Correlation of Quadratic Boolean Functions: Cryptanalysis of All Versions of Full MORUS

Danping Shi[1,2], Siwei Sun[1,2,3⋆], Yu Sasaki[4], Chaoyun Li[5], and Lei Hu[1,2,3]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, China
[2] Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, China
[3] School of Cyber Security, University of Chinese Academy of Sciences, China
{shidanping, sunsiwei, hulei}@iie.ac.cn
[4] NTT Secure Platform Laboratories, Japan sasaki.yu@lab.ntt.co.jp
[5] imec-COSIC, Dept. Electrical Engineering (ESAT), KU Leuven, Belgium chaoyun.li@esat.kuleuven.be

Abstract. We show that the correlation of any quadratic Boolean function can be read out from its so-called *disjoint quadratic form*. We further propose a polynomial-time algorithm that can transform an arbitrary quadratic Boolean function into its disjoint quadratic form. With this algorithm, the exact correlation of quadratic Boolean functions can be computed efficiently.
We apply this method to analyze the linear trails of MORUS (one of the seven finalists of the CAESAR competition), which are found with the help of a generic model for linear trails of MORUS-like key-stream generators. In our model, any tool for finding linear trails of block ciphers can be used to search for trails of MORUS-like key-stream generators. As a result, a set of trails with correlation 2^{-38} is identified for all versions of full MORUS, while the correlations of previously published best trails for MORUS-640 and MORUS-1280 are 2^{-73} and 2^{-76} respectively (ASIACRYPT 2018). This significantly improves the complexity of the attack on MORUS-1280-256 from 2^{152} to 2^{76}. These new trails also lead to the first distinguishing and message-recovery attacks on MORUS-640-128 and MORUS-1280-128 with surprisingly low complexities around 2^{76}.
Moreover, we observe that the condition for exploiting these trails in an attack can be more relaxed than previously thought, which shows that the new trails are superior to previously published ones in terms of both correlation and the number of ciphertext blocks involved.

Keywords: Quadratic Boolean function · Disjoint quadratic form · Correlation attack · CAESAR competition · MORUS · MILP

1 Introduction

The notion of authenticated encryption (AE), which provides both confidentiality and authenticity, was first introduced by Bellare and Namprempre around

⋆ The corresponding author

2000 [4,5]. It was further developed and evolved into the notion of authenticated encryption with associated data (AEAD) [26,27,28] to capture the settings of real-world communication networks, where the authenticity of some public information (e.g., packet header) must be ensured. Informally, an AEAD is a secret-key scheme involving an encryption algorithm and a decryption algorithm. Its encryption algorithm receives a plaintext or message M, an associated data A, and a secret key K, and produces a ciphertext C and a tag T. The authenticity of the message and associated data can be checked against the tag T. We refer the reader to [26] for a more rigorous treatment of the definition of AEAD.

The CAESAR competition (the Competition for Authenticated Encryption: Security, Applicability, and Robustness) was announced at the Early Symmetric-key Crypto workshop 2013 [13] and also on-line at [7]. After several years of intensive analysis and comparison of the 57 submissions, the finalists were announced at FSE 2018. In this work, our target is one of the seven finalists — MORUS [36], which provides three main variants: MORUS-640 with a 128-bit key, and MORUS-1280 with either a 128-bit or a 256-bit key.

Related Work. Apart from the analysis provided by the designers, MORUS has received extensive third-party cryptanalysis. These cryptanalysis include differential cryptanalysis [24,30,12], linear cryptanalysis [19], SAT-based cryptanalysis [11], cube cryptanalysis [29,19], state-recovery [17,35] and key-recovery attacks [12], as well as attacks in the nonce-reuse setting [24]. However, these attacks either target round-reduced versions of MORUS, or are launched in the nonce-reuse setting which is contradicting to the nonce-respect assumption assumed by the designers. Therefore, none of these analysis violates the security claims of MORUS.

A major breakthrough on the cryptanalysis of MORUS was made at ASIACRYPT 2018 [2]. In this work, based on rotational-invariant linear approximations, Ashur et al. transfered linear approximations for a state-reduced version of MORUS (named as MiniMORUS) to linear approximations for MORUS. Linear approximations in the ciphertext bits with correlation 2^{-73} and 2^{-76} were identified for MORUS-640 and MORUS-1280 respectively. The approximation of MORUS-1280 leads to distinguishing attacks and message-recovery attacks on the full MORUS-1280 with 256-bit key. Since it requires about $2^{2\times 76} = 2^{152}$ encryptions to exploit the correlation, MORUS-1280 with 128-bit keys remain immune to these attacks. Similarly, to exploit the correlation of MORUS-640, it requires about 2^{146} encryptions, which means MORUS-640-128 is also immune to these attacks.

Our Contribution. In this work, we investigate the problem of computing the correlation of quadratic Boolean functions. By transforming a quadratic Boolean function into its so-called *disjoint quadratic form*, we propose, to the best of our knowledge, the first *polynomial time* algorithm that can determine the correlation of an arbitrary quadratic Boolean function, while in previous

work (e.g., [2]), such correlations are computed with exhaustive or quite ad-hoc approaches which intrinsically limits their effectiveness.

Equipped with this new weapon, we set out to search for more complex rotational invariant linear trails of MORUS, and then compute their correlations with the new method. To this end, we set up a model for finding linear trails of MORUS-like key-stream generators, such that most existing search tools can be applied. The model we proposed is generic and can be applied to many other schemes, which is of independent interest. Eventually, using MILP based approach, we identify trails of all versions of MORUS which lead to significant improvement over the previous attack on MORUS-1280-256 presented by Ashur et al. [2]. Generally, the complexity is reduced from 2^{152} to 2^{76}. Moreover, these trails result in the first attacks on full MORUS-640 and MORUS-1280 with 128-bit key. A summary of the results are given in Table 1, from which we can see that the attack is not marginal and the complexities are approaching the boundary of practical attacks. We verify the attacks on a reduced version of MORUS. Also, following Ashur et al.'s approach [2], we verify all trail fragments for all versions of full MORUS.

Along the way, we make an interesting observation that the condition imposed on Ashur et al.'s attack can be relaxed. Specifically, the attacks actually only require that enough plaintexts with a common prefix of certain size are encrypted, rather than the same plaintext is encrypted enough times as stated in [2]. This observation motivates us to find trails involving a smaller number of ciphertext blocks, since the common-prefix assumption does occur in some practical protocols.

At this point, we would like to mention that even after Ashur et al.'s work [2], many researchers are not sure if MORUS will stay in the competition given the high complexities of the attacks and the status of MORUS-640-128 and MORUS-1280-128. However, we think that the new attacks breaking all versions of full MORUS with complexity around 2^{76} severely shake the security confidence of MORUS and should deserve more attentions. Finally, our technique is purely linear, and most of the attacks presented in our paper are known-plaintext attacks, where we do not rely on any property of the output of the initialization process except its randomness. Hence, it is interesting to see how to improve our analysis by applying the differential-linear framework [18,3].

The exact linear trails we used can be found in an extended version of the paper at https://eprint.iacr.org/2019/172, and the source code is available at https://github.com/siweisun/attack_morus.

Organization. In Sect. 2, we give a brief visualized description of the authenticated encryption scheme MORUS. Then in Sect. 3, we show how to compute the correlation of a quadratic Boolean function by transforming it into the so-called disjoint quadratic form. A generic model for finding linear trails of MORUS-like key-stream generators is constructed in Sect. 4, which is employed in Sect. 5 to search for linear trails of MORUS with high absolute correlations, leading to attacks on all versions of full MORUS. Section 6 discusses the condition of the at-

Table 1: A summary of the results, where the span indicates the number of ciphertext blocks involved in the linear approximations.

