Security Evaluation of the K2 Stream Cipher

Editors: Andrey Bogdanov, Bart Preneel, and Vincent Rijmen

Contributors: Andrey Bodganov, Nicky Mouha, Gautham Sekar, Elmar Tischhauser, Deniz Toz, Kerem Varıcı, Vesselin Velichkov, and Meiqin Wang

Katholieke Universiteit Leuven
Department of Electrical Engineering ESAT/SCD-COSIC
Interdisciplinary Institute for BroadBand Technology (IBBT)
Kasteelpark Arenberg 10, bus 2446
B-3001 Leuven-Heverlee, Belgium

Version 1.1 — 7 March 2011

Contents

1 Executive Summary

K2 is a stream cipher proposed by S. Kiyomoto, T. Tanaka, and K. Sakurai at SECRYPT 2007. The design of K2 bears similarities to SNOW 2.0, though having some important distinctive features. The specification of K2 can be found in [19]. An overview of its architecture is given in Fig. 1. This document provides the results of a cryptographic evaluation of K2 performed by K.U.Leuven. The evaluation deals with attempting to attack K2 in several different ways corresponding to the state-of-the-art in the modern cryptanalysis of stream ciphers. This report includes analysis with respect to linear attacks, algebraic attacks, correlation and fast correlation attacks, differential attacks including the related-key setting, guess-and-determine attacks, statistical properties, period considerations as well as distinguishing attacks.

As regards linear attacks, we apply the linear masking method to a version of K2 ignoring the effect of dynamic feedback controller. The best correlation we find uses 13 linear approximations and is estimated to be 2^{-156}, which cannot be used to mount any successful attack. In the algebraic analysis, we study the structure and the quantitative properties of the resulting systems of equations and conclude that these make an algebraic attack infeasible. Our analysis towards a correlation attack or fast correlation attack (also not taking into account the dynamic feedback controller) indicates that this approach might be infeasible as well. Our differential analysis (it is assumed that there is no dynamic feedback controller and that the modular additions are replaced with XOR) includes a related-key attack, a related IV attack as well as a combination of both attacks and suggests that K2 can be resistant against the differential attacks. Our approaches to both byte- and word-oriented guess-and-determine attacks result in a complexity of 2^{320}, which makes these techniques inapplicable to K2. As to period considerations, no short periods have been found in K2. Our statistical tests do not reveal any structural flaws in the design of K2. Our analysis indicates that K2 also offers good resistance against mod n distinguishing attacks.

Thus, we have not been able to identify any weaknesses in K2 and conclude that that design of K2 is sound.

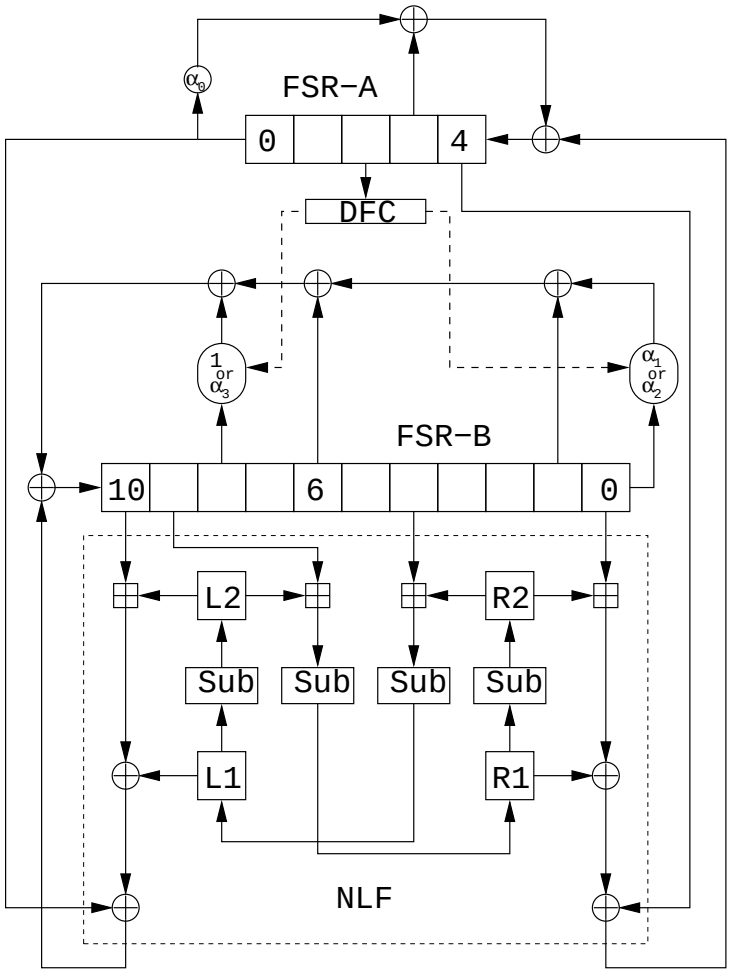

Figure 1: Initialization of KCipher-2 and its major components: FSR-A, FSR-B, DFC (Dynamic Feedback Controller), and NLF

2 Linear Attacks

2.1 Overview

The linear masking method attack was proposed by Coppersmith, Halevi and Jutla in [9]. It is applicable to stream ciphers, that are designed using a combination of a linear and a non-linear part. The output of the non-linear part is *masked* by the linear part. In this way the correlation between subsequent outputs of the non-linear part is removed. In this section we evaluate the applicability of a linear attack based on linear masking to K2.

The attack proceeds in two main steps. In one step, a linear approximation of the non-linear part is found. It relates bits of the output keystream for one clock to some bits of the internal state. The approximation has non-zero bias, hence the output from the non-linear part can be distinguished from random. In another step, a linear combination of the outputs of the linear part at several clocks is found with the property that the outputs cancel out. Finally, the approximation of the non-linear part is expressed for the same clocks for which the outputs of the linear part cancel. The result is a linear combination of some bits of the output keystream for several clocks, that has some bias. The exact value of the bias can be computed using Matsui's *Piling-up Lemma* [22]. If the computed bias is large enough, the output keystream can be distinguished from random.

The linear masking method has been successfully applied to mount a distinguishing attack on the stream cipher SNOW 2.0 [39]. This attack has been later improved in [27]. KCipher-2, having a design similar to SNOW 2.0, is a natural target for the linear masking attack. The non-linear part of K2 is its Non-Linear Function (NLF) [19]. The linear part is composed of the two feedback registers FSR-A and FSR-B. The part that differentiates KCipher-2 from other similar constructions (incl. SNOW 2.0) is the Dynamic Feedback Controller (DFC). The latter provides additional non-linearity in the linear part of the cipher. In order to analyze K2 against the linear-masking attack we ignore the DFC. If it can be shown that the attack cannot be applied to this simplified version of the cipher, then it will not be applicable also to the original version.

In the following analysis we proceed according to the general steps outlined above. First we find linear approximations for FSR-A and FSR-B in which their outputs cancel. Next we find a linear approximation of the NLF and we estimate its bias. Finally we evaluate the complexity of an attack that would use those relations.

2.2 Linear Relations for FSR-A and FSR-B

In the simplified version of K2, we ignore the dynamic feedback controller (DFC) by fixing $\mathrm{cl1} = 1, \mathrm{cl2} = 0$. In this case the following relations for FSR-A and FSR-B, respectively, hold for any linear mask Γ:

$$\Gamma(A_{t+5} \oplus \alpha_0 A_t \oplus A_{t+3}) = 0 \ , \tag{1}$$

$$\Gamma(B_{t+11} \oplus \alpha_1 B_t \oplus B_{t+1} \oplus B_{t+6} \oplus B_{t+8}) = 0 \ . \tag{2}$$

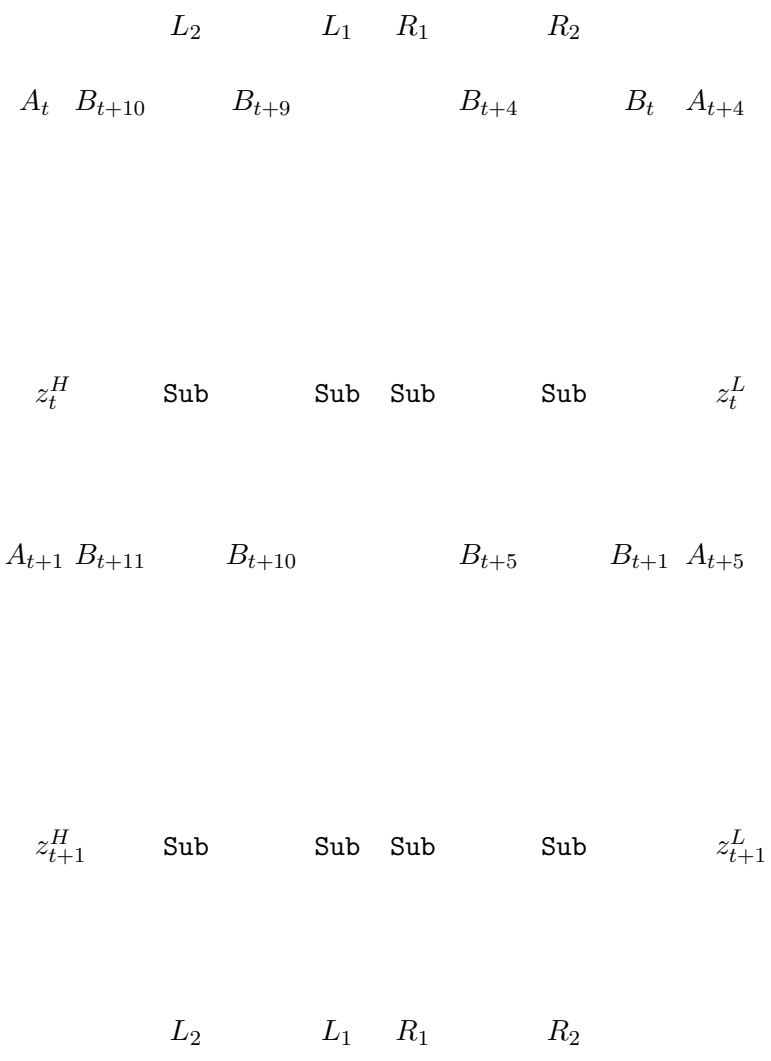

Figure 2: Linear path for two clocks of the NLF of K2.

Since multiplication in GF(2^{32}) can be expressed as a linear transformation in GF(2), equations (1)-(2) are equivalent to

$$(\Gamma\alpha_0)A_t \oplus \Gamma A_{t+3} \oplus \Gamma A_{t+5} = 0 \ , \tag{3}$$

$$(\Gamma\alpha_1)B_t \oplus \Gamma B_{t+1} \oplus \Gamma B_{t+6} \oplus \Gamma B_{t+8} \oplus \Gamma B_{t+11} = 0 \ . \tag{4}$$

2.3 Linear Approximation of the NLF

Consider the linear path for two clocks of the non-linear function (NLF) shown in Fig. 2. For a given linear mask Γ, the following relation holds:

$$\begin{aligned} &\Gamma A_t \oplus \Gamma A_{t+1} \oplus \Gamma A_{t+4} \oplus \Gamma A_{t+5} \oplus \\ &\Gamma B_t \oplus \Gamma B_{t+1} \oplus \Gamma B_{t+4} \oplus \Gamma B_{t+9} \oplus \Gamma B_{t+10} \oplus \Gamma B_{t+11} = \\ &\Gamma z_t^H \oplus \Gamma z_t^L \oplus \Gamma z_{t+1}^H \oplus \Gamma z_{t+1}^L \ . \end{aligned} \tag{5}$$

For masks Γ, $\Gamma\alpha_0$, $\Gamma\alpha_1$ and $\Gamma\alpha_0\alpha_1$, we use the linear relations (3),(4) and the linear approximation of the NLF (5) to obtain a system of linear equations. The smallest number of linear relations for which we find solution to the system is: 14 relations of type (3), 16 relations of type (4) and 13 relations of type (5). All equations are provided in Appendix A. As a result of solving the system of equations for linear masks Γ, $\Gamma\alpha_0$, $\Gamma\alpha_1$ and $\Gamma\alpha_0\alpha_1$ we obtain the following linear relation between the words of the output key-stream:

$$\begin{aligned} &(\Gamma\alpha_1\alpha_0)z_0^H \oplus (\Gamma\alpha_1\alpha_0)z_0^L \oplus (\Gamma\alpha_1\alpha_0)z_1^H \oplus (\Gamma\alpha_1\alpha_0)z_1^L \oplus \\ &(\Gamma\alpha_0)z_1^H \oplus (\Gamma\alpha_0)z_1^L \oplus (\Gamma\alpha_0)z_2^H \oplus (\Gamma\alpha_0)z_2^L \oplus \\ &(\Gamma\alpha_1)z_3^H \oplus (\Gamma\alpha_1)z_3^L \oplus (\Gamma\alpha_1)z_4^H \oplus (\Gamma\alpha_1)z_4^L \oplus \\ &\Gamma z_4^H \oplus \Gamma z_4^L \oplus \Gamma z_5^H \oplus \Gamma z_5^L \oplus \\ &(\Gamma\alpha_1)z_5^H \oplus (\Gamma\alpha_1)z_5^L \oplus (\Gamma\alpha_1)z_6^H \oplus (\Gamma\alpha_1)z_6^L \oplus \\ &\Gamma z_6^H \oplus \Gamma z_6^L \oplus \Gamma z_7^H \oplus \Gamma z_7^L \oplus \\ &(\Gamma\alpha_0)z_6^H \oplus (\Gamma\alpha_0)z_6^L \oplus (\Gamma\alpha_0)z_7^H \oplus (\Gamma\alpha_0)z_7^L \oplus \\ &(\Gamma\alpha_0)z_8^H \oplus (\Gamma\alpha_0)z_8^L \oplus (\Gamma\alpha_0)z_9^H \oplus (\Gamma\alpha_0)z_9^L \oplus \\ &\Gamma z_9^H \oplus \Gamma z_9^L \oplus \Gamma z_{10}^H \oplus \Gamma z_{10}^L \oplus \\ &(\Gamma\alpha_0)z_{11}^H \oplus (\Gamma\alpha_0)z_{11}^L \oplus (\Gamma\alpha_0)z_{12}^H \oplus (\Gamma\alpha_0)z_{12}^L \oplus \\ &\Gamma z_{13}^H \oplus \Gamma z_{13}^L \oplus \Gamma z_{14}^H \oplus \Gamma z_{14}^L \oplus \\ &\Gamma z_{14}^H \oplus \Gamma z_{14}^L \oplus \Gamma z_{15}^H \oplus \Gamma z_{15}^L \oplus \\ &\Gamma z_{16}^H \oplus \Gamma z_{16}^L \oplus \Gamma z_{17}^H \oplus \Gamma z_{17}^L = 0 \ . \end{aligned} \tag{6}$$

The above relation confirms the result reported by the designers in [19].

