A Weakness of the Linear Part of Stream Cipher MUGI

Jovan Dj. Golić

System on Chip, Telecom Italia Lab, Telecom Italia
Via Reiss Romoli 274, I-10148 Turin, Italy
jovan.golic@tilab.com

Abstract. The linearly updated component of the stream cipher MUGI, called the buffer, is analyzed theoretically by using the generating function method. In particular, it is proven that the intrinsic response of the buffer, without the feedback from the nonlinearly updated component, consists of binary linear recurring sequences with small linear complexity 32 and with extremely small period 48. It is then shown how this weakness can in principle be used to facilitate the linear cryptanalysis of MUGI with two main objectives: to reconstruct the secret key and to find linear statistical distinguishers.

Keywords: Stream ciphers, combiners with memory, linear finite-state machines, linear cryptanalysis.

1 Introduction

MUGI is a specific keystream generator for stream cipher applications proposed in [8]. Due to its design rationale, it is suitable for software implementations. Efficient hardware implementations in terms of speed are also possible, but are more complex with respect to the gate count than the usual designs based on linear feedback shift registers (LFSRs), nonlinear combining functions, and irregular clocking. MUGI has been evaluated within the CRYPTREC project of the Information-technology Promotion Agency for possible electronic government applications in Japan.

It is interesting to note that in mathematical terms, the structure of MUGI as well as of PANAMA, which is a stream cipher of a similar type previously proposed in [1], is essentially one of a combiner with memory, which is a well-known type of keystream generators (see [5]). Specific features are the following:

- the nonlinear combining function has a large internal memory size and is based on a round function of the block cipher AES [2]
- the driving linear finite-state machine (LFSM) providing input to the combining function is not an LFSR with a primitive connection polynomial
- the LFSM receives feedback from a part of the internal memory of the combining function

– the output at a given time is a binary word taken from the internal memory of the combining function and then bitwise added to the plaintext word to produce the ciphertext word.

A security analysis of MUGI is presented in [9] and [10]. The main claims are essentially that MUGI is not vulnerable to common attacks on block ciphers and also to some attacks on stream ciphers. In particular, the linear cryptanalysis method for block ciphers [7] is adapted to deal with MUGI and to show that particular linear approximations cannot be effective for MUGI. However, some general methods for analyzing stream ciphers based on combiners with memory, most notably the so-called linear cryptanalysis of stream ciphers [3], [4], [5], and [6], are not addressed. Since MUGI can essentially be regarded as a combiner with memory, such methods are in principle also applicable to MUGI. Also, the underlying LFSM of MUGI, the so-called buffer, is not analyzed in [9] and [10].

Recall that linear cryptanalysis of product block ciphers composed of an iterated round function is based on the fact that the input and output to the block cipher are known, in the known plaintext scenario, and that all the intermediate outputs are unknown. Linear cryptanalysis of stream ciphers is essentially different from the linear cryptanalysis of block ciphers because of the underlying iterative structure in which the initial state is unknown and the output sequence produced from a sequence of internal states is known, in the known plaintext scenario. It essentially consists in finding linear relations among the unknown internal variables, possibly conditioned on the known output sequence, that hold with probabilities different from one half. It has two main objectives:

– to reconstruct the initial state of the keystream generator as well as the secret key
– to derive a linear statistical distinguisher which can distinguish the output sequence from a purely random sequence, defined as a sequence of mutually independent uniformly distributed random variables.

The main objective of this paper is to show that the linear part of MUGI, that is, the buffer is analyzable and that it is surprisingly weak. The second objective is to investigate how this weakness can in principle be used to facilitate the cryptanalysis of MUGI, especially the linear cryptanalysis.

The paper is organized as follows. Section 2 contains a brief description of MUGI. Analysis of the LFSM of MUGI is presented in Section 3, with some elements shown in the Appendix, a related transformation that eliminates the LFSM from the underlying system of nonlinear recurrences is given in Section 4, and the framework for the linear cryptanalysis of MUGI is outlined in Section 5. Section 6 contains a summary of the established weaknesses of MUGI and problems for future investigation.

2 Description of MUGI

A concise description of MUGI is specified here in as much detail as needed for the analysis. More details can be found in [8].

The keystream generator is essentially a combiner with memory, with specific properties described in Section 1. It is a finite-state machine (FSM) whose internal state has two components:

– a linearly updated component, called buffer, $b = b_0 b_1 \cdots b_{15}$, where each b_i is a 64-bit word; the size of this component is 1024 bits
– a nonlinearly updated component, called state, $a = a_0 a_1 a_2$, where each a_i is a 64-bit word; the size of this component is 192 bits.

The next-state or update function is invertible and has two components, $\phi = (\rho, \lambda)$, where ρ updates a and λ updates b, that is, $(a^{(t+1)}, b^{(t+1)}) = (\rho(a^{(t)}, b^{(t)}), \lambda(a^{(t)}, b^{(t)}))$.