Target	Linear masks of the ciphertext blocks	Span	|Correlation|	Data	Time	Source
MiniMORUS-640	08000000 04000105 8800a002 00105040 00080000	5	2^{-16}	2^{32}	2^{32}	[2]
	10000000 08000202 00004103 00002000	4	2^{-8}	2^{16}	2^{16}	Sect. 5
MiniMORUS-1280	0008000000000000 0080000202000001 0008406020000090 0004040000100800 0000000001000000	5	2^{-16}	2^{32}	2^{32}	[2]
	0000000000010000 4000000020100000 0000000220000804 0000000000008000	4	2^{-8}	2^{16}	2^{16}	Sect. 5
MORUS-640-128	10000000 10000000 10000000 10000000 08000202 08000202 08000202 08000202 00004103 00004103 00004103 00004103 00002000 00002000 00002000 00002000	4	2^{-38}	2^{76}	2^{76}	Sect. 5
MORUS-1280-128	0000000000010000 0000000000010000 0000000000010000 0000000000010000 4000000020100000 4000000020100000 4000000020100000 4000000020100000 0000000220000804 0000000220000804 0000000220000804 0000000220000804 0000000000008000 0000000000008000 0000000000008000 0000000000008000	4	2^{-38}	2^{76}	2^{76}	Sect. 5
MORUS-1280-256	0008000000000000 0008000000000000 0008000000000000 0008000000000000 0080000202000001 0080000202000001 0080000202000001 0080000202000001 0008406020000090 0008406020000090 0008406020000090 0008406020000090 0004040000100800 0004040000100800 0004040000100800 0004040000100800 0000000001000000 0000000001000000 0000000001000000 0000000001000000	5	2^{-76}	2^{152}	2^{152}	[2]
	0000000000010000 0000000000010000 0000000000010000 0000000000010000 4000000020100000 4000000020100000 4000000020100000 4000000020100000 0000000220000804 0000000220000804 0000000220000804 0000000220000804 0000000000008000 0000000000008000 0000000000008000 0000000000008000	4	2^{-38}	2^{76}	2^{76}	Sect. 5

tacks presented in the previous section and clarifies why trails involving a smaller number of ciphertext blocks are preferred. We propose some open problems and conclude in Sect. 7.

2 Specification of MORUS and MiniMORUS

We give a brief description of MORUS and MiniMORUS, which largely follows the notations used by Ashur et al. [2] to facilitate cross checking.

2.1 MORUS

MORUS is a family of AEAD schemes [36] whose interfaces are shown in Fig. 1. The encryption algorithm of MORUS operates on a $5q$-bit state composed of five q-bit registers ($q \in \{128, 256\}$), and each register is divided into four $q/4$-bit words as shown in Fig. 2, where we use $S_{i,j}$ to denote the jth bit of the ith register S_i of the $5q$-bit state S. The three recommended parameter sets of MORUS are listed in Table 2. Note that when the exact key size is not important, we use MORUS-640 and MORUS-1280 to denote the versions with 640-bit state and 1280-bit state, respectively.

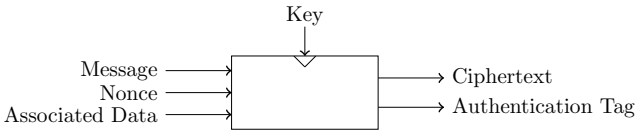


Fig. 1: The high-level structure of the encryption algorithm of an AEAD scheme

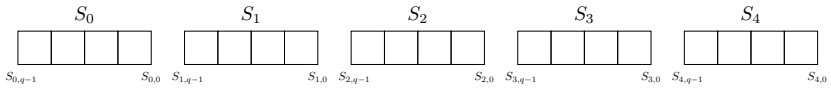


Fig. 2: A view of the MORUS internal state

During the encryption process of MORUS, a function

$$\texttt{StateUpdate} : \mathbb{F}_2^{5q} \times \mathbb{F}_2^{q} \to \mathbb{F}_2^{5q}$$

is repeatedly executed on the internal state. Each call to the `StateUpdate` function is called a step. We denote the state at the very beginning of the encryption process by $S^{-16} = S_0^{-16} \parallel S_1^{-16} \parallel S_2^{-16} \parallel S_3^{-16} \parallel S_4^{-16}$. After a series of steps, a sequence of states is produced:

$$S^{-16} \xrightarrow{\texttt{StateUpdate}} S^{-15} \xrightarrow{\texttt{StateUpdate}} \cdots \xrightarrow{\texttt{StateUpdate}} S^{0} \xrightarrow{\texttt{StateUpdate}} \cdots$$

Table 2: The three variants of MORUS, where the sizes are measured in bits

Name	State size ($5q$)	Register size (q)	Word size ($q/4$)	Key size	Tag size
MORUS-640-128	640	128	32	128	128
MORUS-1280-128	1280	256	64	128	128
MORUS-1280-256	1280	256	64	256	128

Therefore, we can use the notion $S^t = S_0^t \parallel S_1^t \parallel S_2^t \parallel S_3^t \parallel S_4^t$ to reference the state at step t. The detail of the `StateUpdate` function is shown in the following equations:

$$
\begin{aligned}
S_0^{t+1} &\leftarrow (S_0^t \oplus (S_1^t \cdot S_2^t) \oplus S_3^t) \lll_w b_0, & S_3^t &\leftarrow S_3^t \lll b_0^{'},\\
S_1^{t+1} &\leftarrow (S_1^t \oplus (S_2^t \cdot S_3^t) \oplus S_4^t \oplus m_i) \lll_w b_1, & S_4^t &\leftarrow S_4^t \lll b_1^{'},\\
S_2^{t+1} &\leftarrow (S_2^t \oplus (S_3^t \cdot S_4^t) \oplus S_0^t \oplus m_i) \lll_w b_2, & S_0^t &\leftarrow S_0^t \lll b_2^{'},\\
S_3^{t+1} &\leftarrow (S_3^t \oplus (S_4^t \cdot S_0^t) \oplus S_1^t \oplus m_i) \lll_w b_3, & S_1^t &\leftarrow S_1^t \lll b_3^{'},\\
S_4^{t+1} &\leftarrow (S_4^t \oplus (S_0^t \cdot S_1^t) \oplus S_2^t \oplus m_i) \lll_w b_4, & S_2^t &\leftarrow S_2^t \lll b_4^{'},
\end{aligned}
$$

where $\lll_\omega b_i$ means rotation inside every w-bit ($w = q/4$) word of the register to the left by b_i bits, and $\lll$ is the ordinary left bitwise rotation operation. The concrete values for the rotation offsets are listed in Table 3, and we refer the readers to Fig. 3 for a visualization of the `StateUpdate` function.

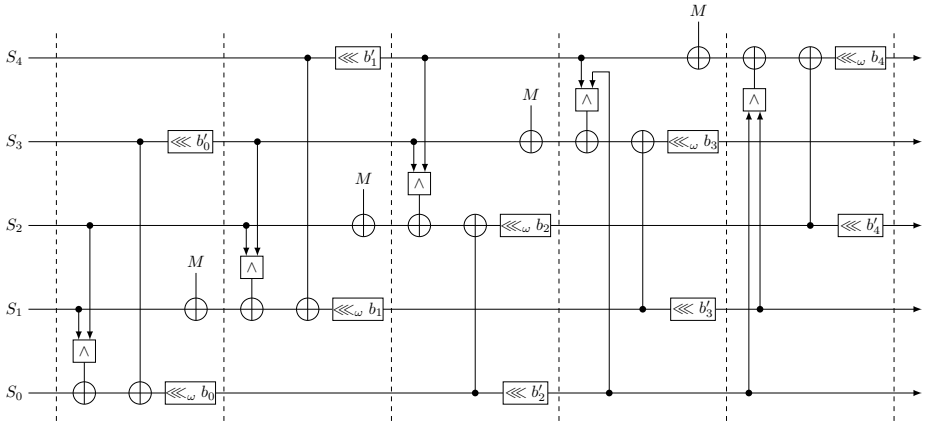


Fig. 3: The `StateUpdate` function of MORUS

The encryption algorithm of MORUS can be divided into four phases. A visualized description of the encryption algorithm of MORUS without the finalization phase can be found in Fig. 4.