2.4 Complexity Estimation

Next we provide an estimation of the correlation with which (6) holds. In this analysis we replace all modular additions by `XOR`. The linear path which we use for the

derivation of (6) has four active Sub operations in one clock (see Fig. 2). Each of them uses four applications of the AES S-box. Let us assume that one AES S-box is active in every Sub operation. Since the linear approximation with maximum correlation of the AES S-box is 2^{-3}, we deduce that the correlation for one clock of the NLF is $(2^{-3})^4 = 2^{-12}$. As relation (6) uses 13 linear approximations, we estimate its correlation to be $c = (2^{-12})^{13} = 2^{-156}$. Its bias is $\epsilon = |\frac{c}{2}| = 2^{-157}$. Therefore $N \approx \epsilon^{-2} = 2^{314}$ words of the key stream are required in order to distinguish the output sequence of K2 from random.

In [19, Section 4.3, Distinguishing Attacks] the authors note that *"we have not found a combination of linear masks with a bias value higher than* 2^{-128}*"*. The estimated bias 2^{-157} of the reported linear relation (6) is lower than 2^{-128} and therefore confirms the estimation of the designers.

We have not tried to solve a system with more than 13 equations of type (6). This can be investigated in future work.

3 Algebraic Attacks

As Shannon postulated in 1949, breaking a good cipher should require "as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type". In fact, Shannon formulated the concept of the algebraic attack. In an algebraic attack, a cipher is expressed as a non-linear system of equations with a large number of unknown variables. Next a solution of the system is attempted. In general, solving multivariate non-linear equations is an NP-hard problem even for quadratic equations. However if the system of equations is overdefined and sparse, there are some methods such as XL [10], XSL [11] ,Gröbner bases [1] and SAT-Solvers. Both XL and XSL are based on the theory of linear algebra, Gröbner Bases is a math-heavy approach and SAT-Solvers and XL are CS-heavy approaches. However, Gröbner bases algorithms such as F4 or F5 are not always necessary. In many cases they can be replaced by a simpler attack that does not require a fixed monomial ordering and is essentially a linear algebra attack. In most cases, SAT-solvers are much faster and can break more instances than the current Gröbner bases techniques. SAT-solvers are based on heuristic search algorithm; then approach consists of guessing some variables and examining the consequences. If a contradiction has been found, a new restriction of equation can be added saying that in this set of constraints one is false. As open-source software, MiniSat2.0 has been a winner of SAT-Race 2006 competition for its amazing results in algebraic cryptanalysis of some simple ciphers.

Algebraic attacks have been applied to block ciphers such as AES and Serpent, with limited success. The algebraic cryptanalysis can attack only 6-round DES with one known plaintext and it has not given the expected results for the block cipher. However, the algebraic attacks have been applied successfully to break LFSR-based stream ciphers. Billet *et al.* [4] presented algebraic attacks for variants of the stream cipher SNOW-2. In the attack, Billet *et al.* compute the number of monomials in the variants of SNOW-2; it is so small that one can solve the system with linearization methods. However their attack is not applicable to the original cipher.

The main reason that Billet et al can attack variants of SNOW-2 is that the

registers for any clock t can be expressed linearly in terms of the initial state variables and the initial register value. In this way, the system of equations has fewer monomials, and it can be solved easily.

The stream cipher K2 is designed with the same non-linear components as SNOW-2. However it uses more registers than SNOW-2 and it also uses the dynamic feedback control to strengthen the non-linearity of the cipher. Therefore there are no such linear relations between some register and the initial state variables and the initial register value as in SNOW-2. In order to evaluate the resistance of K2 against an algebraic attack, we first analyze variants of K2. In these variants, we replace the addition operation with the `XOR` operation and we ignore the dynamic feedback controller. For these variants of K2, we try to derive a system of equations containing only key-stream words and internal-state words. Since there are four register words in the FSM of K2 cipher, we will use the derived system equations for multiple clock times to increase the equations-to-variables ratio.

For any cycle, we can express the register states words R_2^t , R_1^t and R_2^{t+1} as follows,

$$\begin{aligned}
&R_t^2 = A_{t+4} + B_t + R_t^1 + Z_t^L,\\
&R_t^1 = \mathrm{Sub}(A_{t-1} + B_{t+8} + B_{t+9} + \mathrm{Sub}(B_{t+2} + R_{t-2}^2) + Z_{t-1}^H),\\
&R_{t+1}^2 = \mathrm{Sub}(R_1^t).
\end{aligned}$$

In order to transform the above system of equations into a system of quadratic equations, we introduce the new variables N_t to replace the output of one component Sub.

$$\begin{aligned}
&N_t = \mathrm{Sub}(B_{t+2} + R_{t-2}^2),\\
&R_t^2 = A_{t+4} + B_t + R_t^1 + Z_t^L,\\
&R_t^1 = \mathrm{Sub}(A_{t-1} + B_{t+8} + B_{t+9} + N_t + Z_{t-1}^H),\\
&R_{t+1}^2 = \mathrm{Sub}(R_t^1).
\end{aligned}$$

Next we write the system equations from $2 \le t \le 8$ to compute the number R_t of equations, the number S_t of unknown variables and the number T_t of monomials.

For $t = 2$, we obtain

$$\begin{aligned}
&N_0 = \mathrm{Sub}(B_4 + R_0^2),\\
&R_2^2 = A_6 + B_2 + R_2^1 + Z_2^L,\\
&R_2^1 = \mathrm{Sub}(A_1 + B_{10} + B_{11} + N_0 + Z_1^H),\\
&R_3^2 = \mathrm{Sub}(R_2^1).
\end{aligned}$$

In the above system of equations, the number of equations $R_2 = 23 \cdot 4 \cdot 3 + 32 = 308$, the number of unknown variables $S_2 = 11 \cdot 32 = 352$, and the number of monomials $T_2 = ((81-8) \cdot 2 + 8) \cdot 4 + 4 \cdot 32 + (81-8) \cdot 4 \cdot 4 - 32 + (81-8) \cdot 4 = 2172$.

For $t = 3$,

$$\begin{aligned}
&N_1 = \mathrm{Sub}(B_5 + R_1^2),\\
&R_3^2 = A_7 + B_3 + R_3^1 + Z_3^L,\\
&R_3^1 = \mathrm{Sub}(A_2 + B_{11} + B_{12} + N_1 + Z_2^H),\\
&R_4^2 = \mathrm{Sub}(R_3^1).
\end{aligned}$$

In the above system of equations, the number of equations $R_3 = 23 \cdot 4 \cdot 3 + 32 = 308$, the number of unknown variables $S_3 = 9 \cdot 32 = 288$ and the number of monomials $T_3 = ((81-8) \cdot 2 + 8) \cdot 4 + 4 \cdot 32 + (81-8) \cdot 4 \cdot 4 - 32 + (81-8) \cdot 4 - 32 \cdot 2 = 2108$.

For $t = 4$,

$$\begin{aligned} N_2 &= \mathrm{Sub}(B_6 + R_2^2), \\ R_4^2 &= A_8 + B_4 + R_4^1 + Z_4^L, \\ R_4^1 &= \mathrm{Sub}(A_3 + B_{12} + B_{13} + N_2 + Z_3^H), \\ R_5^2 &= \mathrm{Sub}(R_4^1). \end{aligned}$$

In the above system of equations, the number of equations $R_4 = 23 \cdot 4 \cdot 3 + 32 = 308$, the number of unknown variables $S_4 = 6 \cdot 32 = 192$ and the number of monomials $T_4 = ((81-8) \cdot 2 + 8 - 8) \cdot 4 + 2 \cdot 32 + (81-8) \cdot 4 \cdot 4 - 32 \cdot 3 + (81-8) \cdot 4 = 2012$.

For $t = 5$,

$$\begin{aligned} N_3 &= \mathrm{Sub}(B_7 + R_3^2), \\ R_5^2 &= A_9 + B_5 + R_5^1 + Z_5^L, \\ R_5^1 &= \mathrm{Sub}(A_4 + B_{13} + B_{14} + N_3 + Z_4^H), \\ R_6^2 &= \mathrm{Sub}(R_5^1). \end{aligned}$$

In the above system of equations, the number of equations $R_5 = 23 \cdot 4 \cdot 3 + 32 = 308$, the number of unknown variables $S_5 = 5 \cdot 32 = 160$ and the number of monomials $T_5 = ((81-8) \cdot 2 + 8 - 8) \cdot 4 + 32 + (81-8) \cdot 4 \cdot 4 - 32 \cdot 3 + (81-8) \cdot 4 = 1980$.

For $t = 6$,

$$\begin{aligned} N_4 &= \mathrm{Sub}(B_8 + R_4^2), \\ R_6^2 &= A_{10} + B_6 + R_6^1 + Z_6^L, \\ R_6^1 &= \mathrm{Sub}(A_5 + B_{14} + B_{15} + N_4 + Z_5^H), \\ R_7^2 &= \mathrm{Sub}(R_6^1). \end{aligned}$$

In the above system of equations, the number of equations $R_6 = 23 \cdot 4 \cdot 3 + 32 = 308$, the number of unknown variables $S_6 = 4 \cdot 32 = 128$ and the number of monomials $T_6 = ((81-8) \cdot 2 + 8 - 8) \cdot 4 + 32 + (81-8) \cdot 4 \cdot 4 - 32 \cdot 4 + (81-8) \cdot 4 = 1948$.

For $t = 7$,

$$\begin{aligned} N_5 &= \mathrm{Sub}(B_9 + R_5^2), \\ R_7^2 &= A_{11} + B_7 + R_7^1 + Z_7^L, \\ R_7^1 &= \mathrm{Sub}(A_6 + B_{15} + B_{16} + N_5 + Z_6^H), \\ R_8^2 &= \mathrm{Sub}(R_7^1). \end{aligned}$$

In the above system of equations, the number of equations $R_7 = 23 \cdot 4 \cdot 3 + 32 = 308$, the number of unknown variables $S_7 = 3 \cdot 32 = 96$ and the number of monomials $T_7 = ((81-8-8) \cdot 2 + 8) \cdot 4 + 32 + (81-8-8) \cdot 4 \cdot 4 + (81-8) \cdot 4 = 1916$.

For $t = 8$,

$$\begin{aligned} N_6 &= \mathrm{Sub}(B_{10} + R_6^2), \\ R_8^2 &= A_{12} + B_8 + R_8^1 + Z_8^L \\ R_8^1 &= \mathrm{Sub}(A_7 + B_{16} + B_{17} + N_6 + Z_7^H), \\ R_9^2 &= \mathrm{Sub}(R_8^1). \end{aligned}$$

Table 1: Number of Equations, Unknown Variables and Terms

t	R_t	S_t	T_t	$Total_R$	$Total_S$	$Total_T$
2	308	352	2172	308	352	2172
3	308	288	2108	716	640	4280
4	308	192	2012	924	832	6292
5	308	160	1980	1232	992	8272
6	308	128	1948	1540	1120	10220
7	308	96	1916	1848	1216	12126
8	308	96	1916	2156	1312	14052
...	...	...	...	...	...	...
n	308	96	1916	$308 \cdot (n-1)$	$(n-6) \cdot 96 + 1120$	$(n-6) \cdot 1916 + 10220$

In the above system of equations, the number of equations $R_8 = 23 \cdot 4 \cdot 3 + 32 = 308$, the number of unknown variables $S_8 = 3 \cdot 32 = 96$ and the number of monomials $T_8 = ((81-8-8) \cdot 2 + 8) \cdot 4 + 32 + (81-8-8) \cdot 4 \cdot 4 + (81-8) \cdot 4 = 1916$. Finally, $R_t = 308$, $S_t = 96$, $T_t = 1916$ with $t \geq 8$.

The above results are summarized in Table 1, where Total_R, Total_S and Total_T are the total number of equations, variables and monomials, respectively, for clock t. As t increases, the ratio $\text{Total}_R/\text{Total}_T$ approaches 0.160752.

As shown in Table 1, there are $(n-6) \cdot 96 + 1120$ variables and $308 \cdot (n-1)$ equations. For a larger value of n, the system will be more overdefined, but the equations-to-variables ratio is bounded above by $2^{1.67}$. The equations-to-monomials ratio is bounded above by $2^{-2.64}$.

For AES-128, the number of equations is 8000, the number of variables is 1600 and the number of monomials is about 89600, so the equations-to-variables ratio is $2^{2.32}$ and the equations-to-monomials ratio is about $2^{-3.48}$.

For Serpent-128, the number of equations is 43680, the number of variables is 8192 and the number of monomials is 270336, so the equations-to-variables ratio is $2^{2.41}$ and the equations-to-monomials ratio is about $2^{-2.63}$.