The ρ component is a nonlinear function defined in terms of an invertible $(64 \times 64)-$bit function F by a kind of Feistel structure. More precisely:

$$\begin{aligned} a_0^{(t+1)} &= a_1^{(t)} \\ a_1^{(t+1)} &= a_2^{(t)} \oplus F(a_1^{(t)} \oplus b_4^{(t)}) \oplus C_1 \\ a_2^{(t+1)} &= a_0^{(t)} \oplus F(a_1^{(t)} \oplus {}^{<17}b_{10}^{(t)}) \oplus C_2. \end{aligned} \tag{1}$$

The ρ function is invertible for any given $b_4^{(t)}$ and ${}^{<17}b_{10}^{(t)}$. Here, for a 64-bit word x, ${}^{<i}x$ and ${}^{>i}x$ denote the rotations of x by i bits to the left and right, respectively. The function F is derived from the round function of AES and is a composition of (G, G) and a permutation of 8 bytes, where a $(32 \times 32)-$bit function G is a composition of a parallel combination of 4 $(8 \times 8)-$bit S-boxes and a linear $(32 \times 32)-$bit function, MixColumn, of AES (see [2]). The byte-permutation is (4,5,2,3,0,1,6,7) and C_1 and C_2 are 64-bit constants.

The λ component is a linear function defined by the following equations:

$$\begin{aligned} b_i^{(t+1)} &= b_{i-1}^{(t)}, \quad i \neq 0, 4, 10 \\ b_0^{(t+1)} &= b_{15}^{(t)} \oplus a_0^{(t)} \\ b_4^{(t+1)} &= b_3^{(t)} \oplus b_7^{(t)} \\ b_{10}^{(t+1)} &= b_9^{(t)} \oplus {}^{<32}b_{13}^{(t)}. \end{aligned} \tag{2}$$

The λ function is invertible for any given $a_0^{(t)}$.

The 64-bit output of the keystream generator at time t is defined as $a_2^{(t)}$.

The initial internal state of the keystream generator is produced from a 128-bit secret key K and a 128-bit initialization vector IV in three stages, by using the keystream generator itself. In the first two stages only ρ is used and in the third stage both ρ and λ are used. At the beginning of the third stage, the initial value of a, $a^{(0)}$, depends on both K and IV, but the initial value of the buffer b, $b^{(0)}$, depends on K only. The third stage consists of iterating ϕ (i.e., both ρ and λ) 15 times, without producing the output $(a_2^{(t)})_{t=0}^{15}$, and the keystream generation starts from the 16-th iteration on.

3 Analysis of Buffer

In this section, the buffer is analyzed as a non-autonomous LFSM with one input sequence, namely, $a_0 = (a_0^{(t)})_{t=0}^{\infty}$. The input sequence and all the internal sequences in the buffer are 64-bit sequences. Our objective is to derive expressions for the internal sequences in the buffer in terms of the input sequence a_0 and the initial state of the buffer, $b^{(0)} = b_0^{(0)} b_1^{(0)} \cdots b_{15}^{(0)}$.

In view of the λ update function, the 16 internal sequences in the buffer can be divided in three groups, in each group the sequences being phase shifts of each other (see Fig. 1, where R_j denotes the rotation by j bits to the left, which is a linear transformation of a 64-bit word).

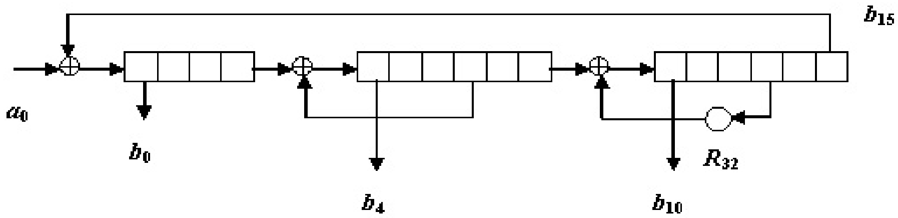

Fig. 1. The buffer as LFSM.

3.1 Linear Recurrences

¿From the λ update function, we directly obtain the following linear recurrences, all holding for $t \geq 1$:

$$\begin{aligned} b_4^{(t)} &= b_4^{(t-4)} \oplus b_0^{(t-4)} \\ b_{10}^{(t)} &= R_{32} b_{10}^{(t-4)} \oplus b_4^{(t-6)} \\ b_0^{(t)} &= a_0^{(t-1)} \oplus b_{10}^{(t-6)}. \end{aligned} \tag{3}$$

In vectorial notation where vectors are represented as one-column matrices, R_{32} is represented as a matrix. The initial state of the buffer can now be represented as $(b_0^{(t)})_{t=-3}^{0} (b_4^{(t)})_{t=-5}^{0} (b_{10}^{(t)})_{t=-5}^{0}$. Then, by eliminating b_0, we obtain

$$\begin{aligned} b_4^{(t)} &= b_4^{(t-4)} \oplus b_{10}^{(t-10)} \oplus a_0^{(t-5)}, \quad t \geq 5 \\ b_{10}^{(t)} &= b_4^{(t-6)} \oplus R_{32} b_{10}^{(t-4)}, \quad t \geq 1. \end{aligned} \tag{4}$$

This is a system of two 64-bit linear recurrences (that is, 128 binary linear recurrences) in terms of 64-bit sequences b_4 and b_{10}, where a_0 is regarded as a given 64-bit sequence.