Table 3: Rotation constants b_i for $\lll_w$ and b'_i for $\lll$ in round i of `StepUpdate`

Cipher	Rotation offsets for $\lll_w$					Rotation offsets for $\lll$				
	b_0	b_1	b_2	b_3	b_4	b'_0	b'_1	b'_2	b'_3	b'_4
MORUS-640-128	5	31	7	22	13	32	64	96	64	32
MORUS-1280-128	13	46	38	7	4	64	128	192	128	64
MORUS-1280-256	13	46	38	7	4	64	128	192	128	64

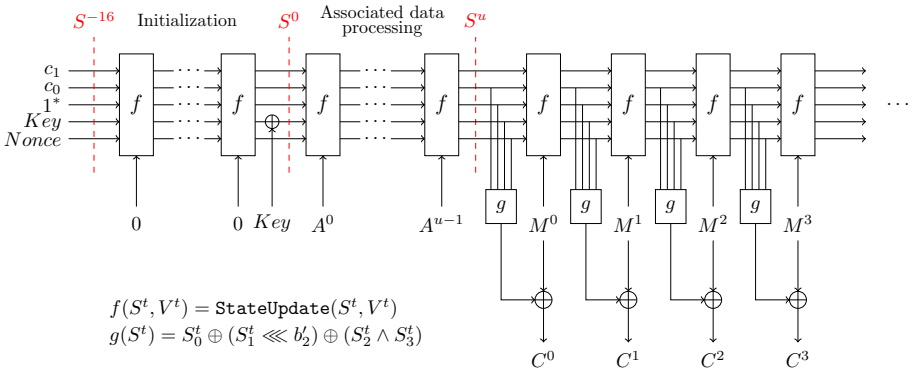


Fig. 4: The encryption algorithm of MORUS

Initialization. The initialization of every MORUS instance starts by loading the key and nonce materials into the state to produce the starting state S^{-16}. Then update the state by calling `StateUpdate` 16 times, and finally the key is exclusive-ored into the state to produce the resulting state S^0. Let c_0 and c_1 be two 128-bit constants, and we use N_{128}, K_{128}, and K_{256} to denote the 128-bit nonce, 128-bit key and 256-bit key, respectively. The details of the initialization processes for different versions of MORUS are given in the following.

MORUS-640-128: $S^{-16} = N_{128} \parallel K_{128} \parallel 1^{128} \parallel c_0 \parallel c_1$. Then for $t = -16, -15, \cdots, -1$, $S^{t+1} = \texttt{StateUpdate}(S^t, 0^{128})$. Finally, we set $S^0 \leftarrow S^0_0 \parallel S^0_1 \oplus K_{128} \parallel S^0_2 \parallel S^0_3 \parallel S^0_4$.

MORUS-1280-128: $S^{-16} = (N_{128} \parallel 0^{128}) \parallel (K_{128} \parallel K_{128}) \parallel 1^{256} \parallel 0^{256} \parallel (c_0 \parallel c_1)$. Then for $t = -16, -15, \cdots, -1$, $S^{t+1} = \texttt{StateUpdate}(S^t, 0^{256})$. Finally, we set $S^0 \leftarrow S^0_0 \parallel S^0_1 \oplus (K_{128} \parallel K_{128}) \parallel S^0_2 \parallel S^0_3 \parallel S^0_4$.

MORUS-1280-256: $S^{-16} = (N_{128} \parallel 0^{128}) \parallel K_{256} \parallel 1^{256} \parallel 0^{256} \parallel (c_0 \parallel c_1)$. Then for $t = -16, -15, \cdots, -1$, $S^{t+1} = \texttt{StateUpdate}(S^t, 0^{256})$. Finally, we set $S^0 \leftarrow S^0_0 \parallel S^0_1 \oplus K_{256} \parallel S^0_2 \parallel S^0_3 \parallel S^0_4$.

Associated Data Processing. If there is no associated data, this process is omitted. Otherwise, the associated data is padded with zeros when necessary to form a multiple of q-bit (register size) block. Then the state is updated with the

associated data A as $S^{t+1} = \texttt{StateUpdate}(S^t, A^t)$, for $t = 0, \cdots, u-1$, where $u = \lceil |A|/q \rceil$ is the number of q-bit blocks of the (padded) associated data A.

Encryption. The plaintext is processed in q-bit blocks to update the state and generate the ciphertext block at the same time. Similar to associated data processing, the plaintext is padded with zeros if the last block is fractional. For $t = 0, \cdots, v-1$, the following is performed.

$$\begin{aligned} C^t &= M^t \oplus S_0^{u+t} \oplus (S_1^{u+t} \lll b_2') \oplus (S_2^{u+t} \wedge S_3^{u+t}), \\ S^{u+t+1} &= \texttt{StateUpdate}(S^{u+t}, M^t), \end{aligned}$$

where $v = \lceil |M|/q \rceil$ is the number of q-bit blocks of the padded plaintext.

Finalization. The authentication tag T is generated in the finalization phase by calling `StateUpdate` ten more times. Since our attacks are completely irrelevant to how the tag is generated, we omit its details.

2.2 MiniMORUS and Rotational Invariance

MiniMORUS, proposed by Ashur et al. [2], is a family of helper constructions derived from MORUS. For every MORUS instance with a $5q$-bit state, there is a MiniMORUS instance with $5 \cdot (q/4)$-bit state. To be more specific, each register in MiniMORUS contains a single word of $w = q/4$ bits. Therefore, the word-oriented rotations in the `StateUpdate` function of MORUS are removed in MiniMORUS, and the rotations within words ($\lll_\omega b_i$) are equivalent to ordinary bit-wise rotations ($\lll b_i$) in MiniMORUS. We refer the reader to Fig. 5 and Fig. 3 for a comparison.

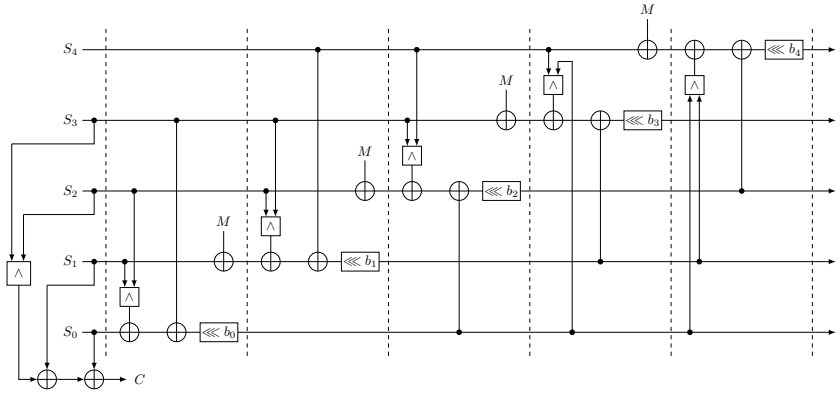


Fig. 5: The `StateUpdate` function of MiniMORUS.

Obviously, MiniMORUS can be regarded as a reduced version of MORUS. Therefore, it is easier to search for linear trails of MiniMORUS. When a linear trail of MiniMORUS is identified, we can consider the trail for MORUS where the bits involved in every $q/4$-bit register of MiniMORUS are copied into all the four $q/4$-bit words in the corresponding register of MORUS. To put it simply, we only consider trails of MORUS involving the same bits within each word of one register. This kind of patterns are invariant under word-wise rotations. Therefore, the trails for MiniMORUS can be regarded as truncated representations of the trails for MORUS with rotational invariant patterns. We refer the reader to [2] for more details.

3 Correlation of Quadratic Boolean Functions

In this section, we give a brief introduction of necessary background of Boolean functions, prove that the correlation of a quadratic Boolean function can be read out from its disjoint quadratic form, and show how to convert an arbitrary quadratic Boolean function into its so-call *disjoint quadratic term* with a polynomial time algorithm.

Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function with algebraic normal form (ANF)

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{u} \in \mathbb{F}_2^n} a_{\boldsymbol{u}} \boldsymbol{x}^{\boldsymbol{u}},$$

where $\boldsymbol{x} = (x_1, \cdots, x_n), \boldsymbol{u} = (u_1, \cdots, u_n), a_{\boldsymbol{u}} \in \mathbb{F}_2$, and $\boldsymbol{x}^{\boldsymbol{u}} = \prod_{i=1}^n x_i^{u_i}$. The degree of the Boolean function f is defined as

$$\deg(f) = \max_{\boldsymbol{u} \in \mathbb{F}_2^n : a_{\boldsymbol{u}} \neq 0} wt(\boldsymbol{u}),$$

where $wt(\boldsymbol{u})$ is the Hamming weight of $\boldsymbol{u}$.