The equations for the original SNOW-2 are as follows:

$$
\begin{aligned}
&r_1^t = c_2^{t-1} \oplus r_2^{t-1} \oplus s^{t+4}, \\
&r_2^t = r_2^{t-1} \oplus z^t \oplus s^{t+15} \oplus s^{t+14} \oplus s^t \oplus c_1^t \oplus c_2^{t-1} \\
&c_{1,[0]}^t = 0 \\
&c_{1,[1]}^t = s_{[0]}^{t+15} r_{1,[0]}^t \\
&c_{1,[i+1]}^t = s_{[i]}^{t+15} r_{1,[i]}^t \oplus s_{[i]}^{t+15} c_{1,[i]}^t \oplus c_{1,[i]}^t r_{1,[i]}^t, 0 < i < 32; \\
&c_{2,[0]}^t = 0 \\
&c_{2,[1]}^t = s_{[0]}^{t+5} r_{2,[0]}^t \\
&c_{2,[i+1]}^t = s_{[i]}^{t+5} r_{2,[i]}^t \oplus s_{[i]}^{t+5} c_{2,[i]}^t \oplus c_{2,[i]}^t r_{2,[i]}^t, 0 < i < 32; \\
&r_2^{t+1} = S(r_1^t)
\end{aligned}
$$

The total number of equations is $(32 + 32 + 32 + 32 + 39 \cdot 4)n = 282n$, the number of variables is $544 + (64 + 62)n = 544 + 126n$ and the number of monomials is

Table 2: R/T and R/S

Cipher	R/T	R/S
AES-128	$2^{-3.48}$	$2^{2.32}$
Serpent-128	$2^{-2.63}$	$2^{2.41}$
Variants K2	$2^{-2.64}$	$2^{1.67}$
SNOW-2	$2^{-1.49}$	$2^{1.16}$

$(137 \cdot 4 + (3 \cdot 30 + 1) \cdot 2 + 62)n + 544 = 792n + 544$. Therefore the equations-to-monomials ratio is about $2^{-1.49}$ and the equations-to-variables ratio is about $2^{1.16}$. These results are summarized in Table 2.

From Table 2, we can see that the sparsity for variants of K2 cipher is weaker than that of SNOW-2 and Serpent-128, but stronger than that of AES-128 and Serpent-128. Since an algebraic attack on SNOW-2 is difficult, the attack for variants of K2 will be even more difficult.

If we consider the original K2 cipher then the `XOR` operation is replaced by addition operation and the dynamic feedback controller is considered. This makes an algebraic attack on K2 more difficult than on variants of K2. We conclude that an algebraic attack for K2 is infeasible.

4 Correlation Attacks

4.1 Introduction

The *correlation attack* is an attack on stream ciphers, proposed by Siegenthaler in 1985 [34, 35]. The attack can be either a known-plaintext attack or a ciphertext-only attack. Redundancy in the plaintext is required for the ciphertext-only attack. The attack is applicable to keystream generators composed of several LFSRs (linear feedback shift registers). We denote the number of LFSRs as n, and their respective lengths as ℓ_i, $0 < i \leq n$. A divide-and-conquer approach is used: the correlation attack aims to recover the initial state of each LFSR separately.

4.2 Combination Generators and Linear Complexity

Several variants of the correlation attack exist. The original attack of [35] applies to combination generators, but can be extended to other keystream generators as well.

In a combination generator (see Fig. 3), the output bits of the n LFSRs are combined by using a Boolean function $f(x_1, \ldots, x_n)$. The output of a combination generator is a linear recurring sequence s_t. The length of the shortest LFSR that can generate this infinite recurring series s_t, is denoted by $\Lambda(S)$. Using $2\Lambda(S)$ output bits, the Berlekamp-Massey algorithm [21] can be used to construct this shortest LFSR.

If all n LFSR lengths ℓ_i, $0 < i \leq n$ are different and greater than two, the outputs of the combination generator has a linear complexity of

$$f^*(x_1, x_1, \ldots, x_n) \ , \tag{7}$$

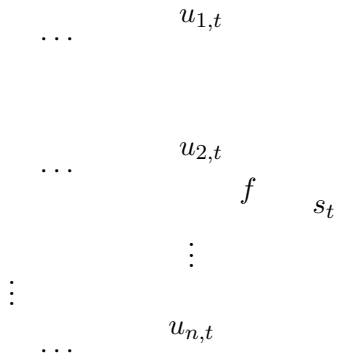

Figure 3: Schematic representation of a combination generator.

provided that the n LFSRs are all started in a non-zero state [31]. Here, f^* is the algebraic normal form of f, evaluated over the integers. For example, if three LFSRs of lengths ℓ_1, ℓ_2 and ℓ_3 are combined by the Boolean function

$$x_1x_2 \oplus x_2x_3 \oplus x_3 \ , \tag{8}$$

then the linear complexity of the output is

$$\ell_1\ell_2 + \ell_2\ell_3 + \ell_3 \ . \tag{9}$$

A high linear complexity ensures that the Berlekamp-Massey algorithm algorithm becomes computationally infeasible. This implies that to preclude attacks, the Boolean function f must be non-linear.

4.3 Description of the Correlation Attack

Assume that the output $f(x_1, \ldots, x_n)$ of the Boolean function is correlated to one of its inputs x_i. In other words, the probability

$$p = \Pr[f(x_1, \ldots, x_n) \neq x_i] \tag{10}$$

is not $1/2$. Denote the keystream by s_t, $0 \leq t < N$, and the output of the i-th LFSR as s_t, $0 \leq t < N$.

Then,

$$X = \sum_{t=0}^{N-1} s_t \oplus u_t \tag{11}$$

Table 3: Pseudocode for the correlation attack.

Input:
N keystream bits $s_0, s_1, \ldots, s_{N-1}$.
Output:
The internal state of the i-th LFSR: $u_0, u_1, \ldots, u_{\ell_i-1}$.
Required:
$p = \Pr[f(x_1, \ldots, x_n) \neq x_i]$
For every value of the internal state $u_0, u_1, \ldots, u_{\ell_i-1}$:
Calculate $u_0, u_1, \ldots, u_{N-1}$.
Compute the correlation between $u_0, u_1, \ldots, u_{N-1}$ and $s_0, s_1, \ldots, s_{N-1}$:
$X = \sum_{t=0}^{N-1} s_t \oplus u_t$.
If X is sufficiently close to Np, **then**
output the initial state $u_0, u_1, \ldots, u_{N-1}$.

will be a binomially distributed random variable, with an average of $\bar{X} = Np$ and variance of $\sigma = Np(1-p)$.

To perform the correlation attack, we initialize the i-th LFSR to each of the possible states. We then generate the first N bits of the sequence u_t produced by the i-th LFSR, and calculate the correlation X between u_t and the keystream output s_t. If X is sufficiently close to Np, the value to which the i-th LFSR was initialized will be correct with a high probability. This correlation attack is explained in pseudocode in Table 3. To avoid correlation attacks, the concept of correlation immunity was introduced by Siegenthaler in [34], and further investigated in [6, 7, 33, 37, 40, 41].

For example, assume 3 LFSRs are used, and the output of $f(x_1, x_2, x_3)$ is correlated to both x_1 and x_2. Then, the correlation attack can be used to recover the initial state of the first and second LFSR, independently of each other and of the third LFSR. The initial state of the third LFSR can then be recovered by exhaustive search. The initial state of all LFSRs can then be recovered with a complexity of at most

$$\sum_{i=1}^{3} 2^{\ell_i} - 1 \ , \tag{12}$$

instead of

$$\prod_{i=1}^{3} 2^{\ell_i} - 1 \tag{13}$$

if exhaustive search is used for all LFSRs. This analysis assumes that each of the LFSRs can be initialized to every possible initial state, except the all-zero state.

4.4 Application of the Correlation Attack to KCipher-2

The KCipher-2 stream cipher consists of two LFSRs, denoted as FSR-A and FSR-B, that are combined using a non-linear function. Additionally, a dynamic feedback controller is used to provide irregular clocking of FSR-B. To simplify our analysis, we will ignore the irregular clocking.

If we apply a correlation attack to KCipher-2, the goal is to recover the internal state of each LFSR separately. The following observations make it clear why the original correlation attack is not feasible.

The first four words of the internal state of FSR-A are initialized with the 128-bit key k. The fifth word of FSR-A ensures the LFSR is never initialized to an all-zero value. Any attack to exhaustively search over all possible values of FSR-A, corresponds to searching over all values of the 128-bit key k. Therefore, even if the keystream is correlated in some way to the output of FSR-A, such an attack would require searching over 2^{k-1} keys on average. Therefore, this attack corresponds to searching over the entire 128-bit key space.

Another line of attack, might be to recover FSR-B separately of FSR-A. Again, the key schedule ensures that FSR-B never contains the all-zero state. The key schedule can be rewritten as follows:

$$K_i = \begin{cases} K_{i+4} \oplus S[K_{i+3} \lll 8] \oplus \mathrm{Rcon}[i/4] & \text{for } i = 4n, \\ K_{i+4} \oplus K_{i+3} & \text{for } i \neq 4n. \end{cases} \tag{14}$$

Therefore, the value of four consecutive expanded keys $\{K_i, K_{i+1}, K_{i+2}, K_{i+3}\}$ uniquely determines the value of the key $k = \{K_0, K_1, K_2, K_3\}$. As a result, obtaining the initial state of FSR-B is equivalent to obtaining the key k. The divide-and-conquer approach of the original correlation attack will not give any advantage over exhaustive search: even if the keystream and the output of FSR-B are correlated in some way, it will be as difficult to obtain the initial state of FSR-B as it is to obtain the key k (both imply knowledge of the key k).

Therefore, we conclude that the original correlation attack cannot be applied to KCipher-2, because once the initial state of either FSR-A or FSR-B is recovered, the key k and therefore the contents of the other FSR are determined as well. Our simplified analysis assumes that there is no dynamic feedback controller, and that a Boolean function is used to combine both LFSRs. In KCipher-2, the combination of both LFSRs is done using a complicated function, containing a mix of

- the exclusive-or operation,
- addition modulo 2^{32},
- a non-linear permutation, namely the AES 8×8-bit S-box, applied four times in parallel on each byte of a 32-bit word,
- a linear permutation, implemented as the AES 32×32-bit *MixColumn* operation,
- four state registers ($R1$, $R2$, $L1$ and $L2$).

As we explained that the complexity of correlation attacks increases with the non-linearity of the combining function, this observation further complicates the application of the correlation attack.

4.5 Fast Correlation Attacks

For this report, *fast correlation attacks* were investigated as well. Fast correlation attacks were proposed by Meier and Staffelbach at EUROCRYPT 1998 [23, 24].

In the correlation attack of Table 3, the initial state of one LFSR is recovered by exhaustive search. In fast correlation attacks, the combining function is modeled by an LFSR followed by a binary symmetric channel (BSC) with crossover probability p^*. If there is a correlation between the output of the LFSR and the keystream generator, the crossover probability p^* of the BSC will be lower than 1/2.

In this case, the fast correlation attack will use techniques based on coding theory (e.g. maximum likelihood decoding) to recover the initial state of the LFSR faster than exhaustive search. In [15], Johansson and Jönsson introduce correlation attacks using convolutional codes, and in [14] using turbo codes. The memory requirements of these attacks are further reduced in [8] and [16].

Similar to correlation attacks, the complexity of fast correlation attacks increases with the non-linearity of the combining function f. Therefore, the application of the fast correlation attack will have the same complications as the correlation attack described earlier.

5 Differential Attacks

Differential cryptanalysis, introduced by Biham and Shamir, is one of the most powerful attacks against iterated ciphers. It is usually a chosen plaintext attack in which the attacker observes how the differences in an input can affect the resultant difference at the output. This difference can be defined in several ways, but the XOR operation is usual. A pair of constant input and output differences is called a *differential*. The probabilities of such differentials can be used to determine a lower bound on the complexity of an differential attack to show when an cipher is vulnerable to these attacks.

In this section, we concentrate on the propagation of (XOR) differences throughout the cipher in each clocking. We consider three attack scenarios: related key attack, related IV attack and a combination of these two attacks.

In order to simplify the analysis, we use a modified (probably weaker) version of the algorithm. The modular additions are replaced with the XOR operation, the dynamic feedback controller is removed (i.e., multiplication with constants α_1, α_2 and α_3 are omitted) and multiplication with α_0 is ignored. The new version can be seen in Fig. 4.

5.1 Properties of Components

In this section, we investigate the differential properties of the building blocks in the `Sub` transformation.

5.1.1 Substitution

The `Sub` transformation uses an 8-bit to 8-bit bijective S-box four times in parallel to process each word. Since KCipher-2 uses a bijective S-box, thus, $S(\Delta x) = 0$ *if and only if* $\Delta x = 0$. The difference distribution table (DDT) of the S-box contains exactly 127 nonzero output differences for a given nonzero input difference. Only one of these values has probability of 2^{-6} while the other 126 remaining nonzero values have probability of 2^{-7}.

Table 4: Probability Distribution of the Linear Layer

m\ n	1	2	3	4
1	0	0	0	1.000
2	0	0	$2^{-7.994}$	0.984
3	0	$2^{-15.988}$	$2^{-8.017}$	0.985
4	$2^{-23.983}$	$2^{-16.011}$	$2^{-8.016}$	0.985

5.1.2 Linear Permutation

The 32-to-32 linear permutation used in KCipher-2 is based on an MDS matrix with a branch number of 5. The detailed input-output distribution of differences are given in Table 5.

Based on this distribution we can compute the probability of an activity pattern with m active S-boxes to go to a fixed activity pattern with n active S-boxes after the linear permutation. This probability is denoted by $Pr[m \rightarrow n]$ and it is given in Table 4 for all possible patterns.