3.2 Generating Functions

The system can be solved by using the generating function technique dealing with the z-transforms of 64-bit sequences. In vectorial notation, the z-transforms or generating functions of b_4, b_{10}, and a_0 are defined as formal power series

$$B_4 = \sum_{t=0}^{\infty} b_4^{(t)} z^t, \quad B_{10} = \sum_{t=0}^{\infty} b_{10}^{(t)} z^t, \quad A_0 = \sum_{t=0}^{\infty} a_0^{(t)} z^t, \tag{5}$$

respectively. It is shown in the Appendix how to convert the system (4) of linear recurrences from the time domain into the generating function domain. Namely, letting I denote the 64×64 identity matrix, we thus obtain the representation

$$(1 \oplus z^4)B_4 \oplus z^{10} B_{10} \;=\; z^5 A_0 \oplus \Delta_1 \tag{6}$$

$$z^6 B_4 \oplus (I \oplus z^4 R_{32}) B_{10} \;=\; \Delta_2 \tag{7}$$

where

$$\Delta_1 \;=\; b_4^{(0)} \oplus z^4 b_0^{(0)} \oplus \sum_{t=1}^{3} (b_4^{(t-4)} \oplus b_0^{(t-4)}) z^t \oplus \sum_{t=5}^{9} b_{10}^{(t-10)} z^t \tag{8}$$

$$\Delta_2 \;=\; \sum_{t=1}^{5} b_4^{(t-6)} z^t \oplus b_{10}^{(0)} \oplus R_{32} \sum_{t=1}^{3} b_{10}^{(t-4)} z^t \tag{9}$$

are 64-dimensional vectors (64×1 matrices) whose elements are polynomials in z defined by the initial state of the buffer and whose degrees are at most 9 and 5, respectively. Essentially, we obtain a system of 128 linear equations with coefficients being polynomials in z and with unknowns being 128 generating functions of 64 binary sequences in b_4 and 64 binary sequences in b_{10}.

3.3 Solution

The system has a unique solution which can be found in the following way. First, by elimination we obtain

$$F(z)B_4 \;=\; z^5 (I \oplus z^4 R_{32}) A_0 \oplus (I \oplus z^4 R_{32}) \Delta_1 \oplus z^{10} \Delta_2 \tag{10}$$

$$F(z)B_{10} \;=\; z^{11} A_0 \oplus z^6 \Delta_1 \oplus (1 \oplus z^4) \Delta_2 \tag{11}$$

where

$$F(z) \;=\; I \oplus z^4 (I \oplus R_{32}) \oplus z^8 R_{32} \oplus z^{16} I \tag{12}$$

denotes a 64×64 matrix whose coefficients are polynomials in z of degree at most 16.

When regarded over a field of rational functions in z, $F(z)$ is invertible as is seen from the following equation:

$$\begin{aligned} F(z)F(z) &= (I \oplus z^4(I \oplus R_{32}) \oplus z^8 R_{32} \oplus z^{16} I)^2 \\ &= I \oplus z^8(I \oplus R_{32}^2) \oplus z^{16} R_{32}^2 \oplus z^{32} I \\ &= I \oplus z^8(I \oplus I) \oplus z^{16} I \oplus z^{32} I \;=\; (1 \oplus z \oplus z^2)^{16} I \end{aligned} \tag{13}$$

because of $R_{32}^2 = I$. Thus we get that $F(z)/f(z)$ is the inverse of $F(z)$, where

$$f(z) \;=\; 1 \oplus z^{16} \oplus z^{32} \;=\; (1 \oplus z \oplus z^2)^{16} \;=\; \frac{1 \oplus z^{48}}{1 \oplus z^{16}}. \tag{14}$$

Accordingly, we obtain the solution for the generating functions B_4 and B_{10} in the form of

$$\begin{aligned} B_4 &= \frac{1}{f(z)} z^5 G(z) A_0 \oplus \frac{1}{f(z)} \Delta_1' \\ &= \frac{1}{1 \oplus z^{48}} z^5 (1 \oplus z^{16}) G(z) A_0 \oplus \frac{1}{1 \oplus z^{48}} (1 \oplus z^{16}) \Delta_1' \end{aligned} \tag{15}$$

$$\begin{aligned} B_{10} &= \frac{1}{f(z)} z^{11} F(z) A_0 \oplus \frac{1}{f(z)} \Delta_2' \\ &= \frac{1}{1 \oplus z^{48}} z^{11} (1 \oplus z^{16}) F(z) A_0 \oplus \frac{1}{1 \oplus z^{48}} (1 \oplus z^{16}) \Delta_2' \end{aligned} \tag{16}$$

where

$$G(z) \;=\; F(z)(I \oplus z^4 R_{32}) \;=\; (1 \oplus z \oplus z^2 \oplus z^3 \oplus z^4)^4 I \oplus z^{20} R_{32} \tag{17}$$

denotes a 64×64 matrix whose coefficients are polynomials in z of degree at most 20, and

$$\Delta_1' \;=\; G(z)\Delta_1 \oplus z^{10} F(z) \Delta_2 \tag{18}$$

$$\Delta_2' \;=\; F(z)(z^6 \Delta_1 \oplus (1 \oplus z^4)\Delta_2) \tag{19}$$

are 64-dimensional vectors (64×1 matrices) whose elements are polynomials in z defined by the initial state of the buffer and whose degrees are at most 31.

3.4 Discussion

Both b_4 and b_{10} have two components, one being a linear transform of the input sequence a_0 to the buffer and the other being a linear transform of the initial conditions contained in Δ_1 and Δ_2. For both b_4 and b_{10}, the other, intrinsic component consists of 64 binary linear recurring subsequences each produced by an LFSR with the feedback polynomial $f(z)$, or alternatively, by a cycling LFSR with the feedback polynomial $1 \oplus z^{48}$. The following properties should then be considered as serious weaknesses of the buffer design.