Definition 1 (Correlation). *The correlation of an n-variable Boolean function f is $\mathrm{cor}(f) = \frac{1}{2^n} \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{f(\boldsymbol{x})}$, and the weight of the correlation is defined as $-\log_2 |\mathrm{cor}(f)|$.*

In the following, we use $\mathrm{Var}(f)$ to denote the set of variables involved in the Boolean function f. For example, if $h = x_1x_2 + x_1x_3 + 1$ and $g = x_2x_3x_4 + x_3x_4$, then $\mathrm{Var}(h) = \{x_1, x_2, x_3\}$ and $\mathrm{Var}(g) = \{x_2, x_3, x_4\}$. Note that the variables are treated as symbolic objects. A variable x_i is *degenerate* if it does not appear in the ANF of f, i.e., $x_i \notin \mathrm{Var}(f)$. For example, if $f(x_1, x_2, x_3, x_4, x_5) = x_1 + x_2x_3 + x_4$, then x_5 is degenerate.

Lemma 1. *Let $g(x_1, \cdots, x_n) = \sum_{t=1}^k f_t$ be a Boolean function such that the k sets $\mathrm{Var}(f_t)$ for $1 \leq t \leq k$ are mutually disjoint. Then $\mathrm{cor}(g) = \prod_{t=1}^k \mathrm{cor}(f_t)$.*

Proof. Let f_t be a Boolean function with n_t variables for $1 \le t \le k$, and $m = n - n_1 - \cdots - n_k$. According to Definition 1, we have

$$\begin{aligned}\mathrm{cor}(g) &= \frac{1}{2^n}\sum_{\boldsymbol{x}\in\mathbb{F}_2^n}(-1)^{g(\boldsymbol{x})} = \sum_{\boldsymbol{x}\in\mathbb{F}_2^n}\frac{(-1)^{(f_1+f_2+\cdots+f_k)(\boldsymbol{x})}}{2^n}\\ &= \sum_{\boldsymbol{x}_1\in\mathbb{F}_2^{n_1}}\frac{(-1)^{f_1(\boldsymbol{x}_1)}}{2^{n_1}}\cdots\sum_{\boldsymbol{x}_k\in\mathbb{F}_2^{n_k}}\frac{(-1)^{f_k(\boldsymbol{x}_1)}}{2^{n_k}}\cdot\sum_{\boldsymbol{x}\in\mathbb{F}_2^{m}}\frac{(-1)^{0}}{2^{m}}\\ &= \prod_{t=1}^{k}\mathrm{cor}(f_t),\end{aligned}$$

as desired. □

Example 1. $\mathrm{cor}(x_1x_2 + x_3x_4) = \mathrm{cor}(x_1x_2)\cdot\mathrm{cor}(x_3x_4) = 2^{-2}$.

Corollary 1. *Let $f(x_1,\cdots,x_n)$ be a Boolean function, and $f = g + x_j$ such that $x_j \notin \mathrm{Var}(g)$ is a separated linear term. Then* $\mathrm{cor}(f) = 0$.

Example 2. $\mathrm{cor}(x_1x_2+x_2x_3x_4+x_3x_5+x_6) = \mathrm{cor}(x_1x_2+x_2x_3x_4+x_3x_5)\cdot\mathrm{cor}(x_6) = \mathrm{cor}(x_1x_2 + x_2x_3x_4 + x_3x_5)\cdot 0 = 0$.

Lemma 2. *Let $f(x,y) = xy + ax + by$ be a Boolean function and $a, b \in \mathbb{F}_2$ are constants. Then* $\mathrm{cor}(f) = (-1)^{ab}\cdot 2^{-1}$.

Proof. Prove by exhaustive analysis of a and b with Definition 1. □

Definition 2. *Two Boolean functions $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are called cogredient if there exists an invertible matrix M, such that $g(\boldsymbol{x}) = f(\boldsymbol{x}M)$.*

Lemma 3. *Let $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ be two Boolean functions cogredient to each other. Then* $\mathrm{cor}(f) = \mathrm{cor}(g)$.

Proof. Since $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are cogredient to each other, $g(\boldsymbol{x}) = f(\boldsymbol{x}M)$ for some invertible matrix M. The result follows from the following equation

$$\begin{aligned}\mathrm{cor}(g) &= \tfrac{1}{2^n}\sum_{\boldsymbol{x}\in\mathbb{F}_2^n}(-1)^{g(\boldsymbol{x})} = \tfrac{1}{2^n}\sum_{\boldsymbol{x}\in\mathbb{F}_2^n}(-1)^{f(\boldsymbol{x}M)}\\ &= \tfrac{1}{2^n}\sum_{\boldsymbol{x}M^{-1}\in\mathbb{F}_2^n}(-1)^{f(\boldsymbol{x})} = \tfrac{1}{2^n}\sum_{\boldsymbol{x}\in\mathbb{F}_2^n}(-1)^{f(\boldsymbol{x})}.\end{aligned}$$

□

Lemma 3 implies that the correlation of a Boolean function is invariant by applying an invertible linear transformation to the input variables. Also, it is sufficient to consider functions with constant term 0 since $\mathrm{cor}(f) = -\mathrm{cor}(f+1)$ for any f.

Definition 3 (Quadratic form). *A Boolean function f is* quadratic *if* $\deg(f) = 2$. *A quadratic Boolean function is called a quadratic form if its constant term is* 0. *Hence, a quadratic form can be written as*

$$f(x_1, \cdots, x_n) = \sum_{1 \le i \le j \le n} a_{i,j} x_i x_j = Q_f(x_1, \cdots, x_n) + L_f(x_1, \cdots, x_n)$$

where $a_{i,j} \in \mathbb{F}_2$, Q_f contains all quadratic terms of f while L_f consists of all linear terms of f.

Let $f(x_1, \cdots, x_n)$ be a quadratic Boolean function. For $i \in \{1, \cdots, n\}$, we use $\sigma(f, x_i)$ to denote the number of terms of Q_f involving variable x_i.

Definition 4 (Disjoint quadratic form). *Let $f(x_1, \cdots, x_n)$ be a quadratic form. A term $x_i x_j$ of f is a separated quadratic term if $\sigma(f, x_i) = \sigma(f, x_j) = 1$. In particular, f is disjoint if all its quadratic terms are separated quadratic terms.*

Example 3. The two functions $x_1x_2 + x_3x_4$ and $x_1x_3 + x_2x_4 + x_2 + x_5$ are both disjoint quadratic forms, while $x_1x_2 + x_2x_3$ is not a disjoint quadratic form.

Lemma 4. *Let $f = x_{i_1}x_{i_2} + \cdots + x_{i_{2k-1}}x_{i_{2k}} + x_{j_1} + \cdots + x_{j_s}$ be a disjoint quadratic form. Then*

$$\mathrm{cor}(f) = \begin{cases} (-1)^{\sum_{t=1}^{k} \mathrm{Coe}_f(x_{i_{2t-1}})\mathrm{Coe}_f(x_{i_{2t}})} \cdot 2^{-k} & \{j_1, \cdots, j_s\} \subseteq \{i_1, \cdots, i_{2k}\} \\ 0 & \{j_1, \cdots, j_s\} \subsetneq \{i_1, \cdots, i_{2k}\} \end{cases}$$

where $\mathrm{Coe}_f(\boldsymbol{x}^{\boldsymbol{u}})$ denotes the coefficient of the monomial $\boldsymbol{x}^{\boldsymbol{u}}$ in the ANF of f.

Proof. It follows from Lemma 1, Corollary 1, and Lemma 2. □

With Lemma 4, it is easy to obtain the correlation of a disjoint quadratic form. In the remainder of this section, we will present an efficient algorithm for converting any given quadratic form to a cogredient disjoint quadratic form. Hence, we can efficiently compute the correlation of any given quadratic form. Before diving into the details of the algorithm, we first introduce some useful notations and subroutines employed in Algorithm 1.

Subroutine 1 (`PickIndex`). *Given a quadratic Boolean function $f(\boldsymbol{x})$ with $\boldsymbol{x} = (x_1, \cdots, x_n)$,* `PickIndex`$(f)$ *returns the index t of x_t, where t is the smallest integer $t \in \{1, \cdots, n\}$, such that $\sigma(f, x_t) \ge \sigma(f, x_{t'})$ for all $t' \in \{1, \cdots, n\}$.*

Example 4. Let $n = 3$, $f(\boldsymbol{x}) = x_1x_2 + x_2x_3 + x_3$. Then `PickIndex`$(f) = 2$.