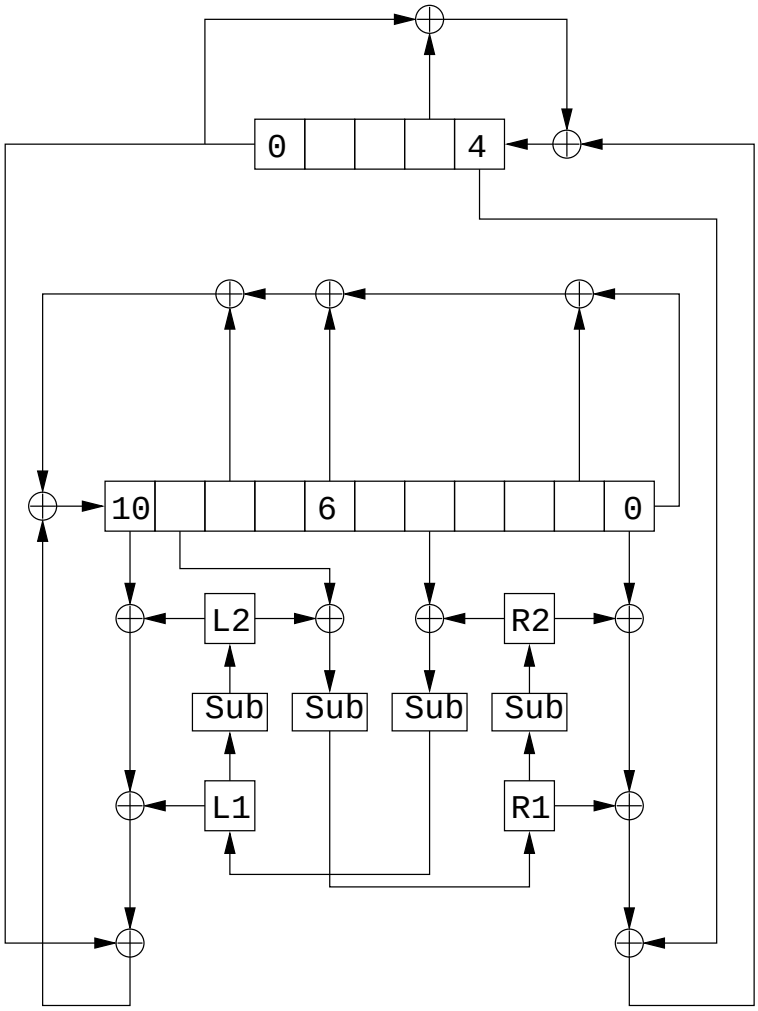

Figure 4: Initialization for Modified KCipher-2: modular additions are replaced with XORs, the DFC is ignored

Table 5: Input (I) - Output (O) Distribution of the Linear Layer

I \ O	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	255
0010	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	255
0011	0	0	0	0	0	0	0	255	0	0	0	255	0	255	255	64005
0100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	255
0101	0	0	0	0	0	0	0	255	0	0	0	255	0	255	255	64005
0110	0	0	0	0	0	0	0	255	0	0	0	255	0	255	255	64005
0111	0	0	0	255	0	255	255	64005	0	255	255	64005	255	64005	64005	16323825
1000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	255
1001	0	0	0	0	0	0	0	255	0	0	0	255	0	255	255	64005
1010	0	0	0	0	0	0	0	255	0	0	0	255	0	255	255	64005
1011	0	0	0	255	0	255	255	64005	0	255	255	64005	255	64005	64005	16323825
1100	0	0	0	0	0	0	0	255	0	0	0	255	0	255	255	64005
1101	0	0	0	255	0	255	255	64005	0	255	255	64005	255	64005	64005	16323825
1110	0	0	0	255	0	255	255	64005	0	255	255	64005	255	64005	64005	16323825
1111	0	255	255	64005	255	64005	64005	16323825	255	64005	64005	16323825	64005	16323825	16323825	4162570275

5.2 Key Ideas of the Attacks

We first start with a primitive algorithm to get the initial results on KCipher-2. In this scheme we concentrate only on 32-bit words, where each word is denoted by one if there is an XOR difference in it and zero otherwise. Then, the number of active `Sub` functions in the whole state is counted for each clocking. This way we can observe the best cases (the states with minimum number of active words) without getting into the details of the algorithm.

In order to count the minimum number of active S-boxes, we treat each word as a combination of four bytes and denote it by a number between zero and four. Again, zero corresponds to a passive word and a positive number indicates the number of active bytes within a word. When two active words enter the XOR operation, we take the minimum possible number as the output difference. (i.e. let n_1 and n_2 be the values of the first and second words respectively, then the resulting output will be $|n_1 - n_2|$). So, the XOR operation is replaced by `ABS` operation to obtain an active word with the minimum number of active bytes.

The linear layer in the `Sub` operation is also analyzed in two different ways. In the first approach, if the input value of the linear layer (denoted by x) is greater than zero, then the output value is always taken as four since it occurs with a high probability (see the probability distribution of the linear layer in Table 4 for details). In the second approach, the output value is taken as $5 - x$ by the property of the linear layer. Since the branch number of the corresponding MDS code is 5, this value will be the minimum that one can get after the linear layer. If the input value is zero, then output value is zero for both cases.

For the detailed analysis, we combine the ideas presented above. We still treat each active word as a combination of four bytes denoted by a number between one and four. But this time, we take all possible output values of each XOR operation and the `Sub` functions individually into consideration. So, the states in clocking are represented as a tree structure where each state corresponds to a branch. Then we perform a depth first search in which the states whose total active S-box number exceeding a predefined value for each round are eliminated immediately.

To be more specific and clear: in each clocking of modified KCipher-2 six values $(A_4, B_{10}, R_1, R_2, L_1, L_2)$ are updated and there exists eighteen variables used in fourteen XOR operations. Each word can take five values, so for the the worst case $5^{18} \approx 2^{41.8}$ possible branches can occur after one clocking, whereas on average this value is $3.7^{18} \approx 2^{34.0}$. This means that complexity is too high to analyze multiple rounds. Therefore, the XOR operations used to update B_{10} and A_4 are divided into two groups (see (15) and (16)), with either four variables or two variables. A similar idea is also applied to the update of L_1 and R_1 (see (17) and (18)).

$$B_{10} = (A_0 \oplus B_{10} \oplus L_1 \oplus L_2) \oplus (B_0 \oplus B_1 \oplus B_6 \oplus B_8) \quad (15)$$

$$A_4 = (A_4 \oplus B_0 \oplus R_1 \oplus R_2) \oplus (A_0 \oplus A_3) \quad (16)$$

$$L_1 = Sub(B_4 \oplus R_2) \quad (17)$$

$$R_1 = Sub(B_9 \oplus L_2) \quad (18)$$

The output values of the linear layer are restricted to either zero, four or both,

depending on the input values. If both words entering the XOR operation before `Sub` are passive, then the output value is also passive and it is set to zero. If one of the inputs is active and the other is not, then the output value will be active and the number of active bytes at the output will be four, since it is the most probable case as indicated in Table 5. Finally, if both of the inputs entering the XOR operation is active, then the output value can be four (when a difference remains) or zero (if the input differences cancel each other).

All these changes reduce the worst case complexity to $5^4 \times 3^2 \approx 2^{12.5}$ for one clocking. As a result, the computation of more rounds becomes feasible.

5.3 Related-Key Attacks

Related-key attacks are a special type of differential attacks that allow the keys to be chosen with specific differences. In this section, we analyze the resistance of KCipher-2 against the related-key attacks. Our aim is to obtain the minimum number of active S-boxes throughout the cipher by using differential cryptanalysis.

We start with the *key loading step* to obtain the number of active words and to find some useful properties that can be exploited in the related-key attacks. To do so, we assume there exists an XOR difference in the initial key $K = (K_0, K_1, K_2, K_3)$.

First of all, the key schedule algorithm is rewritten where $\Delta K_3^{'}$ and $\Delta K_7^{'}$ are the values obtained from K_3 and K_7 (i.e., $K_i^{'} = \texttt{Sub}(K_i \lll 8)$). Since all the subkey values are generated from the initial key, 15 possible cases could be obtained from key initialization by introducing differences.

$$
\begin{aligned}
\Delta K_0 &= \Delta K_0 \\
\Delta K_1 &= \Delta K_1 \\
\Delta K_2 &= \Delta K_2 \\
\Delta K_3 &= \Delta K_3 \\
\Delta K_4 &= \Delta K_0 \oplus \Delta K_3^{'} \\
\Delta K_5 &= \Delta K_0 \oplus \Delta K_1 \oplus \Delta K_3^{'} \\
\Delta K_6 &= \Delta K_0 \oplus \Delta K_1 \oplus \Delta K_2 \oplus \Delta K_3^{'} \\
\Delta K_7 &= \Delta K_0 \oplus \Delta K_1 \oplus \Delta K_2 \oplus \Delta K_3 \oplus \Delta K_3^{'} \\
\Delta K_8 &= \Delta K_0 \oplus \Delta K_3^{'} \oplus \Delta K_7^{'} \\
\Delta K_9 &= \Delta K_1 \oplus \Delta K_7^{'} \\
\Delta K_{10} &= \Delta K_0 \oplus \Delta K_2 \oplus \Delta K_3^{'} \oplus \Delta K_7^{'} \\
\Delta K_{11} &= \Delta K_1 \oplus \Delta K_3 \oplus \Delta K_7^{'}
\end{aligned}
$$

Next, some conditions are introduced to get the minimum number of active subkeys. We try to avoid a difference from $\Delta K_3^{'}$ and $\Delta K_7^{'}$ and finally count the number of active subkeys. Table 6 represents the best results and as it can be seen, the minimum number of active subkeys is five after the key scheduling algorithm.

The best results in terms of minimum number of active `Sub` functions for this attack are given in Table 7. By using the methods given in Section 5.2, we can obtain

Table 6: Propagation of key differences

	Conditions:			
Subkey Differences:	$\Delta K_0 = \Delta K_2$	$\Delta K_1 = \Delta K_2$	$\Delta K_1 = \Delta Sub(\Delta K_1 \lll 8)$	$\Delta K_2 = \Delta Sub(\Delta K_2 \lll 8)$
ΔK_0	ΔK	-	-	-
ΔK_1	-	ΔK	ΔK	-
ΔK_2	ΔK	ΔK	-	ΔK
ΔK_3	-	-	-	-
ΔK_4	ΔK	-	-	-
ΔK_5	ΔK	ΔK	ΔK	-
ΔK_6	-	-	ΔK	ΔK
ΔK_7	-	-	ΔK	ΔK
ΔK_8	ΔK	-	ΔK	ΔK
ΔK_9	-	ΔK	-	ΔK
ΔK_{10}	-	ΔK	ΔK	ΔK
ΔK_{11}	-	ΔK	-	-
	5	6	6	6
	Number of Active Subkeys			

the minimum number of active S-boxes for KCipher-2. For the best case, in terms of maximum number of cycles by defining a threshold on the number of active S-boxes, there exits 33 active S-boxes through the 8 cycles of key initialization, so that the probability of any differential characteristic is upper bounded by a probability of 2^{-198} which means that the related-key scenario is not feasible for 24 rounds or more (see Table 8).

5.4 Related-IV Attacks

In this section, we analyze the resistance of KCipher-2 against related-IV attacks. These attacks are similar to related-key attacks and are based on the same methodology. However, the attacker has advantage in these attacks since the IV words are independent of each other unlike the key words that are related to each other by a *key loading step*. Therefore one has more control over the choice of the IV words and can start with a lower number of active words.

As a first step, we initialized the state with all possible $2^4 - 1 = 15$ values of $IV = (IV_0, IV_1, IV_2, IV_3)$ and clocked the internal state for 24 rounds in the *initialization process*. Then, we checked if all zero state can be observed in any clocking step. We observed that the propagation of the differences is quite fast and, hence, it is impossible to achieve all zero state in KCipher-2.

The best results in terms of minimum number of active `Sub` functions for this attack are given in Table 9. The best results in terms of maximum number of cycles by defining a threshold on the number of active S-boxes are given in Table 10. For the best case there exist 30 active S-boxes through the 9 cycles of key initialization, so that any differential characteristic has a probability of at most 2^{-180}.

5.5 Related Key/IV Attacks

In this type of attack we assume differences both in the IV and the initial key K. The best results in terms of minimum number of active `Sub` functions for this attack are given in Table 11. For the cases with few active words for the related-IV attack, we searched for possible IV activity patterns. Based on our experiments, the minimum number of active S-boxes is 32 after 8 rounds. The propagation of differences in the best case is given in Table 12.

5.6 Conclusion and Remarks

We can conclude that KCipher-2 is resistant against differential attacks. The high number of active S-boxes in the initialization process makes it difficult for an attacker to control the differences at the keystream.

Some other observations are as follows:

- If one can obtain a zero difference in FSR-A after the initialization step, since there is no feedback in the keystream output process FSR-A becomes redundant. Hence it is possible to remove it from the system and analyze only FSR-B.

- If there exist differences only in A_0 and B_0 at a given time then it is possible to cancel them and have an all zero difference. But some conditions must be satisfied like: $\Delta A_0.\alpha_0 = \Delta B_0$ and $\Delta B_0.\alpha_i = \Delta A_0$ (i.e., $\alpha_0.\alpha_i = 1$) $i \in \{0, 1\}$.

Table 7: Results for the related key attack

A_0	A_1	A_2	A_3	A_4	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	L_2	L_1	R_2	R_1	#Active `Sub`
1	0	0	1	1	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	0	1	1	0	0	1	0	1	0	0	2
0	1	1	0	0	1	1	0	0	0	0	0	1	1	0	0	1	1	0	0	1
1	1	0	0	0	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0
1	0	0	0	1	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0	1
0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0	1	2
0	0	1	1	0	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	3
0	1	1	0	1	0	0	1	1	1	1	1	0	0	0	0	0	1	1	1	2
1	1	0	1	1	1	0	0	1	1	1	1	1	0	0	0	1	0	1	0	1
1	0	1	1	0	0	1	0	0	1	1	1	1	1	0	0	0	0	0	1	3
0	1	1	0	1	0	0	1	0	0	1	1	1	1	1	0	0	1	1	1	2
1	1	0	1	1	1	0	0	1	0	0	1	1	1	1	1	1	0	1	0	1
1	0	1	1	1	1	1	0	0	1	0	0	1	1	1	1	0	0	0	1	2
0	1	1	1	1	1	1	1	0	0	1	0	0	1	1	1	0	0	1	1	3
1	1	1	1	1	0	1	1	1	0	0	1	0	0	1	1	0	1	1	1	3
1	1	1	1	0	1	0	1	1	1	0	0	1	0	0	1	1	0	1	1	3
1	1	1	0	1	0	1	0	1	1	1	0	0	1	0	0	0	1	1	1	4
1	1	0	1	0	1	0	1	0	1	1	1	0	0	1	0	1	1	1	1	3
1	0	1	0	0	1	1	0	1	0	1	1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	0	1	1	0	1	0	1	1	1	0	0	0	0	1	0	1
1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	0	0	0	0	1	1
0	0	0	1	1	1	0	0	1	1	0	1	0	1	1	1	0	0	1	0	0
0	0	1	1	0	0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	1
0	1	1	0	0	1	0	1	0	0	1	1	0	1	0	1	0	0	0	1	2
1	1	0	0	0	1	1	0	1	0	0	1	1	0	1	0	0	1	1	0	
Total																				43