– The exponent (period) of $f(z)$ is only 48; the period of each of the intrinsic binary subsequences is thus equal to 48 or divides 48.
– The degree of $f(z)$ is only 32, and with an appropriate design it could have been as large as 1024, which is the bit-size of the internal state of the buffer.
– Due to their extremely small period, the statistical properties of the intrinsic binary subsequences are very bad.
– In common designs of keystream generators, the linear component, with the feedback from the nonlinear component disconnected, ensures a large period of the corresponding internal state sequence which itself very likely provides a lower bound on the period of the keystream sequence as well as good statistical properties. Consequently, the design of MUGI does not satisfy this criterion.
– The polynomial $f(z)$ defines a linear sequential transform of any buffer sequence, in particular b_4 or b_{10}, that is equal to a linear sequential transform of the input sequence, which is produced by the nonlinear component. Its low degree, 32, small number of nonzero coefficients, 3, and small period, 48, facilitate the initial state reconstruction and finding statistical distinguishers for the keystream sequence (see Sections 4 and 5).
– The intrinsic binary subsequences do not depend on the initialization vector, but only on the secret key. Namely, for each $1 \leq j \leq 64$, the j-th binary subsequence of both b_4 and b_{10} solely depends on $(b_{i,j}^{(0)})_{i=0}^{15}(b_{i,(j+32)_{\bmod 64}}^{(0)})_{i=0}^{15}$, that is, on the j-th and the $(j+32)_{\bmod 64}$-th binary subsequences of $b^{(0)}$, which are defined by 32 secret key bits only. This is because the mixing between different binary subsequences in the buffer, provided by the linear transform R_{32}, is not good. Divide-and-conquer secret key reconstruction attacks may be facilitated by this property.

4 Elimination of Buffer

The obtained expressions (15) and (16) for the generating functions of the 64-bit buffer sequences b_4 and b_{10}, respectively, can be transformed into the time domain and then appropriately substituted in the recurrences (1) for the update function ρ. In this way, we can derive the recurrences involving only the state sequences a_1 and a_2, where the output sequence a_2 is assumed to be known, in the known plaintext scenario, except for the first 16 outputs $(a_2^{(t)})_{t=0}^{15}$, which are discarded. Namely, from (1) we first eliminate a_0 and use the fact that F is invertible to get for $t \geq 0$

$$a_1^{(t)} \oplus b_4^{(t)} = F^{-1}(a_1^{(t+1)} \oplus a_2^{(t)} \oplus C_1) \tag{20}$$

$$a_1^{(t)} \oplus {}^{<17}b_{10}^{(t)} = F^{-1}(a_1^{(t-1)} \oplus a_2^{(t+1)} \oplus C_2) \tag{21}$$

where F^{-1} is the inverse of F and formally $a_1^{(-1)} = a_0^{(0)}$. Then, by converting (15) and (16) into the time domain we get the following linear recurrences holding for $t \geq 48$:

$$\begin{aligned} b_4^{(t)} \oplus b_4^{(t-48)} &= a_0^{(t-5)} \oplus a_0^{(t-9)} \oplus a_0^{(t-13)} \oplus a_0^{(t-17)} \oplus a_0^{(t-25)} \oplus a_0^{(t-29)} \\ &\oplus a_0^{(t-33)} \oplus a_0^{(t-37)} \oplus {}^{<32}a_0^{(t-25)} \oplus {}^{<32}a_0^{(t-41)} \end{aligned} \tag{22}$$

$$\begin{aligned} b_{10}^{(t)} \oplus b_{10}^{(t-48)} &= a_0^{(t-11)} \oplus a_0^{(t-15)} \oplus a_0^{(t-31)} \oplus a_0^{(t-43)} \oplus {}^{<32}a_0^{(t-15)} \\ &\oplus {}^{<32}a_0^{(t-19)} \oplus {}^{<32}a_0^{(t-31)} \oplus {}^{<32}a_0^{(t-35)}. \end{aligned} \tag{23}$$

Finally, by combining (20) with (22) and (21) with (23), we get the following recurrences involving a_1 and a_2 only, which hold for $t \geq 48$:

$$\begin{aligned} &a_1^{(t)} \oplus a_1^{(t-48)} \oplus F^{-1}(a_1^{(t+1)} \oplus a_2^{(t)} \oplus C_1) \oplus F^{-1}(a_1^{(t-47)} \oplus a_2^{(t-48)} \oplus C_1) = \\ &a_1^{(t-6)} \oplus a_1^{(t-10)} \oplus a_1^{(t-14)} \oplus a_1^{(t-18)} \oplus a_1^{(t-26)} \oplus a_1^{(t-30)} \oplus a_1^{(t-34)} \oplus a_1^{(t-38)} \\ &\oplus {}^{<32}a_1^{(t-26)} \oplus {}^{<32}a_1^{(t-42)} \end{aligned} \tag{24}$$

$$\begin{aligned} &a_1^{(t)} \oplus a_1^{(t-48)} \oplus F^{-1}(a_1^{(t-1)} \oplus a_2^{(t+1)} \oplus C_2) \oplus F^{-1}(a_1^{(t-49)} \oplus a_2^{(t-47)} \oplus C_2) = \\ &{}^{<17}a_1^{(t-12)} \oplus {}^{<17}a_1^{(t-16)} \oplus {}^{<17}a_1^{(t-32)} \oplus {}^{<17}a_1^{(t-44)} \oplus {}^{<49}a_1^{(t-16)} \\ &\oplus {}^{<49}a_1^{(t-20)} \oplus {}^{<49}a_1^{(t-32)} \oplus {}^{<49}a_1^{(t-36)}. \end{aligned} \tag{25}$$

The recurrences are nonlinear because of nonlinear F^{-1}. As the first 16 outputs are not known, it is interesting to consider (24) and (25) only for $t \geq 64$.

