18300285672158	0.18503703454899
46	1	0.32928368833975	0.33013088006721
47	0	0.012686744581452	0.016001784479426
48	0	0.048815536468909	0.060934452639392
49	1	0.166666666666667	0.19918172033602
50	1	0.32064658875188	0.33333333333333
51	0	0	0.048815536468909
52	0	0	0.166666666666667

11110110100110110011010100111 10110 as in previous example.

Assume that the attacker knows the values of the first 53 output bits Z, but do not know initial values x_0^1, x_0^2, x_0^3. In this attack, attacker guesses values of x_0^2, x_0^3 and generates corresponding binary sequences ζ_i^2, ζ_i^3 (for $i < 53$). If the attacker tries all possible values of x_0^2, x_0^3, he will need a maximum of 2^{106} guesses in order to obtain initial values used for generation of sequence Z. When the search reaches the values of the users secret key $x_0^2 = 0.519067023060468$, $x_0^3 = 0.881509362057897$, attacker can generate corresponding binary sequences ζ_i^2, ζ_i^3 which are shown in Table 1. After that, attacker can calculate values of sequence ζ_i^1 (shown in Tables 1, 2) by using $\zeta_i^1 = Z_i \oplus \zeta_i^2 \oplus \zeta_i^3$.

Based on the values of ζ_i^1 for $i < 53$, attacker can calculate value of $x_0^1 = 0.149284547945563$ by using reverse interval mapping procedure described in Example 1. Because x_0^1 can be calculated on the basis of x_0^2, x_0^3 and known Z, this attack requires complexity of only 2^{106} in order to recover initial values (secret key) x_0^1, x_0^2, x_0^3.

3.5 Cryptanalysis of the gray-scale image cryptosystem based on the analyzed PRNG

In paper [1] a stream cipher algorithm for gray-scale images, based on analyzed PRNG, is proposed. The proposed cipher should be able to resist the chosen-plaintext attack. However, in this section, vulnerability of this cipher to known-plaintext attack will be demonstrated on example when $k = 3$. Consequently, analyzed cipher is vulnerable to chosen-plaintext attack as well.

If attacker know 53 consecutive plaintext bits $P_i, \ldots, P_{i+52}$ for some $i \geq 8$, then it is simple to obtain values of $Z_i, \ldots, Z_{i+52}$ by using Eq. 2. Based on Eq. 2, $Z_i = P_i \oplus C_{i-8} \oplus C_i$. When attacker know 53 consecutive output bits of analyzed PRNG, by applying one of the attacks from previous subsections, initial values (secret key) x_0^1, x_0^2, x_0^3 can be obtained with attack which complexity is only 2^{106}. One column of random values $R(l, 1) \in \{0, 1, \ldots, 255\}$ which is added to the plaintext image can to a lesser extent slow down this attack. Attacker can simply guess value of corresponding $R(l, 1) \in \{0, 1, \ldots, 255\}$, and ordinal number of byte of augmented image in which it is located. Because there are only 53 output bits that attacker try to reconstruct, there are only $\frac{53}{8}$ options for the location of this byte. Therefore, complexity of this attack, when only 53 bits of plaintext are known, is approximately $2^{106} \cdot 2^8 \cdot 2^3 = 2^{117}$ which is less than 2^{159} reported in [1] or 2^{128} recommended in [17]. Therefore, the example of analyzed cipher (when $k = 3$) cannot be considered safe.

Complexity of this attack for some $k > 0$ is $(2^{k-1})^n$. By using some $k > 3$, required complexity of this attack would reach over 2^{159}. However, for larger k, the number of weak keys significantly increases because there are $2 \cdot 4 \cdot \ldots \cdot 2(k-1) \cdot 2k$ combinations of fixed points for which analyzed PRNG would generate a sequence in which all bits would be equal to each other. The number of keys leading to this non-chaotic behavior is much greater. Therefore, there is no value of k for which analyzed cipher can be considered completely safe.

4 Conclusion

In this paper, a security analysis of the pseudo-random bit generator based on multi-modal maps [1] is presented. Unfortunately, some serious security problems are found. Depending on parameter k, there are certain number of fixed points and much greater number of values which lead to these fixed points of chaotic k-modal map, which can be considered as weak keys. Also, based on certain number of consecutive output bits, the initial values of this PRNG can be obtained with attack which complexity is significantly less than estimated key space. Although the assumed safety of the example of cipher based on this PRNG (when $k = 3$) is estimated at 2^{159}, it is possible to carry out successful known-plaintext attack with the complexity less than 2^{128}. For above-mentioned reasons, analyzed PRNG cannot be considered safe for the use in cryptographic systems.