Table 8: Results for the related key attack

A_0	A_1	A_2	A_3	A_4	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	L_2	L_1	R_2	R_1	#Active S-boxes
4	0	0	4	4	0	0	0	0	0	4	4	0	0	4	4	0	0	0	0	4
0	0	4	4	0	4	0	0	0	0	0	4	4	0	0	4	0	4	0	0	8
0	4	4	0	0	4	4	0	0	0	0	0	4	4	0	0	4	4	0	0	4
4	4	0	0	0	0	4	4	0	0	0	0	0	4	4	0	4	0	0	0	0
4	0	0	0	4	0	0	4	4	0	0	0	0	0	4	4	0	0	0	0	0
0	0	0	4	4	1	0	0	4	4	0	0	0	0	0	4	0	0	0	0	0
0	0	4	4	4	0	1	0	0	4	4	0	0	0	0	0	0	0	0	0	1
0	4	4	4	0	4	0	1	0	0	4	4	0	0	0	0	0	0	0	4	8
4	4	4	0	0	3	4	0	1	0	0	4	4	0	0	0	0	4	4	0	8
Total																				33

Table 9: Results for the related IV attack

A_0	A_1	A_2	A_3	A_4	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	L_2	L_1	R_2	R_1	#Active `Sub`
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1
0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1
0	1	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	1	0	0	2
1	1	1	1	0	1	1	1	0	0	1	0	0	0	0	0	1	0	0	1	1
1	1	1	0	1	0	1	1	1	0	0	1	0	0	0	0	0	0	1	0	1
1	1	0	1	1	0	0	1	1	1	0	0	1	0	0	0	0	0	0	1	1
1	0	1	1	0	1	0	0	1	1	1	0	0	1	0	0	0	0	1	0	1
0	1	1	0	1	1	1	0	0	1	1	1	0	0	1	0	0	1	0	0	3
1	1	0	1	1	0	1	1	0	0	1	1	1	0	0	1	1	1	0	1	3
1	0	1	1	1	1	0	1	1	0	0	1	1	1	0	0	1	1	1	0	2
0	1	1	1	0	1	1	0	1	1	0	0	1	1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	1	0	1	1	0	0	1	1	1	0	0	1	0	2
1	1	0	0	1	1	0	1	1	0	1	1	0	0	1	1	0	1	0	1	3
1	0	0	1	0	0	1	0	1	1	0	1	1	0	0	1	1	1	1	0	1
0	0	1	0	0	1	0	1	0	1	1	0	1	1	0	0	1	0	0	0	1
0	1	0	0	0	0	1	0	1	0	1	1	0	1	1	0	0	0	0	1	3
1	0	0	0	1	1	0	1	0	1	0	1	1	0	1	1	0	1	1	1	2
0	0	0	1	1	1	1	0	1	0	1	0	1	1	0	1	1	0	1	0	1
0	0	1	1	0	1	1	1	0	1	0	1	0	1	1	0	0	1	0	0	3
0	1	1	0	1	1	1	1	1	0	1	0	1	0	1	1	1	1	0	1	
																				34

Table 10: Results for the related IV attack

A_0	A_1	A_2	A_3	A_4	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	L_2	L_1	R_2	R_1	#Active S-boxes
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0		0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0		0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0		0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1		0	0	0	0
0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0		0	0	0	1
0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0		0	0	0	4
0	0	1	1	2	1	0	0	1	0	0	0	0	0	0	0		0	0	4	4
0	1	1	2	1	1	1	0	0	1	0	0	0	0	0	0		0	4	0	10
1	1	2	1	1	2	1	1	0	0	1	0	0	0	0	0		4	0	0	7
1	2	1	1	3	0	2	1	1	0	0	1	0	0	0	0		0	0	4	9
Total																				30

Table 11: Results for the related key/IV attack

A_0	A_1	A_2	A_3	A_4	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	L_2	L_1	R_2	R_1	#Active `Sub`
0	0	1	1	0	0	1	0	1	0	1	0	0	1	1	1	0	0	0	0	1
0	1	1	0	0	0	0	1	0	1	0	1	0	0	1	1	0	0	0	1	2
1	1	0	0	0	0	0	0	1	0	1	0	1	0	0	1	0	1	1	0	2
1	0	0	0	1	1	0	0	0	1	0	1	0	1	0	0	1	1	0	0	3
0	0	0	1	0	1	1	0	0	0	1	0	1	0	1	0	1	1	0	1	2
0	0	1	0	0	0	1	1	0	0	0	1	0	1	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1	1	0	0	0	1	0	1	0	0	0	0	0	0
1	0	0	0	0	1	1	0	1	1	0	0	0	1	0	1	0	0	0	0	1
0	0	0	0	0	0	1	1	0	1	1	0	0	0	1	0	0	0	0	1	2
0	0	0	0	1	1	0	1	1	0	1	1	0	0	0	1	0	0	1	1	1
0	0	0	1	0	1	1	0	1	1	0	1	1	0	0	0	0	0	1	0	1
0	0	1	0	0	0	1	1	0	1	1	0	1	1	0	0	0	0	0	1	2
0	1	0	0	1	0	0	1	1	0	1	1	0	1	1	0	0	0	1	1	1
1	0	0	1	1	0	0	0	1	1	0	1	1	0	1	1	0	0	1	0	0
0	0	1	1	1	0	0	0	0	1	1	0	1	1	0	1	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	1
1	1	1	1	0	1	0	0	0	0	0	1	1	0	1	1	0	1	0	0	2
1	1	1	0	1	1	1	0	0	0	0	0	1	1	0	1	1	1	0	0	1
1	1	0	1	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0
1	0	1	1	1	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0	1
0	1	1	1	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0	1	2
1	1	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	3
1	1	0	1	0	1	0	1	1	1	1	1	0	0	0	0	0	1	1	1	2
1	0	1	0	0	1	1	0	1	1	1	1	1	0	0	0	1	0	1	0	0
0	1	0	0	0	0	1	1	0	1	1	1	1	1	0	0	0	0	0	0	
Total																				30

Table 12: Results for the combined related attack

A_0	A_1	A_2	A_3	A_4	B_{10}	B_9	B_8	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	L_2	L_1	R_2	R_1	#Active S-boxes
0	0	4	4	0	0	4	0	0	2	4	0	4	0	4	4	0	0	0	0	4
0	4	4	0	0	0	0	4	0	0	2	4	0	4	0	4	0	0	0	4	8
4	4	0	0	0	0	0	0	4	0	0	2	4	0	4	0	0	4	4	0	6
4	0	0	0	0	4	0	0	0	4	0	0	2	4	0	4	4	4	0	0	8
0	0	0	0	0	0	4	0	0	0	4	0	0	2	4	0	4	0	0	4	4
0	0	0	0	4	0	0	4	0	0	0	4	0	0	2	4	0	0	4	0	0
0	0	0	4	0	0	0	0	4	0	0	0	4	0	0	2	0	0	0	0	0
0	0	4	0	2	2	0	0	0	4	0	0	0	4	0	0	0	0	0	0	0
0	4	0	2	2	2	2	0	0	0	4	0	0	0	4	0	0	0	0	0	2
Total																				32

6 Guess-and-Determine Attacks

Guess-and-determine is an attack strategy that has been successfully applied to many stream cipher proposals. In particular, the stream ciphers Sosemanuk [3] and SNOW [12] whose design philosophy shares common elements with KCipher-2 have been subjected to guess-and-determine attacks [13, 38].

The principle of a guess-and-determine attack is as follows. The attacker is assumed to have access to a sequence of contiguous key stream which has been created with one particular key, which he then attempts to recover. Depending on the attack setting, the IV is fixed as well or the attacker may be able to obtain keystream created with different known or even chosen IVs and the same key. Then, a guess for part of the internal state of the cipher is made. By using the equations describing the relation between the known keystream, the guessed and the unknown parts of the state, the attacker tries to determine the remaining unknown parts of the state. Once a new part of the state is known, the attacker can often verify his guess by checking consistency between multiple clockings, which usually improve the attack complexity. This procedure is then repeated until the entire state has been guessed or determined.

We now investigate the applicability of this type of attack to KCipher-2.

6.1 Word-Oriented Guess-and-Determine

Since the B register is controlled by the smaller A register, a straightforward application of the guess-and-determine approach to KCipher-2 involves guessing

$$\begin{gathered} A_t, A_{t+3}, A_{t+4} \\ \text{and} A_{t+2}[30, 31] \end{gathered}$$

at time t, which completely determines the influence of A on the update cycle of B and the L_i and R_i registers at time t. The goal is then to determine the values of the remaining parts of the state, namely

$$A_{t+1}, A_{t+2}[0, \ldots, 29], B_t, \ldots, B_{t+10},$$

and the values of L_i, R_i at clocks $t, \ldots, t+10$. We further assume that the keystream z_t^L, z_t^H from $t = 0, \ldots, 10$ is known. Once the complete internal state of KCipher-2 is known, the attacker can arbitrarily calculate forwards and backwards due to the invertibility of the state update. Note that so far, $3 \cdot 32 + 2 = 98$ bits have been guessed.

After guessing the above-mentioned values, the state is updated as follows. A and B are clocked:

$$\begin{aligned} A_{t+1+i} &= A_{t+i} \quad \text{for } i = 0, \ldots, 3 \\ A_{t+5} &= A_{t+3} \oplus \alpha_0 A_t \\ B_{t+1+i} &= B_{t+i} \quad \text{for } i = 0, \ldots, 9 \\ B_{t+11} &= c_1^t B_t \oplus B_{t+1} \oplus B_{t+6} \oplus c_2^t B_{t+8} \end{aligned} \tag{19}$$

and the FSM registers are updated:

$$\begin{aligned} R1_{t+1} &= Sub(L2_t \boxplus B_{t+9}) \\ L1_{t+1} &= Sub(R2_t \boxplus B_{t+4}) \\ R2_{t+1} &= Sub(R1_t) \\ L2_{t+1} &= Sub(L1_t), \end{aligned} \tag{20}$$

with constants c_1^t, c_2^t depending on the dynamic feedback bits which have already been guessed. Additionally, the following keystream words are generated:

$$\begin{aligned} z_t^L &= B_t \boxplus R2_t \oplus R1_t \oplus A_{t+4} \\ z_t^H &= B_{t+10} \boxplus L2_t \oplus L1_t \oplus A_t. \end{aligned} \tag{21}$$

We observe that the equations for the keystream words z_t^L and z_t^H only involve the FSM registers R_i and L_i, respectively; and we have $R2_{t+1} = Sub(R1_t)$ and $L2_{t+1} = Sub(L1_t)$. Consequently, after guessing $R1_0$, we know $R2_1 = Sub(R1_0)$. For the next clock, we need to guess only two bits of A, namely the new $A_{t+2}[30, 31]$. However, to use the keystream relation

$$z_1^L = B_1 \boxplus R2_1 \oplus R1_1 \oplus A_{t+5}, \tag{22}$$

we need to guess $R1_1$ to determine the value of B_1. Since we can already calculate $R2_2 = Sub(R1_1)$, we only need to clock once more and guess $R1_2$ to use the next keystream relation

$$z_2^L = B_2 \boxplus R2_2 \oplus R1_2 \oplus A_{t+6}.$$

in an analogous manner to determine B_2. To determine A_{t+6}, however, A_{t+1} has to be guessed. The same holds for A_{t+7} in the next clock ($t = 3$), so the attacker can equivalently guess the entire contents of the LFSR A in the beginning, accounting for a work factor of 2^{160}. In order to determine the entire contents of B, three more clockings with guesses of $R1$ have to be made. Afterwards, at $t = 6$, the following values are known:

$$R1_{0,\ldots,4}, R2_{1,\ldots,5}, L1_{0,\ldots,6}, L2_{1,\ldots,7}, B_{0,\ldots,5}.$$

Now we can use the relation for z_t^H to determine more words of the B register by

$$z_t^H = B_{t+10} \boxplus L2_t \oplus L1_t \oplus A_t,$$

which yields $B_{11,\ldots,16}$ and in turn the corresponding values of the $L2_t$ by

$$R1_{t+1} = Sub(L2_t \boxplus B_{t+9}),$$

which yields $L1_{1,\ldots,6}$ by

$$L1_t = Sub^{-1}(L2_t).$$

By substituting the new values obtained for $L1_t$ and $L2_t$ into

$$R1_{t+1} = Sub(L2_t \boxplus B_{t+9})$$

and

$$z_t^H = B_{t+10} \boxplus L2_t \oplus L1_t \oplus A_t,$$

we can determine all words of B except B_6, B_7, B_8. Finally, we can use the LFSR equation $B_{t+11} = c_1^t B_t \oplus B_{t+1} \oplus B_{t+6} \oplus c_2^t B_{t+8}$ (with the constants c_1^t and c_2^t depending on the already guessed bits of A) to determine the last three remaining words of B. In total, this implies that ten 32-bit words have to be guessed, which accounts for a complexity of 2^{320} and makes this approach inferior to brute force of the 128-bit key.

6.2 Byte-Oriented Guess-and-Determine

It is sometimes possible to reduce the amount of material that needs to be guessed by decomposing the state update at byte (or bit) level instead of operating on words. The complexity of the attack then depends on the number of bytes (bits) that have to be guessed in order to determine the solution of the first smaller subsystem. In this section, we simplify the cipher by replacing all modular additions ($\boxplus$) by XORs. We use $[W]_i$ to denote byte $i = 0, \ldots, 3$ of a 32-bit word W, with W_3 denoting the most significant byte.

One then obtains the following systems of equations for one step. The clocking of A is described as

$$[A_{t+1+k}]_i = [A_{t+k}]_i \quad \text{for } k = 0, \ldots, 3 \text{ and } i = 0, \ldots, 3.$$

For $i = 1, \ldots, 3$ we have

$$\begin{aligned}
[A_{t+5}]_i &= [A_{t+3}]_i \oplus [\alpha_0 A_t]_i \\
&= [A_{t+3}]_i \oplus \left[\alpha_0 [A_t]_3 \, \alpha_0^3 \oplus \cdots \oplus \alpha_0 [A_t]_0\right]_i \\
&= [A_{t+3}]_i \oplus [A_t]_{(i-1)} \oplus [T_{\alpha_0}[[A_t]_3]]_i \,,
\end{aligned}$$

with a 8-to-32 bit lookup table T_{α_0}. This becomes

$$\begin{aligned}
[A_{t+5}]_0 &= [A_{t+3}]_0 \oplus [\alpha_0 A_t]_0 \\
&= [A_{t+3}]_0 \oplus [T_{\alpha_0}[[A_t]_3]]_0 \,,
\end{aligned}$$

for the least significant byte $i = 0$. The last equalities follows from the special choice of the field representations used in KCipher-2 that allows implementing multiplication by α_0 by a left shift by one byte and a table lookup based on the most significant byte.