One can also obtain different recurrences by using the polynomial $f(z)$ of degree 32 instead of the polynomial $1 \oplus z^{48}$. They will hold for $t \geq 32$, but each will involve F^{-1} three times which makes them less useful. In principle, there are at least two general ways of using the recurrences (24) and (25).

- One way is to try to eliminate a_1, possibly with certain approximation probabilities, thus yielding a recurrence in a_2 holding with a certain probability which will represent a statistical distinguisher between the keystream sequence and a purely random sequence (see Section 5).
- The other way is to assume that a_2 is known, in the known plaintext scenario, and try to solve the corresponding nonlinear equations for a_1, possibly by an algebraic approach or a probabilistic approach using approximations to the nonlinear equations.

5 Linear Cryptanalysis of MUGI

A general way of conducting the linear cryptanalysis of stream ciphers is to linearize the (vectorial) next-state function and the output function, with certain approximation probabilities, and to analyze the LFSM resulting from these linear approximations (see [3], [4], [5], and [6]). In particular, this method is also applicable if only one bit of the internal state is known at a time. This is essentially different from the linear cryptanalysis of iterated block ciphers where one concatenates mutually correlated linear functions of the input and output to the round function. The obtained LFSM is in fact an LFSM approximation to the keystream generator, which itself is a nonlinear autonomous FSM. Our objective here is to present a framework for conducting the linear cryptanalysis of MUGI by using the results from Sections 3 and 4.

Linearizing the next-state function of MUGI reduces to linearizing the nonlinear function F or its inverse F^{-1}. More precisely, we linearize equations (20) and (21), where, for convenience, $t-1$ is substituted for t, and the sequences b_4 and b_{10} are determined by (15) and (16), respectively. In turn, linearizing F^{-1} reduces to linearizing the S-boxes of AES. The effectiveness of the linear cryptanalysis depends on the underlying linear approximations and the corresponding approximation probabilities.

A linear approximation to an S-box is a pair of input, α, and output, β, linear functions with a nonzero correlation coefficient $c(\alpha,\beta) = \Pr(\alpha=\beta)-\Pr(\alpha\neq\beta)$. It is well known that the maximal correlation coefficient magnitude is 1/8. This value is relatively small, but allows a lot of freedom when choosing the linear approximations. In particular, for any given α, there are 5 values of β with $c(\alpha,\beta) = 1/8$, 16 values of β with $c(\alpha,\beta) = 7/64$, and 36 values of β with $c(\alpha,\beta) = 6/64$.

A linear approximation to F^{-1} consists of linear approximations to the 64 component Boolean functions and each of the 64 correlation coefficients is determined by the corresponding active S-box. More generally, one can also consider linear approximations to any 64 linearly independent linear combinations of the 64 component Boolean functions of F^{-1}. In this case more than just one S-box can be active for each linear combination considered, so that the resulting correlation coefficient is the product of the involved correlation coefficients of active S-boxes. In the more general scenario, let L_2 define a set of 64 linear approximations to 64 linearly independent linear combinations of the component Boolean functions of F^{-1} defined by L_1F^{-1}. More precisely, an invertible matrix L_1 is first applied to both (20) and (21) and then a matrix L_2 is substituted for L_1F^{-1} on the right-hand sides of the equations. We thus get

$$L_1a_1^{(t-1)} \oplus L_1b_4^{(t-1)} = L_2a_1^{(t)} \oplus L_2a_2^{(t-1)} \oplus L_2C_1 \oplus e_1^{(t)} \tag{26}$$

$$L_1a_1^{(t-1)} \oplus L_1{}^{<17}b_{10}^{(t-1)} = L_2a_1^{(t-2)} \oplus L_2a_2^{(t)} \oplus L_2C_2 \oplus e_2^{(t)} \tag{27}$$

where e_1 and e_2 are the 64-bit approximation-error sequences whose binary component subsequences are expressed as nonbalanced Boolean functions of the corresponding inputs to F^{-1}. The greater the imbalance, the better the underlying linear approximations.

Equations (26) and (27) in fact define an LFSM with input sequences e_1 and e_2. The LFSM can be solved for a_1 and a_2 by using the generating function method. Let A_1, A_2, E_1, E_2, and $A_0 = zA_1 \oplus a_0^{(0)}$ denote the generating functions of a_1, a_2, e_1, e_2, and a_0, respectively. Then after some algebraic manipulations, by using the expressions (15) and (16) for B_4 and B_{10}, in terms of $1 \oplus z^{48}$, respectively, we get

$$F_1(z)A_1 = z(1 \oplus z^{48})L_2A_2 \oplus (1 \oplus z^{48})E_1 \oplus \Delta_1'' \oplus (1 \oplus z^{48})C_1' \tag{28}$$

$$F_2(z)A_1 = (1 \oplus z^{48})L_2A_2 \oplus (1 \oplus z^{48})E_2 \oplus \Delta_2'' \oplus (1 \oplus z^{48})C_2' \tag{29}$$

where $C_1' = C_1/(1 \oplus z)$ and $C_2' = C_2/(1 \oplus z)$ denote the generating functions of the constant sequences, of period equal to 1, corresponding to the constants C_1 and C_2, respectively, and

$$F_1(z) = (1 \oplus z^{48})(zL_1 \oplus L_2) \oplus z^7(1 \oplus z^{16})L_1G(z) \tag{30}$$

$$F_2(z) = z\left((1 \oplus z^{48})(L_1 \oplus zL_2) \oplus z^{12}(1 \oplus z^{16})L_1R_{17}F(z)\right) \tag{31}$$

$$\Delta_1'' = (1 \oplus z^{16})(zL_1\Delta_1' \oplus z^6L_1G(z)a_0^{(0)}) \oplus (1 \oplus z^{48})L_2a_1^{(0)} \tag{32}$$

$$\Delta_2'' = z(1 \oplus z^{16})L_1R_{17}\Delta_2' \oplus z\left(z^{11}(1 \oplus z^{16})L_1R_{17}F(z) \oplus (1 \oplus z^{48})L_2\right)a_0^{(0)}$$
$$\oplus (1 \oplus z^{48})L_2a_2^{(0)}. \tag{33}$$