Subroutine 2 (`Substitute`). *Given a Boolean function $f(\boldsymbol{x}) = f(x_1, \cdots, x_n)$ and an $n \times n$ invertible matrix M,* `Substitute`(f, M) *returns the Boolean function $f(\boldsymbol{x}M)$.*

Example 5. Let $f = x_1x_2 + x_2x_3 + x_3$, and $M = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$. Then `Substitute`$(f, M)$ gives $f(\boldsymbol{x}M) = (x_1 + x_2)(x_2 + x_3) + (x_2 + x_3)x_3 + x_3 = x_1x_2 + x_1x_3 + x_2$.

In Algorithm 1, for a given Boolean function $f(x_1, \cdots, x_n)$, we repeatedly use a substitution of variables of the form:

$$\begin{cases} x_u \leftarrow x_{t_1} + x_{t_2} + \cdots + x_{t_m} \\ x_j \leftarrow x_j, \quad \forall j \in \{1, \cdots, n\} - \{u\} \end{cases},$$

where $m \geq 2$, $u \in \{t_1, \cdots, t_m\}$, and $t_1 < t_2 < \cdots < t_m$. This substitution can be reformulated in the matrix form as $\boldsymbol{x} \leftarrow \boldsymbol{x} I_{u \leftarrow t_1, \cdots, t_m}$, where $I_{u \leftarrow t_1, \cdots, t_m}$ is obtained from the $n \times n$ identity matrix I by substituting the u-th column with a column vector whose t_j-th entry is 1 for $1 \leq j \leq m$ and other entries are 0. Note that we always have $I_{u \leftarrow t_1, \cdots, t_m} = I^{-1}_{u \leftarrow t_1, \cdots, t_m}$.

Algorithm 1: Transform to disjoint quadratic form

Input: A quadratic form $f(\boldsymbol{x}) = f(x_1, \cdots, x_n)$
Output: An invertible matrix M and a disjoint quadratic form $\hat{f}(\boldsymbol{x})$ such that $\hat{f}(\boldsymbol{x}) = f(\boldsymbol{x}M)$

```
/* Initialization */
M ← I                                   /* I is the n × n identity matrix */
f̂(x) ← f(x_1, ··· , x_n)
v ← PickIndex(f̂)
/* Transformation */
while σ(f̂, x_v) ≥ 2 do
    m ← σ(f̂, x_v)        /* The number of quadratic terms involving x_v */
    Find all t_1 < t_2 < ··· < t_m, such that x_v x_{t_i} is a term of f̂.
    f̂ ← Substitute(f̂, I_{t_1 ← t_1, ··· , t_m})
    M ← I_{t_1 ← t_1, ··· , t_m} · M
    if σ(f̂, x_{t_1}) ≥ 2 then
        k ← σ(f̂, x_{t_1})
        Find all s_1 < s_2 < ··· < s_k, such that x_{t_1} x_{s_i} is a term of f̂.
        f̂ ← Substitute(f̂, I_{v ← s_1, ··· , s_k})
        M ← I_{v ← s_1, ··· , s_k} · M
    end
    v ← PickIndex(f̂)
end
return M and f̂
```

Example 6. Let $\hat{f} \leftarrow f(x_1, x_2, x_3, x_4, x_5) = x_1x_2 + x_1x_5 + x_2x_3 + x_2x_4 + x_1 + x_2$. Then $\sigma(\hat{f}, x_1) = 2$, $\sigma(\hat{f}, x_2) = 3$, $\sigma(\hat{f}, x_3) = 1$, $\sigma(\hat{f}, x_4) = 1$, and $\sigma(\hat{f}, x_5) = 1$. Thus, $v \leftarrow \texttt{PickIndex}(\hat{f}) = 2$. Now we extract the common factor $x_v = x_2$ in

$Q_{\hat{f}}$:

$$\hat{f}(\boldsymbol{x}) = x_2(x_1 + x_3 + x_4) + x_1x_5 + x_1 + x_2.$$

Then we apply the following substitution of variables:

$$\begin{cases} x_1 \leftarrow x_1 + x_3 + x_4 \\ x_j \leftarrow x_j, \quad j \in \{1, \cdots, 5\} - \{1\} \end{cases}. \tag{1}$$

This variable substitution gives $\hat{f} \leftarrow x_2x_1+(x_1+x_3+x_4)x_5+(x_1+x_3+x_4)+x_2 = x_1x_2+x_1x_5+x_3x_5+x_4x_5+x_1+x_2+x_3+x_4$. Then we need to check whether x_1 (the variable corresponding to a sum of the original variables rather than a single x_j) appears multiple times in $Q_{\hat{f}}$. Since $\sigma(\hat{f}, x_1) = 2$ (x_1 appears multiple times), we extract the common factor: $f = x_1(x_2+x_5)+x_3x_5+x_4x_5+x_1+x_2+x_3+x_4$. Then we apply the variable substitution:

$$\begin{cases} x_2 \leftarrow x_2 + x_5 \\ x_j \leftarrow x_j, \quad j \in \{1, \cdots, 5\} - \{2\} \end{cases}. \tag{2}$$

This variable substitution gives $\hat{f} \leftarrow x_1x_2 + x_3x_5 + x_4x_5 + x_1 + (x_2 + x_5) + x_3 + x_4 = x_1x_2 + x_3x_5 + x_4x_5 + x_1 + x_2 + x_3 + x_4 + x_5$. At this point (a whole **while** loop is done), we can observe that x_1x_2 is a separated quadratic term of $\hat{f}$. Actually, as shown in Theorem 1, every execution of the **while** loop will make one quadratic term separated. Then `PickIndex`$(\hat{f})$ returns 5, and we have $\hat{f} = x_1x_2 + (x_3 + x_4)x_5 + x_1 + x_2 + x_3 + x_4 + x_5$. Applying the substitution

$$\begin{cases} x_3 \leftarrow x_3 + x_4 \\ x_j \leftarrow x_j, \quad j \in \{1, \cdots, 5\} - \{3\} \end{cases}, \tag{3}$$

gives $\hat{f} = x_1x_2 + x_3x_5 + x_1 + x_2 + x_3 + x_5$, which is a disjoint quadratic form. It follows from Equations (1) – (3) that

$$M = \begin{bmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&0&1&0&0 \\ 0&0&1&1&0 \\ 0&0&0&0&1 \end{bmatrix} \cdot \begin{bmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&0&1&0&0 \\ 0&0&0&1&0 \\ 0&1&0&0&1 \end{bmatrix} \cdot \begin{bmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 1&0&1&0&0 \\ 1&0&0&1&0 \\ 0&0&0&0&1 \end{bmatrix} = \begin{bmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 1&0&1&0&0 \\ 0&0&1&1&0 \\ 0&1&0&0&1 \end{bmatrix}.$$

It is readily to verify that $\hat{f} = f(\boldsymbol{x}M)$. Consequently, according to Lemma 3, the correlation of f is $(-1)^{1\cdot1+1\cdot1} \cdot 2^{-2} = 2^{-2}$.

To show the validity of Algorithm 1, we present the following result.

Lemma 5. *For any input quadratic form $f(\boldsymbol{x}) = f(x_1, \cdots, x_n)$ of Algorithm 1, each* **while** *loop will generate at least one separated quadratic terms.*

Proof. Let $\hat{f} = x_v(x_{t_1} + x_{t_2} + \cdots + x_{t_m}) + g$, where $v = \texttt{PickIndex}(\hat{f})$ and $t_1, t_2, \cdots, t_m$ be all the indices such that $x_{t_i}x_v$ is a term of $\hat{f}$ with $t_1 < t_2 < \cdots < t_m$. Then we have $\sigma(g, x_v) = 0$ according to the way we choose t_i's. After the variable substitution $\boldsymbol{x} \leftarrow \boldsymbol{x} \cdot I_{t_1 \leftarrow t_1, \cdots, t_m}$, we have

$$\hat{f} \leftarrow x_v x_{t_1} + g(\boldsymbol{x} \cdot I_{t_1 \leftarrow t_1, \cdots, t_m}).$$

Since x_v is unchanged under $I_{t_1 \leftarrow t_1, \cdots, t_m}$, we have $\sigma(\hat{f}, x_v) = 1 + \sigma(g, x_v) = 1$.