The update of register B is analogously described by

$$[B_{t+1+k}]_i = [B_{t+k}]_i \quad \text{for } k = 0, \ldots, 9 \text{ and } i = 0, \ldots, 3.$$

For $i = 1, \ldots, 3$, we have

$$\begin{aligned}
[B_{t+11}]_i &= \left[c_1^t B_t\right]_i \oplus [B_{t+1}]_i \oplus [B_{t+6}]_i \oplus \left[c2^t B_{t+8}\right]_i \\
&= \left[c_1^t [B_t]_3 \, \alpha_0^3 \oplus \cdots \oplus c_1^t [B_t]_0\right]_i \oplus [B_{t+1}]_i \oplus [B_{t+6}]_i \oplus \\
&\quad \left[c_2^t [B_{t+8}]_3 \, \alpha_0^3 \oplus \cdots \oplus c_2^t [B_{t+8}]_0\right]_i \\
&= [B_t]_{(i-1)} \oplus [T_{c_1}[[B_t]_3]]_i \oplus [B_{t+1}]_i \oplus [B_{t+6}]_i \oplus [B_{t+8}]_{(i-1)} \oplus [T_{c_2}[[B_{t+8}]_3]]_i \,,
\end{aligned}$$

with 8-to-32 bit lookup tables $T_{\alpha_0}, T_{c_1}, T_{c_2}$ for multiplication in $GF(2^{32})$ by those constants. This becomes

$$\begin{aligned}[B_{t+11}]_0 &= \left[c_1^t B_t\right]_0 \oplus [B_{t+1}]_0 \oplus [B_{t+6}]_0 \oplus \left[c2^t B_{t+8}\right]_0 \\ &= [T_{c_1}[[B_t]_3]]_0 \oplus [B_{t+1}]_0 \oplus [B_{t+6}]_0 \oplus [T_{c_2}[[B_{t+8}]_3]]_0\end{aligned}$$

for the least significant byte $i = 0$.

The FSM update equations are given by

$$\begin{aligned}[R1_{t+1}]_i &= [Sub(L2_t \oplus B_{t+9})]_i \\ &= 2 \cdot S([L2_t \oplus B_{t+9}]_{(3-i) \bmod 4}) \oplus 3 \cdot S([L2_t \oplus B_{t+9}]_{(2-i) \bmod 4}) \\ &\quad \oplus S([L2_t \oplus B_{t+9}]_{(1-i) \bmod 4}) \oplus S([L2_t \oplus B_{t+9}]_{(-i) \bmod 4}) \\ [L1_{t+1}]_i &= [Sub(R2_t \oplus B_{t+4})]_i \\ &= 2 \cdot S([R2_t \oplus B_{t+4}]_{(3-i) \bmod 4}) \oplus 3 \cdot S([R2_t \oplus B_{t+4}]_{(2-i) \bmod 4}) \\ &\quad \oplus S([R2_t \oplus B_{t+4}]_{(1-i) \bmod 4}) \oplus S([R2_t \oplus B_{t+4}]_{(-i) \bmod 4}) \\ [R2_{t+1}]_i &= [Sub(R1_t)]_i \\ &= 2 \cdot S([R1_t]_{(3-i) \bmod 4}) \oplus 3 \cdot S([R1_t]_{(2-i) \bmod 4}) \\ &\quad \oplus S([R1_t]_{(1-i) \bmod 4}) \oplus S([R1_t]_{(-i) \bmod 4}) \\ [L2_{t+1}]_i &= [Sub(L1_t)]_i \\ &= 2 \cdot S([L1_t]_{(3-i) \bmod 4}) \oplus 3 \cdot S([L1_t]_{(2-i) \bmod 4}) \\ &\quad \oplus S([L1_t]_{(1-i) \bmod 4}) \oplus S([L1_t]_{(-i) \bmod 4}),\end{aligned}$$

for all $i = 0, \ldots, 3$. The byte-level keystream equations are

$$\begin{aligned}\left[z_t^L\right]_i &= [B_t]_i \oplus [R2_t]_i \oplus [R1_t]_i \oplus [A_{t+4}]_i \\ \left[z_t^H\right]_i &= [B_{t+10}]_i \oplus [L2_t]_i \oplus [L1_t]_i \oplus [A_t]_i\end{aligned}$$

for $i = 0, \ldots, 3$ in each clock.

We observe that each keystream relation of the type (22) involves a word $R1_t = Sub(L2_t \oplus B_{t+9})$ and $R2_t = Sub(R1_t)$. Hence, in order to determine one byte of $B_{t'}$, four bytes of $R1_t$ have to be guessed, which effectively couples four formerly independent systems of equations at byte level. Consequently, it seems that the byte-oriented approach results in exactly the same complexity as the word-level approach.

7 Period Considerations

While the feedback polynomial of the LFSR A is primitive, the dynamic feedback operation causes the feedback polynomial of the B register to be primitive only in two out of four cases:

$$(\alpha_1^{c1_t} + \alpha_2^{1-c1_t} - 1)B_t \oplus B_{t+1} \oplus B_{t+6} \oplus \alpha_3^{c2_t} B_{t+8} \tag{23}$$

is not primitive whenever $c2_t = 1$.

Even if the individual periods in the two cases $c1_t = 0, c2_t = 1$ and $c1_t = c2_t = 1$ are sufficiently long, there is a possibility of having a short period for nontrivial sequences of $c1_t$ and $c2_t$. Since B is an LFSR, this can be efficiently checked by solving a linear system of equations for each possible $2k$-bit sequence of $c1_t$ and $c2_t$, with $k \geq 11$. Since the work factor amounts to 2^{2k} Gaussian eliminations, we have so far only checked for this periodicity property up to $k = 14$. No short periods of those lengths k have been found.

8 Statistical Properties

A great deal of statistical tests have been developed to discover various types of irregularities on the bit sequences produced by stream ciphers. In their cryptanalytic essence, statistical tests are generic distinguishing attacks that do not take the concrete structure of the cipher into account. However, several types of tests do aim at some specific statistical properties for classes of stream cipher designs, e.g. the linear properties for LFSR-based designs in the linear complexity test.

If a stream cipher fails at least a single test of any test suite, this immediately means that the cipher is structurally flawed. The fact that a stream cipher passes a vast suite of statistical tests can be seen as a rather rough indicator for the soundness of its design. Apart from the structural unawareness, a major limitation of any feasibly computable test from a generic test suite is its low distinguishing power, since severe limitations exist regarding its complexity. For example, if a test requires no more than 2^{20} operations, it cannot detect any distinguishing attack of a higher complexity. If a cipher passes a suite of tests, this fact cannot be seen as anything more than the absence of a subset of structural flaws.

As a part of our analysis effort, we applied the statistical test suite [32] developed by NIST for evaluating random number generators for cryptographic applications to K2. Note that a previous version of this test suite was used by NIST to analyze the statistical properties of the AES candidate algorithms.

A typical statistical test focuses on a property of a bit sequence, computes a specific statistic for this property based on a sample sequence obtained from the tested sequence generator (such as a stream cipher or a random number generator), and compares the values of the computed statistic to those expected for a randomly drawn bit sequence. The statistical proximity of the bit sequence tested to a randomly drawn bit sequence is usually measured by the P-value which is interpreted as the probability that a randomly drawn sequence would behave worse than the tested sequence with respect to the test statistic.

To perform the analysis of statistical properties for K2, we used the K2 reference implementation to generate 100 output bit streams (with 100 different combinations of key and IV values) of length 10^6 bits each and tested their (pseudo)randomness using the NIST suite comprising the following types of tests (see [32] for test description):

- Monobit Frequency Test,
- Block Frequency Test for block length 128,

Table 13: P-values for tests from the NIST statistical test suite

Test	**P-value**
Frequency	0.616305
BlockFrequency	0.699313
CumulativeSums	0.657933
Runs	0.616305
LongestRun	0.798139
Rank	0.699313
FFT	0.494392
NonOverlappingTemplate	0.798139
OverlappingTemplate	0.040108
Universal	0.816537
ApproximateEntropy	0.739918
RandomExcursions	0.689019
RandomExcursionsVariant	0.941144
Serial	0.759756
LinearComplexity	0.994250

- Runs Test,
- Test for the Longest Run of Ones in a Block,
- Binary Matrix Rank Test,
- Spectral Test,
- Non-overlapping Template Matching Test for block length 9,
- Overlapping Template Matching Test for block length 9,
- Maurer's Universal Statistical Test,
- Linear Complexity Test for block length 500,
- Serial Test for block length 16,
- Approximate Entropy Test for block length 10,
- Cumulative Sums Test,
- Random Excursions Test, and
- Random Excursions Variant Test.

These tests yielded P-values provided in Table 13. Thus, the bit sequences generated with K2 passed all tests implemented by the NIST statistical test suite at the significance level $\alpha = 0.01$. The applied statistical tests did not reveal any structural flaws in the design of K2.

9 Distinguishing Attacks

9.1 Preliminaries

At first we provide some background on distinguishing attacks and mod n cryptanalysis.

Distinguishing Attacks: In a distinguishing attack, the attacker is able to tell apart a cipher from an ideal cipher. The adversary (i.e., an algorithm) that does this job, taking the outputs of the examined cipher as inputs, is called a distinguisher. A distinguisher works when the output of a cipher is biased; its efficiency depends on the magnitude(s) of the bias(es). The efficiency of the distinguisher is measured by a parameter called the *advantage*. Let Z denote an t-tuple, $(z_1, \ldots, z_t)$, where each z_i is a function of one or more output bits of a cipher. Each value z_i is generated independently of another, under a key (or an alternative input to the cipher) chosen uniformly at random. Suppose that a distinguishing algorithm A, that takes Z as input, outputs '*Cipher*' if it identifies Z as coming from the cipher and '*Random*' if it identifies Z as coming from an ideal source. Let Z_{Cipher} (resp. Z_{Ideal}) denote the event that Z is generated by the cipher (resp. an ideal source). Then, the advantage Adv of A is given by:

$$Adv(A) = |Pr(A(Z) = Cipher|Z_{Cipher}) - Pr(A(Z) = Cipher|Z_{Ideal})|.$$

A distinguishing attack that requires an extremely large t for a considerable advantage may not be seen as very useful. Nevertheless, the analysis that leads to the detection of biased output may give good insights into the structural properties of the cipher and could serve as a launch pad for more serious attacks.

A distinguishing attack may be further developed into a key or state recovery attack. Examples include the attacks on ESSENCE [25] and Moustique [17] for key recovery and the Künzli-Meier distinguishing attack on the stream cipher MAG for state recovery [20].

Mod n Cryptanalysis: This technique was introduced by Kelsey *et al.* in [18]. It is a kind of statistical distinguishing attack that has been used against several block ciphers (e.g., RC5P, M6 [18]) and stream ciphers (e.g., Rabbit [5,29]) that are based on modular additions, bit-shifts and bit-rotations.

In the case of bit-rotation, the idea behind the mod n cryptanalysis technique is the following. Let X be some 32-bit integer and Y denote $X \lll 1$. Then,

$$Y = \begin{cases} 2X & \text{if } X < 2^{31}, \\ 2X + 1 - 2^{32} & \text{if } X \geq 2^{31}, \end{cases} \tag{24}$$

and hence,

$$Y \bmod n = \begin{cases} 2X \bmod n & \text{if } X < 2^{31}, \\ 2X + 1 - 2^{32} \bmod n & \text{if } X \geq 2^{31}. \end{cases} \tag{25}$$

Therefore, when $2^{32} - 1 \equiv 0 \bmod n$, we have $Y = 2X \bmod n$ irrespective of the value of X. Else, every $X \bmod n$ results in one of the two values of $Y \bmod n$ in (25) with equal probability.

In the case of addition modulo 2^{32} (denoted by the symbol $\boxplus$), the idea is as follows. Let $S = X \boxplus Y$. Then,

$$S = \begin{cases} X + Y & \text{with probability } 0.5 + 2^{-33}, \\ X + Y - 2^{32} & \text{with probability } 0.5 - 2^{-33}. \end{cases} \tag{26}$$

The probabilities in (26) were computed using [28, (18)],[1] under the assumption that X and Y are independent and uniformly distributed. Here again, when $2^{32} - 1 \equiv 0 \bmod n$, we have $S \bmod n = X + Y \bmod n$, irrespective of the values of X and Y. Even otherwise, this relation may hold with very high probability; for example when the 4 most significant bits of X (or Y) are all 1's, then the probability is 0.98.

9.2 Mod n Cryptanalysis of Weakened KCipher-2

In the independent evaluation of KCipher-2 by the Institute for Infocomm Research (I^2R), Singapore, guess-and-determine attacks have been considered. Here, the analysts reduce the cipher as follows. First, they assume that the entire contents of FSR-A are guessed (in their linear cryptanalysis they disregard FSR-A). This comes with a cost of $O(2^{160})$ time. Next, they replace XOR with $\boxplus$ in the keystream generation steps and ignore the α coefficients (in the feedback polynomials)[2] and the Sub transformations. The keystream generation algorithm of this reduced cipher is represented by the following equations:

$$\begin{aligned} z_t^H &= B_{t+10} \boxplus L2_t \boxplus L1_t, && (27) \\ z_t^L &= B_t \boxplus R2_t \boxplus R1_t, && (28) \\ L2_{t+1} &= L1_t, \\ R2_{t+1} &= R1_t, \\ L1_{t+1} &= R2_t \boxplus B_{t+4}, \\ R1_{t+1} &= L2_t \boxplus B_{t+9}. \end{aligned}$$

When the keystream words take the forms (27) and (28), it is likely that they are biased when reduced modulo 3. Any 32-bit integer chosen from $\{0, \ldots, 2^{32} - 1\}$ is congruent 1 mod 3 with probability $(2^{32} - 1)/(3 \cdot 2^{32})$, 2 mod 3 with the same probability and 0 mod 3 with probability $(2^{32} + 2)/(3 \cdot 2^{32})$. Therefore, the summands in (27) and (28) are biased when reduced modulo 3. If the carry terms generated in the equations reduced modulo 3 are biased as well, $z_t^H \bmod 3$ and $z_t^L \bmod 3$ are biased. A point to be noted here is that the carries cannot be immediately assumed to be biased because the summands modulo 3 are not uniformly distributed and consequently the formulae of [28,36] cannot be applied. Carry biases are best determined experimentally; an analytic evaluation is expected to be quite involved.