Note that Δ_1'' and Δ_2'' are 64-dimensional vectors whose elements are polynomials in z defined by the initial state of the whole keystream generator ($b^{(0)}$ and $a_0^{(0)}a_1^{(0)}a_2^{(0)}$) and whose degrees are at most 49. Here E_1 and E_2 are generating functions of unknown, but nonbalanced sequences, so that (28) and (29) in fact constitute a system of binary linear recurrences each holding with a probability different from one half.

To eliminate the unknown A_1 from the system, assume for simplicity that $F_1(z)$ is invertible, where $F_1(z)^{-1} = F_1(z)^*/\det F_1(z)$, $F_1(z)^*$ being the adjunct matrix of $F_1(z)$. (As L_1 is invertible, it is likely that $F_1(z)$ or $F_2(z)$ are invertible.) We then obtain a correlation equation

$$\frac{F_2(z)F_1(z)^*\Delta_1'' \oplus \det F_1(z)\Delta_2''}{1 \oplus z^{48}} = (F_2(z)F_1(z)^*E_1 \oplus \det F_1(z)E_2)$$

$$\oplus\ (zF_2(z)F_1(z)^* \oplus \det F_1(z)I)\,L_2A_2 \oplus (F_2(z)F_1(z)^*C_1' \oplus \det F_1(z)C_2')\,. \tag{34}$$

In the time domain, the left-hand side of (34) is a 64-bit sequence, x , in the generating function domain denoted as X, which depends on the initial conditions, and the last two terms on the right-hand side of (34) are linear sequential transforms of the (known) sequence a_2 and of the constant sequences corresponding to C_1 and C_2 , respectively. The first term on the right-hand side of (34), $E = F_2(z)F_1(z)^*E_1 \oplus \det F_1(z)E_2$, is the noise term depending on the performed linear approximations. Consequently, (34) means that the linear recurring sequence x depending on the initial conditions which is ultimately periodic with period of only 48 (or smaller) is termwise correlated to a linear sequential transform of a_2 and of the constant sequences corresponding to C_1 and C_2. Equivalently, this is the case for each of the 64 constituent binary subsequences.

5.1 Initial State Reconstruction

If $t = 0$ is taken as the initial time, then the first 16 elements of the output sequence, $(a_2^{(t)})_{t=0}^{15}$, are unknown, and the initial state of the buffer, $b^{(0)}$, represented through Δ_1 and Δ_2, depends on the secret key only. Alternatively, if $t = 16$ is taken as the initial time, then the output sequence is known, but the initial state of the buffer, $b^{(16)}$, depends on the initialization vector as well. Note that the initial state of MUGI, $b^{(0)}$ and $a_0^{(0)}a_1^{(0)}a_2^{(0)}$, can be obtained from any internal state of MUGI (e.g., $b^{(16)}$ and $a_0^{(16)}a_1^{(16)}a_2^{(16)}$) by reversing the next-state function. Furthermore, the 128-bit secret key can be obtained from $b_0^{(0)}b_1^{(0)}$ by reversing the update function ρ.

The effectiveness of the correlation equation (34) is determined by how much the probabilities for the 64 underlying binary noise subsequences of the resulting noise sequence e deviate from one half. Let $c_i = 1 - 2p_i$ denote the (positive or negative) correlation coefficient of the i-th binary noise subsequence of e, where p_i is the probability that the noise bit is equal to 1. The correlation coefficients of the involved approximation-error sequences e_1 and e_2 depend on the linearization of S-boxes and their magnitudes are equal to 2^{-3} or are close to this value if only one S-box is active per each linear approximation involved. If we assume that these subsequences are mutually independent sequences of mutually independent and identically distributed binary random variables, then the correlation coefficient magnitude is given as $|c_i| = 2^{-3m_i}$ where m_i is the total number of binary terms from e_1 and e_2 present in this subsequence, that is, the total number of nonzero binary coefficients of the polynomials in the i-th row of $F_2(z)F_1(z)^*$ and in $\det F_1(z)$. Here we used a well-known fact that the correlation coefficient of a binary sum of mutually independent binary random variables is equal to the product of their individual correlation coefficients (see also the piling-up lemma from [7]).