If $\sigma(\hat{f}, x_{t_1}) = 1$, then $x_v x_{t_1}$ is a separated quadratic term. Otherwise, we have $\sigma(\hat{f}, x_{t_1}) \geq 2$. Assume that the current $\hat{f}$ can be written as $\hat{f} = x_{t_1}(x_v + x_{s_1} + \cdots + x_{s_k}) + h$, where $s_1, s_2, \cdots, s_k$ are all the indices such that $x_{t_1}x_{s_i}$ is a term of $\hat{f}$ and $s_1 < s_2 < \cdots < s_k$. It implies that $\sigma(h, x_{t_1}) = 0$. Further, we have $\sigma(h, x_v) = 0$ since $\sigma(h, x_v) \leq \sigma(g, x_v) = 0$. Then the transformation $\boldsymbol{x} \leftarrow I_{v \leftarrow s_1, \cdots, s_k}$ carries the function $\hat{f}$ into

$$\hat{f} \leftarrow x_v x_{t_1} + h(\boldsymbol{x} \cdot I_{v \leftarrow s_1, \cdots, s_k}).$$

Thus, we have $\sigma(\hat{f}, x_{t_1}) = 1$ and $\sigma(\hat{f}, x_v) = 1$. This means that $x_v x_{t_1}$ is a separated quadratic term. □

Theorem 1. *Given a quadratic form $f(\boldsymbol{x}) = f(x_1, \cdots, x_n)$, Algorithm 1 outputs a disjoint quadratic form $\hat{f}(\boldsymbol{x})$ and an invertible $n \times n$ matrix M, such that $\hat{f}(\boldsymbol{x}) = f(\boldsymbol{x}M)$. Moreover, Algorithm 1 has time complexity $\mathcal{O}(n^{3.8})$ and memory complexity $\Omega(n^2)$.*

Proof. According to Lemma 5, each **while** loop will generate at least one separated quadratic term. Hence, after at most $n/2$ **while** loops, all quadratic terms of the current $\hat{f}$ are disjoint quadratic terms.

Now we briefly analyze the complexity of Algorithm 1. From the above analysis, Algorithm 1 will have $n/2$ **while** loops in the worst case. This implies that the time complexity is upper bounded by the n matrix multiplications. Therefore, the time complexity of the algorithm can be estimated as $\mathcal{O}(n^{1+2.8})$, where we take $\mathcal{O}(n^{2.8})$ as the time complexity of the multiplication two $n \times n$ matrices [31]. It is readily seen that the memory complexity is $\Omega(n^2)$. □

To sum up, with Lemma 3, Lemma 4, and Algorithm 1, we can compute the correlation of any quadratic Boolean function with polynomial time complexity.

4 Exploitable Linear Approximations of MORUS-like Key Stream Generators

We consider a typical stream cipher construction shown in Fig. 6. A partially unknown state S^U (initialized with a secret key and some public values) is processed by an initialization algorithm. Then a vectorial Boolean function $\mathcal{G}$ is applied to the state S^0 to produce one key stream word Z^0. For $0 \leq i < k$, a

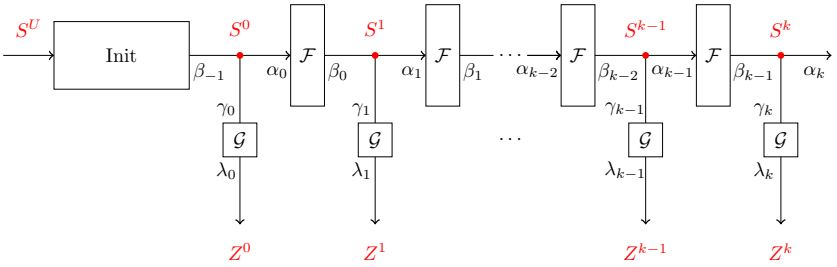


Fig. 6: Linear trails for MORUS-like key-stream generator

state update function is employed to obtain a new state $S^{i+1} = \mathcal{F}(S^i)$, from which a key stream word $Z^{i+1} = \mathcal{G}(S^{i+1})$ is extracted.

For this kind of stream ciphers, a generic attack based on linear cryptanalysis (e.g.,[25]) can be applied, whose goal is to find a sequence of linear masks $(\lambda_0, \cdots, \lambda_k)$ for the key-stream blocks Z^i, such that the absolute value of the correlation $\mathrm{cor}\left(\sum_{i=0}^{k} \lambda_i Z^i\right)$ can be maximized, where the number of ciphertext blocks involved in the linear approximation is called the *span*. In what follows, we establish a model in which finding $(\lambda_0, \cdots, \lambda_k)$ is *conceptually* the same as finding linear trails of a block cipher with additional constraints imposed on some linear masks at some special positions. With this model, existing tools [23,8,32,34,14] for finding good linear trails of block ciphers can be applied to search for $(\lambda_0, \cdots, \lambda_k)$.

Definition 5. *A linear trail of the key stream generator shown in Fig. 6:*

$$(\beta_{-1}, \gamma_0, \lambda_0, \alpha_0, \beta_0, \cdots, \alpha_{k-1}, \beta_{k-1}, \gamma_k, \lambda_k, \alpha_k)$$

is said to be exploitable if and only if $\beta_{-1} = 0$, $\alpha_k = 0$, *and* $\alpha_i + \gamma_i + \beta_{i-1} = 0$ *for* $0 \le i \le k$.

The motivation behind Definition 5 is that when the following equations

$$\begin{cases} \beta_{-1} = 0 \\ \alpha_k = 0 \\ \alpha_i + \gamma_i + \beta_{i-1} = 0, & 0 \le i \le k \\ \gamma_i S^i + \lambda_i Z^i = 0, & 0 \le i \le k \\ \alpha_i S^i + \beta_i S^{i+1} = 0, & 0 \le i \le k-1 \end{cases} \tag{4}$$

hold simultaneously, we have

$$\sum_{i=0}^{k} \lambda_i Z^i = \sum_{i=0}^{k} \gamma_i S^i = \beta_{-1} S^0 + \sum_{i=0}^{k-1} (\alpha_i S^i + \beta_i S^{i+1}) + \alpha_k S^k = 0. \tag{5}$$

Although in Definition 5 we require $\beta_{-1} = 0$, in fact, any characteristic starting with some $\beta_i = 0$ that follows the same pattern specified in Definition 5 across several consecutive ciphertext blocks can be exploited.

In this work, the MILP-based approach [32,34,14] is employed to search for linear trails of MORUS. One solution of the MILP model is a linear characteristic satisfying additional constraints specified in Definition 5. The objective function of the model is to minimize the number of active AND gates. The trails produced by the models are only *locally consistent*, and thus we cannot guarantee their global soundness with respect to optimality and validity, since the models are constructed under the assumption that all AND gates are independent.

Let us inspect a toy example where $f = f_1 + f_2 = x_1x_2 + x_1x_3 + x_2$ and

$$\begin{cases} f_1(x_1, x_2, x_3) = x_1x_2 + x_2 \\ f_2(x_1, x_2, x_3) = x_1x_3 \end{cases}.$$

The reader can check that in this case $\mathrm{cor}(f_1) = \mathrm{cor}(f_2) = 2^{-1}$, but $\mathrm{cor}(f) = 0$, which implies that the sum of biased Boolean functions may be balanced. *Therefore, global consistency of the full trail cannot be ensured by local consistency.* To be more concrete, we show a real example. Table 4 presents an invalid linear trail generated by our MILP model whose span is 3. Note that in this paper, we show our trails in their linear-mask representations. There is a correspondence between the linear-mask representation and the trail-equation representation used in [2]. The five linear masks between α_0 and β_0 listed in Table 4 are the linear masks in the positions shown in Fig. 5 marked with dashed lines. Each row of the linear masks determines which AND gates are activated, and each active AND gate produces one equation containing one product term. By adding up these equations, we can reproduce the trail-equation representations used in [2]. In this work, we always need to convert the linear-mask representation into the trail-equation representation, which is required to determine its overall correlation by using the method proposed in Sect. 3.