We performed simulations of modular additions taking the summands of (27) to be b-bit integers, where $4 \leq b \leq 8$. Our results indicate that the overall bias is preserved by modular addition. In other words, additional summands do not affect the bias. For example, when $b = 8$, the following probabilities were obtained:

[1] A generic treatment of carry propagation in modular addition is given in [36].

[2] In their linear cryptanalysis they fix these coefficients.

- $Pr(z_t^H \equiv 1 \bmod 3) = (2^{24} - 2^{16})/(3 \cdot 2^{24}) = (1 - 2^{-8})/3$
- $Pr(z_t^H \equiv 2 \bmod 3) = (2^{24} - 2^{16})/(3 \cdot 2^{24}) = (1 - 2^{-8})/3$,
- $Pr(z_t^H \equiv 0 \bmod 3) = (2^{24} + 2^{17})/(3 \cdot 2^{24}) = (1 + 2^{-7})/3$.

The respective probabilities when one (8-bit) summand is removed from (27) are:

- $Pr(z_t^H \equiv 1 \bmod 3) = (2^{16} - 2^{8})/(3 \cdot 2^{16}) = (1 - 2^{-8})/3$
- $Pr(z_t^H \equiv 2 \bmod 3) = (2^{16} - 2^{8})/(3 \cdot 2^{16}) = (1 - 2^{-8})/3$,
- $Pr(z_t^H \equiv 0 \bmod 3) = (2^{16} + 2^{9})/(3 \cdot 2^{16}) = (1 + 2^{-7})/3$.

We see that the probabilities, though biased, are not different in the two cases. Intuitively, it appears that the result could be very well extended to 32-bit summands. For instance, our simulation yielded the following probabilities for 4-bit summands:

- $Pr(z_t^H \equiv 1 \bmod 3) = (1 - 2^{-4})/3$
- $Pr(z_t^H \equiv 2 \bmod 3) = (1 - 2^{-4})/3$,
- $Pr(z_t^H \equiv 0 \bmod 3) = (1 + 2^{-3})/3$.

The respective magnitudes of the biases in the 4-bit and 8-bit cases are different. Given this, the only information that the attacker can gain from the biases is the size of each summand in the keystream generation equation (27). However, this information does not appear to be useful, especially considering distinguishing attacks.

9.2.1 Other Reduced Versions of KCipher-2

We may replace (27) and (28) in one of the following ways and arrive at a similar result as above (i.e., likely biases in z_t^H mod 3 and z_t^L mod 3).

- Include FSR-A:

$$z_t^H = B_{t+10} \boxplus L2_t \boxplus L1_t \boxplus A_t, \quad (29)$$
$$z_t^L = B_t \boxplus R2_t \boxplus R1_t \boxplus A_{t+4}. \quad (30)$$

- Assume guessed FSR-A but approximate the equations differently as:

$$z_t^H = B_{t+10} \boxplus (L2_t \oplus L1_t), \quad (31)$$
$$z_t^L = B_t \boxplus (R2_t \oplus R1_t). \quad (32)$$

- Include FSR-A and approximate the equations in the same way as in (31) and (32):

$$z_t^H = B_{t+10} \boxplus (L2_t \oplus L1_t \oplus A_t), \quad (33)$$
$$z_t^L = B_t \boxplus (R2_t \oplus R1_t \oplus A_{t+4}). \quad (34)$$

Here again, our simulations yielded similar results as in Section 9.2 – the keystreams are biased but the biases seem useless in building distinguishing attacks.

9.3 Analysis of the Original KCipher-2

In Sect. 9.2, we analysed several reduced variants of KCipher-2. If $Z = X \oplus Y$, then when $n = 3$ or 5 the distribution of $Z \bmod n$ is, from our experiments, close to uniform.

The approximations in (33) and (34) each hold with probability 2^{-64}. This is because only when $L2_t = 0$ and $L1_t \oplus A_t = 0$, z_t^H can be approximated as (33); if the events are independent and $L1_t$ (or A_t), $L2_t$ are uniformly distributed at random, the approximation holds with probability 2^{-64} (also confirmed by our simulations). Similar arguments apply for the approximation in (34). The probability 2^{-64} is very small for a distinguisher of meaningful complexity to be built (especially if the key size is 128 bits).

Based on the analysis presented in Sections 9.2 and 9.3, KCipher-2 and some of its weakened variants seem to offer good resistance against distinguishing attacks based on the mod n cryptanalysis technique.

10 Conclusions

We attempted to apply a wide spectrum of state-of-the-art cryptanalytic techniques to K2 and its simplified versions. We have not been able to identify any weaknesses and we conclude that K2 possesses a sound design.

Acknowledgements. A part of this research is supported by Cryptography Research and Evaluation Committee (CRYPTREC). Andrey Bogdanov is supported by a Postdoctoral Fellowship from the Flemish Research Foundation (FWO-Vlaanderen). Nicky Mouha is funded by a research grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). Elmar Tischhauser is supported by a Ph.D. Fellowship from the Flemish Research Foundation (FWO-Vlaanderen).

References

[1] Ars, G. and Faugère, J.C. An Algebraic Cryptanalysis of Nonlinear Filter Generators using Groebner Bases. INRIA Research Report, n 4739, 2003.

[2] M. Bellare, editor. *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, volume 1880 of *Lecture Notes in Computer Science*. Springer, 2000.

[3] C. Berbain, O. Billet, A. Canteaut, N. Courtois, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin, and H. Sibert. Sosemanuk, a fast software-oriented stream cipher. In M. J. B. Robshaw and O. Billet, editors, *The eSTREAM Finalists*, volume 4986 of *Lecture Notes in Computer Science*, pages 98–118. Springer, 2008.

[4] O. Billet and H. Gilbert. Resistance of SNOW 2.0 against algebraic attacks. In A. Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 19–28. Springer, 2005.

[5] M. Boesgaard, M. Vesterager, T. Pedersen, J. Christiansen, and O. Scavenius. Rabbit: A New High-Performance Stream Cipher. In T. Johansson, editor, *FSE*, volume 2887 of *Lecture Notes in Computer Science*, pages 307–329. Springer, 2003.

[6] P. Camion, C. Carlet, P. Charpin, and N. Sendrier. On correlation-immune functions. In J. Feigenbaum, editor, *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 86–100. Springer, 1991.

[7] C. Carlet. On the coset weight divisibility and nonlinearity of resilient and correlation-immune functions. In *SEquences and Their Applications – SETA 2001, Revised Selected Papers*, pages 131–144. Springer-Verlag, 2002.

[8] V. V. Chepyzhov, T. Johansson, and B. J. M. Smeets. A Simple Algorithm for Fast Correlation Attacks on Stream Ciphers. In B. Schneier, editor, *FSE*, volume 1978 of *Lecture Notes in Computer Science*, pages 181–195. Springer, 2000.

[9] D. Coppersmith, S. Halevi, and C. S. Jutla. Cryptanalysis of stream ciphers with linear masking. In M. Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 515–532. Springer, 2002.

[10] N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In E. Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359. Springer, 2003.

[11] N. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In Y. Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer, 2002.

[12] P. Ekdahl and T. Johansson. A new version of the stream cipher SNOW. In Nyberg and Heys [26], pages 47–61.

[13] P. Hawkes and G. G. Rose. Guess-and-determine attacks on SNOW. In Nyberg and Heys [26], pages 37–46.

[14] T. Johansson and F. Jönsson. Fast correlation attacks based on turbo code techniques. In M. J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 181–197. Springer, 1999.

[15] T. Johansson and F. Jönsson. Improved fast correlation attacks on stream ciphers via convolutional codes. In *EUROCRYPT*, pages 347–362, 1999.

[16] T. Johansson and F. Jönsson. Fast correlation attacks through reconstruction of linear polynomials. In Bellare [2], pages 300–315.

[17] E. Käsper, V. Rijmen, T. E. Bjørstad, C. Rechberger, M. J. B. Robshaw, and G. Sekar. Correlated Keystreams in Moustique. In S. Vaudenay, editor, *AFRICACRYPT*, volume 5023 of *Lecture Notes in Computer Science*, pages 246–257. Springer, 2008.

[18] J. Kelsey, B. Schneier, and D. Wagner. Mod n Cryptanalysis, with Applications Against RC5P and M6. In L. R. Knudsen, editor, *FSE*, volume 1636 of *Lecture Notes in Computer Science*, pages 139–155. Springer, 1999.

[19] S. Kiyomoto, T. Tanaka, and K. Sakurai. K2: A stream cipher algorithm using dynamic feedback control. In J. Hernando, E. Fernández-Medina, and M. Malek, editors, *SECRYPT*, pages 204–213. INSTICC Press, 2007.

[20] S. Künzli and W. Meier. Distinguishing Attack on MAG. Technical Report 2005/053, eSTREAM, July 2005. Available at `http://www.ecrypt.eu.org/stream/papers.html`.

[21] J. L. Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15(1):122–127, 1969.

[22] M. Matsui. Linear cryptoanalysis method for DES cipher. In *EUROCRYPT*, pages 386–397, 1993.

[23] W. Meier and O. Staffelbach. Fast correltaion attacks on stream ciphers (extended abstract). In *EUROCRYPT*, pages 301–314, 1988.

[24] W. Meier and O. Staffelbach. Fast correlation attacks on certain stream ciphers. *J. Cryptology*, 1(3):159–176, 1989.

[25] N. Mouha, G. Sekar, J.-P. Aumasson, T. Peyrin, S. S. Thomsen, M. S. Turan, and B. Preneel. Cryptanalysis of the ESSENCE Family of Hash Functions. In F. Bao, M. Yung, D. Lin, and J. Jing, editors, *Inscrypt*, volume 6151 of *Lecture Notes in Computer Science*, pages 15–34. Springer, 2009.

[26] K. Nyberg and H. M. Heys, editors. *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers*, volume 2595 of *Lecture Notes in Computer Science*. Springer, 2003.

[27] K. Nyberg and J. Wallén. Improved linear distinguishers for SNOW 2.0. In Robshaw [30], pages 144–162.

[28] S. Paul, B. Preneel, and G. Sekar. Distinguishing Attacks on the Stream Cipher Py. In Robshaw [30], pages 405–421.

[29] V. Rijmen. mod n Cryptanalysis of Rabbit. Technical Report Version 1.0, CRYPTICO A/S, December 2003. Available at `http://www.cryptico.com/DWSDownload.asp?File=Files%2Ffiler%2Fwp_modn_analysis.pdf`.

[30] M. J. B. Robshaw, editor. *Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers*, volume 4047 of *Lecture Notes in Computer Science*. Springer, 2006.

[31] R. A. Rueppel and O. Staffelbach. Products of linear recurring sequences with maximum complexity. *IEEE Transactions on Information Theory*, 33(1):124–131, 1987.

[32] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST Special Publication 800-22. Revision 1a, 2010.

[33] P. Sarkar and S. Maitra. Nonlinearity bounds and constructions of resilient Boolean functions. In Bellare [2], pages 515–532.

[34] T. Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE Transactions on Information Theory*, 30(5):776–, 1984.

[35] T. Siegenthaler. Decrypting a class of stream ciphers using ciphertext only. *IEEE Trans. Computers*, 34(1):81–85, 1985.

[36] O. Staffelbach and W. Meier. Cryptographic Significance of the Carry for Ciphers Based on Integer Addition. In A. Menezes and S. A. Vanstone, editors, *CRYPTO*, volume 537 of *Lecture Notes in Computer Science*, pages 601–614. Springer, 1990.

[37] Y. Tarannikov. On resilient Boolean functions with maximal possible nonlinearity. In B. K. Roy and E. Okamoto, editors, *INDOCRYPT*, volume 1977 of *Lecture Notes in Computer Science*, pages 19–30. Springer, 2000.

[38] Y. Tsunoo, T. Saito, M. Shigeri, T. Suzaki, H. Ahmadi, T. Eghlidos, and S. Khazaei. Evaluation of Sosemanuk with regard to guess-and-determine attacks. In *State of the Art in Stream Ciphers, Leuven, Belgium*, February 2006.

[39] D. Watanabe, A. Biryukov, and C. D. Cannière. A distinguishing attack of SNOW 2.0 with linear masking method. In M. Matsui and R. J. Zuccherato, editors, *Selected Areas in Cryptography*, volume 3006 of *Lecture Notes in Computer Science*, pages 222–233. Springer, 2003.

[40] G.-Z. Xiao and J. L. Massey. A spectral characterization of correlation-immune combining functions. *IEEE Transactions on Information Theory*, 34(3):569–, 1988.

[41] Y. Zheng and X.-M. Zhang. Improved upper bound on the nonlinearity of high order correlation immune functions. In D. R. Stinson and S. E. Tavares, editors, *Selected Areas in Cryptography*, volume 2012 of *Lecture Notes in Computer Science*, pages 262–274. Springer, 2000.

A Linear Attack

In this section the equations are listed which are used in the linear attack analysis (Section 2).