Accordingly, the periodic part of the 64-bit linear recurring sequence x, that is, any corresponding segment composed of 48 consecutive 64-bit words can in principle be statistically reconstructed by a sort of repetition attack based on the fact that any bit of the periodic part of x is repeated with period 48. If c denotes the underlying correlation coefficient for a considered bit of x, then the bit can be reconstructed by the mojority count from the given output segment of length $O(48c^{-2})$ with complexity $O(c^{-2})$. It is easy to see that the periodic part of x depends (linearly) only on $b^{(0)}$ and $a_0^{(0)}$ and not on $a_1^{(0)}$. Namely, this is due to

$$X = \frac{(1 \oplus z^{16}) F_2(z) F_1(z)^* \left(z L_1 \Delta_1' \oplus z^6 L_1 G(z) a_0^{(0)} \right)}{1 \oplus z^{48}}$$

$$\oplus \frac{\det F_1(z)(1 \oplus z^{16}) \left(z L_1 R_{17} \Delta_2' \oplus z^{12} L_1 R_{17} F(z) a_0^{(0)} \right)}{1 \oplus z^{48}}$$

$$\oplus F_2(z) F_1(z)^* L_2 a_1^{(0)} \oplus \det F_1(z) \left(z L_2 a_0^{(0)} \oplus L_2 a_2^{(0)} \right) \tag{35}$$

where Δ_1' and Δ_2' depend only on $b^{(0)}$. Accordingly, the initial time of the 48-word segment of x to be reconstructed should on one hand be sufficiently large so as to render the terms depending on $a_1^{(0)}$ and $a_2^{(0)}$ in (35) vanish and, on the other, should be at least 16 so that the involved terms of the output sequence a_2 in (34) are all known.

The recovered 48 64-bit words of the periodic part of x define a system of $48 \cdot 64$ binary linear equations in the unknown $17 \cdot 64$ initial state bits, $b^{(0)}$ and $a_0^{(0)}$, which can then be obtained by solving the system. The 128-bit secret key can be obtained from $b_0^{(0)} b_1^{(0)}$ by reversing the update function ρ.

Alternatively, if the objective is to recover the initial state of MUGI, regardless of the initialization algorithm and the secret key, but taking into account the fact that the first 16 elements of the output sequence, $(a_2^{(t)})_{t=0}^{15}$, are unknown, then $t = 16$ should be taken as the initial time, $b^{(16)}$ and $a_0^{(16)}$ should be reconstructed by the repetition attack described above, and the 64 bits of $a_1^{(16)}$ can be reconstructed by the exhaustive search in view of the fact that $a_2^{(16)}$ is known. Finally, the initial state, $b^{(0)}$ and $a_0^{(0)} a_1^{(0)} a_2^{(0)}$, is then obtained from the internal state $b^{(16)}$ and $a_0^{(16)} a_1^{(16)} a_2^{(16)}$ by reversing the next-state function.

The feasibility of the attack can be estimated by computing the noise correlation coefficients, as described above, for different vectorial linear transformations L_1 and L_2. Assuming for simplicity that the underlying correlation coefficient $|c| = 2^{-3m}$ is the same for all the bits of x to be reconstructed, the attack would in theory be effective if it is faster than the exhaustive search over the initial states. Note that the exhaustive search requires 18 64-bit output values to be produced for each guessed initial state or, more precisely, for each guessed internal state at time $t = 16$. Roughly speaking, the attack is effective if $48 \cdot 2^{6m} < 2^{18 \cdot 64}$, that is, if $m \leq 191$, where the underlying complexity units are assumed to be the same. Of course, it has to be noted that the required output sequence length would be of the same order of magnitude as the complexity.

5.2 Linear Statistical Weakness

The equation (34) can also be put into the form

$$L(z)A_2 \;=\; \frac{1 \oplus z^{48}}{1 \oplus z}\,(F_2(z)F_1(z)^* C_1 \oplus \det F_1(z)C_2)$$

$$\oplus\; (F_2(z)F_1(z)^* \Delta_1'' \oplus \det F_1(z)\Delta_2'') \oplus (1 \oplus z^{48})E \qquad (36)$$

where the matrix $L(z) = (1 \oplus z^{48})(zF_2(z)F_1(z)^* \oplus \det F_1(z)I)$ defines a linear sequential transform of the output sequence a_2. This equation specifies a linear statistical distinguisher between the output sequence and a purely random sequence.

Namely, all the terms except the noise term on the right-hand side of (36) are polynomials in z and as such vanish in the time domain after a sufficiently large t depending on the degrees of the involved polynomials. So, (36) means that a linear sequential transform of the output sequence is termwise correlated to the all-zero 64-bit sequence where the approximation/correlation noise is defined by $(1 \oplus z^{48})E$. Equivalently, the 64 constituent binary subsequences, obtained as linear sequential transforms of the output sequence, are bitwise correlated to the all-zero binary sequence, where the corresponding correlation coefficients can be approximated as squares of the correlation coefficients of the corresponding binary noise subsequences of e. If c is such a correlation coefficient, then the output sequence length required for detecting the weakness in the corresponding binary subsequence is $O(c^{-4})$. The output sequence length required to detect the weakness by using all the 64 subsequences is then $O(c^{-4}/64)$.

The feasibility of the attack can be estimated by computing the noise correlation coefficients for different vectorial linear transformations L_1 and L_2. Assuming for simplicity that the underlying correlation coefficient $|c| = 2^{-3m}$ is the same for all the bits of x to be reconstructed, the attack would in theory be effective if the total required output sequence length, proportional to the complexity, is smaller than the expected period for the size of the internal state, that is, if $2^{12m}/64 < 2^{18\cdot 64}$, that is, if $m \leq 96$.