For the sake of completeness, we give a complete example of the conversion process based on the trail shown in Table 4. From the linear masks, we can get the following equations:

$$\begin{aligned}
C^0_{30} \oplus S^0_{0,30} \oplus S^0_{1,30} &= S^0_{2,30} \cdot S^0_{3,30} \\
C^0_{22} \oplus S^0_{0,22} \oplus S^0_{1,22} &= S^0_{2,22} \cdot S^0_{3,22} \\
S^0_{0,30} \oplus S^1_{0,3} \oplus S^0_{3,30} &= S^0_{1,30} \cdot S^0_{2,30} \\
S^0_{0,22} \oplus S^1_{0,27} \oplus S^0_{3,22} &= S^0_{1,22} \cdot S^0_{2,22} \\
S^0_{1,22} \oplus S^1_{1,21} \oplus S^0_{4,22} &= S^0_{2,22} \cdot S^0_{3,22} \\
S^0_{4,22} \oplus S^1_{4,3} \oplus S^1_{2,22} &= S^1_{0,22} \cdot S^1_{1,22} \\
C^1_{29} \oplus S^1_{0,29} \oplus S^1_{1,29} &= S^1_{2,29} \cdot S^1_{3,29} \\
C^1_{27} \oplus S^1_{0,27} \oplus S^1_{1,27} &= S^1_{2,27} \cdot S^1_{3,27} \\
C^1_{22} \oplus S^1_{0,22} \oplus S^1_{1,22} &= S^1_{2,22} \cdot S^1_{3,22} \\
C^1_{21} \oplus S^1_{0,21} \oplus S^1_{1,21} &= S^1_{2,21} \cdot S^1_{3,21} \\
C^1_{3} \oplus S^1_{0,3} \oplus S^1_{1,3} &= S^1_{2,3} \cdot S^1_{3,3}
\end{aligned}$$

$$
\begin{aligned}
S^1_{0,29} \oplus S^2_{0,2} \oplus S^1_{3,29} &= S^1_{1,29} \cdot S^1_{2,29} \\
S^1_{0,22} \oplus S^2_{0,27} \oplus S^1_{3,22} &= S^1_{1,22} \cdot S^1_{2,22} \\
S^1_{0,21} \oplus S^2_{0,26} \oplus S^1_{3,21} &= S^1_{1,21} \cdot S^1_{2,21} \\
S^1_{1,27} \oplus S^2_{1,26} \oplus S^1_{4,27} &= S^1_{2,27} \cdot S^1_{3,27} \\
S^1_{1,3} \oplus S^2_{1,2} \oplus S^1_{4,3} &= S^1_{2,3} \cdot S^1_{3,3} \\
S^1_{2,27} \oplus S^2_{2,2} \oplus S^2_{0,27} &= S^1_{3,27} \cdot S^1_{4,27} \\
C^2_{26} \oplus S^2_{0,26} \oplus S^2_{1,26} &= S^2_{2,26} \cdot S^2_{3,26} \\
C^2_{2} \oplus S^2_{0,2} \oplus S^2_{1,2} &= S^2_{2,2} \cdot S^2_{3,2}
\end{aligned}
$$

Adding up the above equations gives the trail equation:

$$
\begin{aligned}
& C^0_{30} \oplus C^0_{22} \oplus C^1_{29} \oplus C^1_{27} \oplus C^1_{22} \oplus C^1_{21} \oplus C^1_{3} \oplus C^2_{26} \oplus C^2_{2} \\
&= S^1_{2,22} \cdot S^1_{3,22} \oplus S^1_{1,22} \cdot S^1_{2,22} \oplus S^1_{2,22} \oplus S^1_{3,22} \oplus S^1_{1,22} \\
&\oplus S^1_{2,21} \cdot S^1_{3,21} \oplus S^1_{1,21} \cdot S^1_{2,21} \oplus S^1_{3,21} \\
&\oplus S^1_{2,29} \cdot S^1_{3,29} \oplus S^1_{1,29} \cdot S^1_{2,29} \oplus S^1_{3,29} \oplus S^1_{1,29} \\
&\oplus S^0_{2,30} \cdot S^0_{3,30} \oplus S^0_{1,30} \cdot S^0_{2,30} \oplus S^0_{3,30} \oplus S^0_{1,30} \\
&\oplus S^0_{1,22} \cdot S^0_{2,22} \\
&\oplus S^1_{0,22} \cdot S^1_{1,22} \\
&\oplus S^1_{3,27} \cdot S^1_{4,27} \oplus S^1_{4,27} \\
&\oplus S^2_{2,26} \cdot S^2_{3,26} \\
&\oplus S^2_{2,2} \cdot S^2_{3,2} \oplus S^2_{2,2} \\
&\oplus S^0_{3,22} \\
&\oplus S^1_{2,27}.
\end{aligned}
$$

The right-hand side of the equation is a quadratic Boolean function. Thus by applying the method shown in Sect. 3, we can obtain its correlation. However, for this special case, we know that its correlation is zero without converting it into the disjoint quadratic form, since the variable $S^1_{2,27}$ never appears in any other term of the quadratic Boolean function. Thus, according to Corollary 1, the correlation of $C^0_{30} \oplus C^0_{22} \oplus C^1_{29} \oplus C^1_{27} \oplus C^1_{22} \oplus C^1_{21} \oplus C^1_{3} \oplus C^2_{26} \oplus C^2_{2}$ is zero.

At this point, we emphasize that Definition 5 is only used as a mental helper to identify *potentially* good trails. Since in practice, we apply search tools that produce "good" linear trails assuming the independencies of the rounds or components within $\mathcal{F}$ and $\mathcal{G}$. However, these assumptions are generally not true as illustrated by the above example. Therefore, the outputs of the search tools are not reliable. We must *recompute the correlation of the full trail* by using dedicated methods which are suitable to the target under consideration. For instance, using the method presented in Sect. 3, we automatically detect such inconsistencies shown in the above examples.

Table 4: An invalid trail of MiniMORUS-640 with span 3

Round		Linear masks
0	α_0	40400000 40400000 00000000 40400000 00000000
		08000008 00400000 00000000 00000000 00000000
		08000008 00200000 00000000 00000000 00400000
		08000008 00200000 00000000 00000000 00400000
		08000008 00200000 00000000 00000000 00400000
	β_0	08000008 00200000 00400000 00000000 00000008
	γ_0	40400000 40400000 00000000 40400000 00000000
	λ_0	40400000
1	α_1	20600000 28400008 00400000 20600000 00000008
		0c000004 08000008 00000000 00000000 00000008
		0c000004 04000004 08000000 00000000 08000000
		04000004 04000004 00000004 00000000 00000000
		04000004 04000004 00000004 00000000 00000000
	β_1	04000004 04000004 00000004 00000000 00000000
	γ_1	28600008 28600008 00000000 20600000 00000000
	λ_1	28600008
2	γ_2	04000004 04000004 00000004 00000000 00000000
	λ_2	04000004

5 Searching for Linear Approximations of MORUS

By setting the plaintext to zero message as in [2], MiniMORUS and MORUS fit exactly into the model established in Sect. 4. Hence, linear trails of MiniMORUS and MORUS can be searched by using any existing tools for finding linear approximations. In our work, we apply the MILP-based approach, where the constraints imposed on the linear trails are encoded into MILP models.

In practice, we must determine the number of ciphertext blocks involved in the final linear combination of the ciphertext bits before we can set up the MILP model. First, we theoretically show that there is no useful linear approximation for MORUS involving only one ciphertext block. Let λ_0 be a linear mask of the key-stream generator shown in Fig. 6 for one ciphertext block. Then we have

$$\begin{aligned}\lambda_0 Z^0 &= \bigoplus_{j,\lambda_{0,j}=1} (S^0_{0,j} \oplus (S^0_{1,j+b'_2} \oplus S^0_{2,j} \cdot S^0_{3,j})) \\ &= \bigoplus_{j,\lambda_{0,j}=1} S^0_{0,j} \oplus \bigoplus_{j,\lambda_{0,j}=1} (S^0_{1,j+b'_2} \oplus S^0_{2,j} \cdot S^0_{3,j}).\end{aligned}$$

Since the variable $S^0_{0,j}$ does not appear in other terms, we have $\mathrm{cor}(\lambda_0 Z^0) = 0$ according to Corollary 1.