A.1 Linear relations for FSR-A

Linear relations for FSR-A (3) using masks Γ, $\Gamma\alpha_0$, $\Gamma\alpha_1$ and $\Gamma\alpha_0\alpha_1$ for clocks 0,1,3,4,5,6,7,9,10,11,12,13,15,16:

$$
\begin{aligned}
(\Gamma\alpha_1\alpha_0)A_0 \oplus (\Gamma\alpha_1)A_3 \oplus (\Gamma\alpha_1)A_5 &= 0 \ , \\
(\Gamma\alpha_0)A_1 \oplus \Gamma A_4 \oplus \Gamma A_6 &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)A_1 \oplus (\Gamma\alpha_1)A_4 \oplus (\Gamma\alpha_1)A_6 &= 0 \ , \\
(\Gamma\alpha_0)A_2 \oplus \Gamma A_5 \oplus \Gamma A_7 &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)A_4 \oplus (\Gamma\alpha_1)A_7 \oplus (\Gamma\alpha_1)A_9 &= 0 \ , \\
(\Gamma\alpha_0)A_5 \oplus \Gamma A_8 \oplus \Gamma A_{10} &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)A_5 \oplus (\Gamma\alpha_1)A_8 \oplus (\Gamma\alpha_1)A_{10} &= 0 \ , \\
(\Gamma\alpha_0)A_7 \oplus \Gamma A_{10} \oplus \Gamma A_{12} &= 0 \ , \\
(\Gamma\alpha_0)A_8 \oplus \Gamma A_{11} \oplus \Gamma A_{13} &= 0 \ , \\
(\Gamma\alpha_0)A_9 \oplus \Gamma A_{12} \oplus \Gamma A_{14} &= 0 \ , \\
(\Gamma\alpha_0)A_{10} \oplus \Gamma A_{13} \oplus \Gamma A_{15} &= 0 \ , \\
(\Gamma\alpha_0)A_{13} \oplus \Gamma A_{16} \oplus \Gamma A_{18} &= 0 \ , \\
(\Gamma\alpha_0)A_{15} \oplus \Gamma A_{18} \oplus \Gamma A_{20} &= 0 \ , \\
(\Gamma\alpha_0)A_{16} \oplus \Gamma A_{19} \oplus \Gamma A_{21} &= 0 \ .
\end{aligned}
$$

A.2 Linear relations for FSR-B

Linear relations for FSR-B (4) using masks Γ, $\Gamma\alpha_0$, $\Gamma\alpha_1$ and $\Gamma\alpha_0\alpha_1$ for clocks 0,1,3,4,5,6,7,9,10,11,12,13,15,16:

$$
\begin{aligned}
(\Gamma\alpha_1\alpha_0)B_0 \oplus (\Gamma\alpha_0)B_1 \oplus (\Gamma\alpha_0)B_6 \oplus (\Gamma\alpha_0)B_8 \oplus (\Gamma\alpha_0)B_{11} &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)B_1 \oplus (\Gamma\alpha_0)B_2 \oplus (\Gamma\alpha_0)B_7 \oplus (\Gamma\alpha_0)B_9 \oplus (\Gamma\alpha_0)B_{12} &= 0 \ , \\
(\Gamma\alpha_1)B_3 \oplus \Gamma B_4 \oplus \Gamma B_9 \oplus \Gamma B_{11} \oplus \Gamma B_{14} &= 0 \ , \\
(\Gamma\alpha_1)B_4 \oplus \Gamma B_5 \oplus \Gamma B_{10} \oplus \Gamma B_{12} \oplus \Gamma B_{15} &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)B_4 \oplus (\Gamma\alpha_0)B_5 \oplus (\Gamma\alpha_0)B_{10} \oplus (\Gamma\alpha_0)B_{12} \oplus (\Gamma\alpha_0)B_{15} &= 0 \ , \\
(\Gamma\alpha_1)B_5 \oplus \Gamma B_6 \oplus \Gamma B_{11} \oplus \Gamma B_{13} \oplus \Gamma B_{16} &= 0 \ , \\
(\Gamma\alpha_1)B_6 \oplus \Gamma B_7 \oplus \Gamma B_{12} \oplus \Gamma B_{14} \oplus \Gamma B_{17} &= 0 \ , \\
(\Gamma\alpha_1)B_7 \oplus \Gamma B_8 \oplus \Gamma B_{13} \oplus \Gamma B_{15} \oplus \Gamma B_{18} &= 0 \ , \\
(\Gamma\alpha_1)B_9 \oplus \Gamma B_{10} \oplus \Gamma B_{15} \oplus \Gamma B_{17} \oplus \Gamma B_{20} &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)B_9 \oplus (\Gamma\alpha_0)B_{10} \oplus (\Gamma\alpha_0)B_{15} \oplus (\Gamma\alpha_0)B_{17} \oplus (\Gamma\alpha_0)B_{20} &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)B_{10} \oplus (\Gamma\alpha_0)B_{11} \oplus (\Gamma\alpha_0)B_{16} \oplus (\Gamma\alpha_0)B_{18} \oplus (\Gamma\alpha_0)B_{21} &= 0 \ , \\
(\Gamma\alpha_1\alpha_0)B_{11} \oplus (\Gamma\alpha_0)B_{12} \oplus (\Gamma\alpha_0)B_{17} \oplus (\Gamma\alpha_0)B_{19} \oplus (\Gamma\alpha_0)B_{22} &= 0 \ , \\
(\Gamma\alpha_1)B_{12} \oplus \Gamma B_{13} \oplus \Gamma B_{18} \oplus \Gamma B_{20} \oplus \Gamma B_{23} &= 0 \ , \\
(\Gamma\alpha_1)B_{13} \oplus \Gamma B_{14} \oplus \Gamma B_{19} \oplus \Gamma B_{21} \oplus \Gamma B_{24} &= 0 \ , \\
(\Gamma\alpha_1)B_{15} \oplus \Gamma B_{16} \oplus \Gamma B_{21} \oplus \Gamma B_{23} \oplus \Gamma B_{26} &= 0 \ , \\
(\Gamma\alpha_1)B_{16} \oplus \Gamma B_{17} \oplus \Gamma B_{22} \oplus \Gamma B_{24} \oplus \Gamma B_{27} &= 0 \ .
\end{aligned}
$$

A.3 Linear approximation of the NLF

Linear approximation of the NLF using 13 relations of type (5):

$$
\begin{aligned}
&(\Gamma\alpha_1\alpha_0)A_0 \oplus (\Gamma\alpha_1\alpha_0)A_1 \oplus (\Gamma\alpha_1\alpha_0)A_4 \oplus (\Gamma\alpha_1\alpha_0)A_5 \oplus \\
&(\Gamma\alpha_1\alpha_0)B_0 \oplus (\Gamma\alpha_1\alpha_0)B_1 \oplus (\Gamma\alpha_1\alpha_0)B_4 \oplus (\Gamma\alpha_1\alpha_0)B_9 \oplus \\
&(\Gamma\alpha_1\alpha_0)B_{10} \oplus (\Gamma\alpha_1\alpha_0)B_{11} = \\
&(\Gamma\alpha_1\alpha_0)z_0^H \oplus (\Gamma\alpha_1\alpha_0)z_0^L \oplus (\Gamma\alpha_1\alpha_0)z_1^H \oplus (\Gamma\alpha_1\alpha_0)z_1^L
\end{aligned} \tag{35}
$$

$$
\begin{aligned}
&(\Gamma\alpha_0)A_1 \oplus (\Gamma\alpha_0)A_2 \oplus (\Gamma\alpha_0)A_5 \oplus (\Gamma\alpha_0)A_6 \oplus \\
&(\Gamma\alpha_0)B_1 \oplus (\Gamma\alpha_0)B_2 \oplus (\Gamma\alpha_0)B_5 \oplus (\Gamma\alpha_0)B_{10} \oplus \\
&(\Gamma\alpha_0)B_{11} \oplus (\Gamma\alpha_0)B_{12} = \\
&(\Gamma\alpha_0)z_1^H \oplus (\Gamma\alpha_0)z_1^L \oplus (\Gamma\alpha_0)z_2^H \oplus (\Gamma\alpha_0)z_2^L
\end{aligned} \tag{36}
$$

$$
\begin{aligned}
&(\Gamma\alpha_1)A_3 \oplus (\Gamma\alpha_1)A_4 \oplus (\Gamma\alpha_1)A_7 \oplus (\Gamma\alpha_1)A_8 \oplus \\
&(\Gamma\alpha_1)B_3 \oplus (\Gamma\alpha_1)B_4 \oplus (\Gamma\alpha_1)B_7 \oplus (\Gamma\alpha_1)B_{12} \oplus \\
&(\Gamma\alpha_1)B_{13} \oplus (\Gamma\alpha_1)B_{14} = \\
&(\Gamma\alpha_1)z_3^H \oplus (\Gamma\alpha_1)z_3^L \oplus (\Gamma\alpha_1)z_4^H \oplus (\Gamma\alpha_1)z_4^L
\end{aligned} \tag{37}
$$

$$\Gamma A_4 \oplus \Gamma A_5 \oplus \Gamma A_8 \oplus \Gamma A_9 \oplus$$
$$\Gamma B_4 \oplus \Gamma B_5 \oplus \Gamma B_8 \oplus \Gamma B_{13} \oplus \Gamma B_{14} \oplus \Gamma B_{15} =$$
$$\Gamma z_4^H \oplus \Gamma z_4^L \oplus \Gamma z_5^H \oplus \Gamma z_5^L \tag{38}$$

$$(\Gamma\alpha_1)A_5 \oplus (\Gamma\alpha_1)A_6 \oplus (\Gamma\alpha_1)A_9 \oplus (\Gamma\alpha_1)A_{10} \oplus$$
$$(\Gamma\alpha_1)B_5 \oplus (\Gamma\alpha_1)B_6 \oplus (\Gamma\alpha_1)B_9 \oplus (\Gamma\alpha_1)B_{14} \oplus$$
$$(\Gamma\alpha_1)B_{15} \oplus (\Gamma\alpha_1)B_{16} =$$
$$(\Gamma\alpha_1)z_5^H \oplus (\Gamma\alpha_1)z_5^L \oplus (\Gamma\alpha_1)z_6^H \oplus (\Gamma\alpha_1)z_6^L \tag{39}$$

$$\Gamma A_6 \oplus \Gamma A_7 \oplus \Gamma A_{10} \oplus \Gamma A_{11} \oplus$$
$$\Gamma B_6 \oplus \Gamma B_7 \oplus \Gamma B_{10} \oplus \Gamma B_{15} \oplus \Gamma B_{16} \oplus \Gamma B_{17} =$$
$$\Gamma z_6^H \oplus \Gamma z_6^L \oplus \Gamma z_7^H \oplus \Gamma z_7^L \tag{40}$$

$$(\Gamma\alpha_0)A_6 \oplus (\Gamma\alpha_0)A_7 \oplus (\Gamma\alpha_0)A_{10} \oplus (\Gamma\alpha_0)A_{11} \oplus$$
$$(\Gamma\alpha_0)B_6 \oplus (\Gamma\alpha_0)B_7 \oplus (\Gamma\alpha_0)B_{10} \oplus (\Gamma\alpha_0)B_{15} \oplus$$
$$(\Gamma\alpha_0)B_{16} \oplus (\Gamma\alpha_0)B_{17} =$$
$$(\Gamma\alpha_0)z_6^H \oplus (\Gamma\alpha_0)z_6^L \oplus (\Gamma\alpha_0)z_7^H \oplus (\Gamma\alpha_0)z_7^L \tag{41}$$

$$(\Gamma\alpha_0)A_8 \oplus (\Gamma\alpha_0)A_9 \oplus (\Gamma\alpha_0)A_{12} \oplus (\Gamma\alpha_0)A_{13} \oplus$$
$$(\Gamma\alpha_0)B_8 \oplus (\Gamma\alpha_0)B_9 \oplus (\Gamma\alpha_0)B_{12} \oplus (\Gamma\alpha_0)B_{17} \oplus$$
$$(\Gamma\alpha_0)B_{18} \oplus (\Gamma\alpha_0)B_{19} =$$
$$(\Gamma\alpha_0)z_8^H \oplus (\Gamma\alpha_0)z_8^L \oplus (\Gamma\alpha_0)z_9^H \oplus (\Gamma\alpha_0)z_9^L \tag{42}$$

$$\Gamma A_9 \oplus \Gamma A_{10} \oplus \Gamma A_{13} \oplus \Gamma A_{14} \oplus$$
$$\Gamma B_9 \oplus \Gamma B_{10} \oplus \Gamma B_{13} \oplus \Gamma B_{18} \oplus \Gamma B_{19} \oplus \Gamma B_{20} =$$
$$\Gamma z_9^H \oplus \Gamma z_9^L \oplus \Gamma z_{10}^H \oplus \Gamma z_{10}^L \tag{43}$$

$$(\Gamma\alpha_0)A_{11} \oplus (\Gamma\alpha_0)A_{12} \oplus (\Gamma\alpha_0)A_{15} \oplus (\Gamma\alpha_0)A_{16} \oplus$$
$$(\Gamma\alpha_0)B_{11} \oplus (\Gamma\alpha_0)B_{12} \oplus (\Gamma\alpha_0)B_{15} \oplus (\Gamma\alpha_0)B_{20} \oplus$$
$$(\Gamma\alpha_0)B_{21} \oplus (\Gamma\alpha_0)B_{22} =$$
$$(\Gamma\alpha_0)z_{11}^H \oplus (\Gamma\alpha_0)z_{11}^L \oplus (\Gamma\alpha_0)z_{12}^H \oplus (\Gamma\alpha_0)z_{12}^L \tag{44}$$

$$\Gamma A_{13} \oplus \Gamma A_{14} \oplus \Gamma A_{17} \oplus \Gamma A_{18} \oplus$$
$$\Gamma B_{13} \oplus \Gamma B_{14} \oplus \Gamma B_{17} \oplus \Gamma B_{22} \oplus \Gamma B_{23} \oplus \Gamma B_{24} =$$
$$\Gamma z_{13}^H \oplus \Gamma z_{13}^L \oplus \Gamma z_{14}^H \oplus \Gamma z_{14}^L \tag{45}$$

$$\Gamma A_{14} \oplus \Gamma A_{15} \oplus \Gamma A_{18} \oplus \Gamma A_{19} \oplus$$
$$\Gamma B_{14} \oplus \Gamma B_{15} \oplus \Gamma B_{18} \oplus \Gamma B_{23} \oplus \Gamma B_{24} \oplus \Gamma B_{25} =$$
$$\Gamma z_{14}^{H} \oplus \Gamma z_{14}^{L} \oplus \Gamma z_{15}^{H} \oplus \Gamma z_{15}^{L} \tag{46}$$

$$\Gamma A_{16} \oplus \Gamma A_{17} \oplus \Gamma A_{20} \oplus \Gamma A_{21} \oplus$$
$$\Gamma B_{16} \oplus \Gamma B_{17} \oplus \Gamma B_{20} \oplus \Gamma B_{25} \oplus \Gamma B_{26} \oplus \Gamma B_{27} =$$
$$\Gamma z_{16}^{H} \oplus \Gamma z_{16}^{L} \oplus \Gamma z_{17}^{H} \oplus \Gamma z_{17}^{L} \tag{47}$$