6 Conclusions

Our main finding is that the linearly updated component of MUGI, the so-called buffer, is not designed properly. It is proven that if the feedback from the nonlinearly updated component is disconnected, then the binary subsequences of the buffer, comprising its intrinsic response, are linear recurring sequences with the linear complexity of only 32 and with the period of only 48. Accordingly, the buffer neither provides a large lower bound on the period nor ensures good statistics of the output sequences of MUGI, unlike the usual designs of keystream generators. Furthermore, as each such subsequence depends on only 32 bits of the initial state of the buffer, the mixing between the 1024 bits of the initial state of the buffer is not good. In addition, it is pointed out that the 128-bit secret key can directly be recovered from the reconstructed internal state of MUGI at any time, which is not desirable.

As a consequence of this small period, it is shown that the buffer sequence can be eliminated from the update equations for the nonlinearly updated component of MUGI, the so-called state, thus yielding nonlinear recurrences involving only the output sequence and a part of the state sequence. It is then pointed out how the weakness of the buffer can be used to facilitate the linear cryptanalysis of MUGI. This is achieved by developing a framework for the linear cryptanalysis with two main objectives: to reconstruct the initial state or the secret key and to find linear statistical distinguishers for MUGI. This framework can be used for future experiments to investigate if the proposed attacks are feasible. More precisely, the feasibility depends on the magnitude of the corresponding correlation coefficients which can be estimated by examining the corresponding matrices with polynomial elements that result from the chosen linearizations of the S-boxes.

Appendix: Generating Function Representation

Firstly, by virtue of (5), the system of linear recurrences (4) results in

$$\sum_{t=0}^{\infty} b_4^{(t)} z^t = z^4 \sum_{t=5}^{\infty} b_4^{(t-4)} z^{t-4} \oplus z^{10} \sum_{t=5}^{\infty} b_{10}^{(t-10)} z^{t-10} \oplus z^5 \sum_{t=5}^{\infty} a_0^{(t-5)} z^{t-5} \tag{37}$$

$$\sum_{t=0}^{\infty} b_{10}^{(t)} z^t = z^6 \sum_{t=1}^{\infty} b_4^{(t-6)} z^{t-6} \oplus z^4 R_{32} \sum_{t=1}^{\infty} b_{10}^{(t-4)} z^{t-4}. \tag{38}$$

Then, we get

$$\begin{aligned}(1 \oplus z^4) B_4 \oplus z^{10} B_{10} &= z^5 A_0 \oplus \sum_{t=0}^{4} b_4^{(t)} z^t \oplus z^4 b_4^{(0)} \oplus \sum_{t=5}^{9} b_{10}^{(t-10)} z^t \\ &= z^5 A_0 \oplus (1 \oplus z^4) b_4^{(0)} \oplus \sum_{t=1}^{4} (b_4^{(t-4)} \oplus b_0^{(t-4)}) z^t \oplus \sum_{t=5}^{9} b_{10}^{(t-10)} z^t \\ &= z^5 A_0 \oplus b_4^{(0)} \oplus z^4 b_0^{(0)} \oplus \sum_{t=1}^{3} (b_4^{(t-4)} \oplus b_0^{(t-4)}) z^t \oplus \sum_{t=5}^{9} b_{10}^{(t-10)} z^t \end{aligned} \tag{39}$$

$$(I \oplus z^4 R_{32}) B_{10} \oplus z^6 B_4 = \sum_{t=1}^{5} b_4^{(t-6)} z^t \oplus b_{10}^{(0)} \oplus R_{32} \sum_{t=1}^{3} b_{10}^{(t-4)} z^t. \tag{40}$$

By using the simplified notation (8) and (9), we then directly obtain (6) and (7).

Acknowledgment

This work is based on a result of evaluation requested by the Japanese CRYPTREC project: http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html.

References

1. J. Daemen and C. Claap, "Fast hashing and stream encryption with PANAMA," Fast Software Encryption - FSE '98, *Lecture Notes in Computer Science*, vol. 1372, pp. 60-74, 1998.
2. J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Berlin: Springer-Verlag, 2002.
3. J. Dj. Golić, "Correlation via linear sequential circuit approximation of combiners with memory," Advances in Cryptology - EUROCRYPT '92, *Lecture Notes in Computer Science*, vol. 658, pp. 124-137, 1993.
4. J. Dj. Golić, "Linear cryptanalysis of stream ciphers," Fast Software Encryption - FSE '94, *Lecture Notes in Computer Science*, vol. 1008, pp. 154-169, 1995.
5. J. Dj. Golić, "Correlation properties of a general combiner with memory," *Journal of Cryptology*, vol. 9, pp. 111-126, 1996.
6. J. Dj. Golić, "Linear models for keystream generators," *IEEE Transactions on Computers*, vol. 45, pp. 41-49, 1996.
7. M. Matsui, "Linear cryptanalysis method for DES cipher," Advances in Cryptology - EUROCRYPT '93, *Lecture Notes in Computer Science*, vol. 765, pp. 159-169, 1994.
8. D. Watanabe, S. Furuya, H. Yoshida, and K. Takaragi, MUGI Pseudorandom number generator, Specification, Ver. 1.2, 2001, available at http://www.sdl.hitachi.co.jp/crypto/mugi/index-e.html.
9. D. Watanabe, S. Furuya, H. Yoshida, and K. Takaragi, MUGI Pseudorandom number generator, Self-evaluation report, Ver. 1.1, 2001, available at http://www.sdl.hitachi.co.jp/crypto/mugi/index-e.html.
10. D. Watanabe, S. Furuya, H. Yoshida, K. Takaragi, and B. Preneel, "A new keystream generator MUGI," Fast Software Encryption 2002, *Lecture Notes in Computer Science*, vol. 2365, pp. 179-194, 2002.