References

1. Garcia-Martinez, M., Campos-Canton, E.: Pseudo-random bit generator based on multi-modal maps. Nonlinear Dyn. **82**, 2119–2131 (2015)
2. Lambić, D.: Security analysis and improvement of a block cipher with dynamic S-boxes based on tent map. Nonlinear Dyn. **79**, 2531–2539 (2015)
3. Arroyo, D., Alvarez, G., Amigo, J.M., Li, S.: Cryptanalysis of a family of self-synchronizing chaotic stream ciphers. Commun. Nonlinear Sci Numer. Simul. **16**, 805–813 (2011)
4. Alvarez, G., Amigo, J.M., Arroyo, D., Li, S.: Lessons learnt from the cryptanalysis of chaos-based ciphers. In: Kocarev, L.J., Lian, S. (eds.) Chaos-Based Cryptography: Theory, Algorithms and Applications, pp. 257–295. Springer, Heidelberg (2011)
5. Murillo-Escobar, M.A., Cruz-Hernandez, C., Cardoza-Avendano, L., Mendez-Ramirez, R.: A novel pseudorandom number generator based on pseudorandomly enhanced logistic map. Nonlinear Dyn. **87**, 407–425 (2017)
6. Lambić, D.: Cryptanalyzing a novel pseudorandom number generator based on pseudorandomly enhanced logistic map. Nonlinear Dyn. **89**, 2255–2257 (2017)
7. Matsumoto, M., Nishimura, T.: Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans Model. Comput. Simul. **8**(1), 3–30 (1998)
8. Patidar, V., Pareek, N., Sud, K.: A new substitution-diffusion based image cipher using chaotic standard and logistic maps. Commun. Nonlinear Sci. Numer. Simul. **14**(7), 3056–3075 (2009)
9. Rhouma, R., Solak, E., Belghith, S.: Cryptanalysis of a new substitution–diffusion based image cipher. Commun. Nonlinear Sci. Numer. Simul. **15**(7), 1887–1892 (2010)
10. Li, C.Q., Xie, T., Liu, Q., Cheng, G.: Cryptanalyzing image encryption using chaotic logistic map. Nonlinear Dyn. **78**(2), 1545–1551 (2014)
11. Huang, X.L.: Image encryption algorithm using chaotic Chebyshev generator. Nonlinear Dyn. **67**(4), 2411–2417 (2012)
12. Wang, X., Luan, D., Bao, X.: Cryptanalysis of an image encryption algorithm using Chebyshev generator. Digit. Signal Process. **25**, 244–247 (2014)
13. Alvarez, G., Li, S.: Cryptanalyzing a nonlinear chaotic algorithm (NCA) for image encryption. Commun. Nonlinear Sci. Numer. Simul. **14**(11), 3743–3749 (2009)
14. Ge, X., Lu, B., Liu, F., Luo, X.: Cryptanalyzing an image encryption algorithm with compound chaotic stream cipher based on perturbation. Nonlinear Dyn. (2017). https://doi.org/10.1007/s11071-017-3715-7

15. Chen, L., Ma, B., Zhao, X., Wang, S.: Differential cryptanalysis of a novel image encryption algorithm based on chaos and Line map. Nonlinear Dyn. **87**(3), 1797–1807 (2017)
16. Schneier, B.: Applied Cryptography. Wiley, New York (1996)
17. Ecrypt II yearly report on algorithms and keysizes (2010). http://www.ecrypt.eu.org/documents/D.SPA.13.pdf
18. Valtierra-Sanchez de la Vega, J.L., Tlelo-Cuautle, E.: Simulation of piecewise-linear one-dimensional chaotic maps by Verilog-A. IETE Tech. Rev. **32**(4), 304–310 (2015)
19. Valtierra-Sanchez de la Vega, J.L., ETlelo-Cuautle, E., Rodrguez-Vzquez, A.: A switched-capacitor skew-tent map implementation for random number generation. Int. J. Circuit Theory Appl. Special Issue: Secure lightweight cryptohardware, **45**(2), 305–315 (2017)
20. Lanford III, O.E.: Some informal remarks on the orbit structure of discrete approximations to chaotic maps. Exp. Math. **7**(4), 317–324 (1998)
21. de la Fraga, L.G., Torres-Prez, E., Tlelo-Cuautle, E., Mancillas-Lpez, C.: Hardware implementation of pseudo-random number generators based on chaotic maps. Nonlinear Dyn. (2017). https://doi.org/10.1007/s11071-017-3755-z
22. Steeb, W.H.: The Nonlinear Workbook. World Scientific, Singapore (2014)