Since the linear trails used in [2] span across 5 ciphertext blocks, we decide to only search for rotational invariant trails with spans greater than 1 and less than 6 (models for larger spans will have more variables which are difficult to solve). The best trails we found are of span 4, and the trails for MiniMORUS-640 and MORUS-640 are listed in Table 5 and Table 6, respectively.

As an illustration, let us compute the correlation of the trail of MiniMORUS-640 shown in Table 5. Firstly, according to the linear masks shown in Table 5,

Table 5: A linear trail of MiniMORUS-640 with correlation -2^{-8}

Round	Linear masks
0	α_0 10000000 10000000 00000000 10000000 00000000 00000002 00000000 00000000 00000000 00000000 00000002 00000000 00000000 00000000 00000000 00000002 00000000 00000000 00000000 00000000 00000002 00000000 00000000 00000000 00000000
	β_0 00000002 00000000 00000000 00000000 00000000
	γ_0 10000000 10000000 00000000 10000000 00000000
	λ_0 10000000
1	α_1 08000200 08000202 00000002 08000200 00000000 00004001 00000002 00000002 00000000 00000000 00004001 00000001 00000000 00000000 00000002 00004001 00000001 00000000 00000000 00000002 00004001 00000001 00000000 00000000 00000002
	β_1 00004003 00000003 00000002 00000000 00004000
	γ_1 08000202 08000202 00000002 08000200 00000000
	λ_1 08000202
2	α_2 00000100 00004100 00000000 00000100 00004000 00002000 00004000 00000000 00000000 00004000 00002000 00002000 00000000 00000000 00000000 00002000 00002000 00000000 00000000 00000000 00002000 00002000 00000000 00000000 00000000
	β_2 00002000 00002000 00000000 00000000 00000000
	γ_2 00004103 00004103 00000002 00000100 00000000
	λ_2 00004103
3	γ_3 00002000 00002000 00000000 00000000 00000000
	λ_3 00002000

we write down the following equations which hold with probability 1.

$$
\begin{aligned}
C^0_{28} \oplus S^0_{0,28} \oplus S^0_{1,28} &= S^0_{2,28} \cdot S^0_{3,28} \\
S^0_{0,28} \oplus S^1_{0,1} \oplus S^0_{3,28} &= S^0_{1,28} \cdot S^0_{2,28} \\
C^1_{27} \oplus S^1_{0,27} \oplus S^1_{1,27} &= S^1_{2,27} \cdot S^1_{3,27} \\
C^1_{9} \oplus S^1_{0,9} \oplus S^1_{1,9} &= S^1_{2,9} \cdot S^1_{3,9} \\
C^1_{1} \oplus S^1_{0,1} \oplus S^1_{1,1} &= S^1_{2,1} \cdot S^1_{3,1} \\
S^1_{0,27} \oplus S^2_{0,0} \oplus S^1_{3,27} &= S^1_{1,27} \cdot S^1_{2,27} \\
S^1_{0,9} \oplus S^2_{0,14} \oplus S^1_{3,9} &= S^1_{1,9} \cdot S^1_{2,9} \\
S^1_{1,1} \oplus S^2_{1,0} \oplus S^1_{4,1} &= S^1_{2,1} \cdot S^1_{3,1} \\
S^1_{4,1} \oplus S^2_{4,14} \oplus S^2_{2,1} &= S^2_{0,1} \cdot S^2_{1,1} \\
C^2_{14} \oplus S^2_{0,14} \oplus S^2_{1,14} &= S^2_{2,14} \cdot S^2_{3,14} \\
C^2_{8} \oplus S^2_{0,8} \oplus S^2_{1,8} &= S^2_{2,8} \cdot S^2_{3,8} \\
C^2_{1} \oplus S^2_{0,1} \oplus S^2_{1,1} &= S^2_{2,1} \cdot S^2_{3,1} \\
C^2_{0} \oplus S^2_{0,0} \oplus S^2_{1,0} &= S^2_{2,0} \cdot S^2_{3,0} \\
S^2_{0,8} \oplus S^3_{0,13} \oplus S^2_{3,8} &= S^2_{1,8} \cdot S^2_{2,8} \\
S^2_{1,14} \oplus S^3_{1,13} \oplus S^2_{4,14} &= S^2_{2,14} \cdot S^2_{3,14}
\end{aligned}
$$

Table 6: A linear trail of MORUS-640 with correlation 2^{-38}, where "*4" stands for 4 copies of the same bit string

Round		Linear masks
0	α_0	10000000*4 10000000*4 00000000*4 10000000*4 00000000*4
		00000002*4 00000000*4 00000000*4 00000000*4 00000000*4
		00000002*4 00000000*4 00000000*4 00000000*4 00000000*4
		00000002*4 00000000*4 00000000*4 00000000*4 00000000*4
		00000002*4 00000000*4 00000000*4 00000000*4 00000000*4
	β_0	00000002*4 00000000*4 00000000*4 00000000*4 00000000*4
	γ_0	10000000*4 10000000*4 00000000*4 10000000*4 00000000*4
	λ_0	10000000*4
1	α_1	08000200*4 08000202*4 00000002*4 08000200*4 00000000*4
		00004001*4 00000002*4 00000002*4 00000000*4 00000000*4
		00004001*4 00000001*4 00000000*4 00000000*4 00000002*4
		00004001*4 00000001*4 00000000*4 00000000*4 00000002*4
		00004001*4 00000001*4 00000000*4 00000000*4 00000002*4
	β_1	00004003*4 00000003*4 00000002*4 00000000*4 00004000*4
	γ_1	08000202*4 08000202*4 00000002*4 08000200*4 00000000*4
	λ_1	08000202*4
2	α_2	00000100*4 00004100*4 00000000*4 00000100*4 00004000*4
		00002000*4 00004000*4 00000000*4 00000000*4 00004000*4
		00002000*4 00002000*4 00000000*4 00000000*4 00000000*4
		00002000*4 00002000*4 00000000*4 00000000*4 00000000*4
		00002000*4 00002000*4 00000000*4 00000000*4 00000000*4
	β_2	00002000*4 00002000*4 00000000*4 00000000*4 00000000*4
	γ_2	00004103*4 00004103*4 00000002*4 00000100*4 00000000*4
	λ_2	00004103*4
3	γ_3	00002000*4 00002000*4 00000000*4 00000000*4 00000000*4
	λ_3	00002000*4

$$C^3_{13} \oplus S^3_{0,13} \oplus S^3_{1,13} = S^3_{2,13} \cdot S^3_{3,13}$$

Combining the above equations, we obtain an equation whose left-hand side involves only cipher-text bits, while the right-hand side of the equation can be regarded as a quadratic Boolean function.

$$\begin{aligned}
&C^0_{28} \oplus C^1_{27} \oplus C^1_9 \oplus C^1_1 \oplus C^2_{14} \oplus C^2_8 \oplus C^2_1 \oplus C^2_0 \oplus C^3_{13} \\
&= S^0_{2,28} \cdot S^0_{3,28} \oplus S^0_{1,28} \cdot S^0_{2,28} \oplus S^0_{3,28} \oplus S^0_{1,28} \\
&\oplus S^1_{2,9} \cdot S^1_{3,9} \oplus S^1_{1,9} \cdot S^1_{2,9} \oplus S^1_{3,9} \oplus S^1_{1,9} \\
&\oplus S^1_{2,27} \cdot S^1_{3,27} \oplus S^1_{1,27} \cdot S^1_{2,27} \oplus S^1_{3,27} \oplus S^1_{1,27} \\
&\oplus S^2_{2,8} \cdot S^2_{3,8} \oplus S^2_{1,8} \cdot S^2_{2,8} \oplus S^2_{3,8} \oplus S^2_{1,8} \\
&\oplus S^2_{0,1} \cdot S^2_{1,1} \oplus S^2_{0,1} \oplus S^2_{1,1} \\
&\oplus S^2_{2,1} \cdot S^2_{3,1} \oplus S^2_{2,1} \\
&\oplus S^2_{2,0} \cdot S^2_{3,0} \\
&\oplus S^3_{2,13} \cdot S^3_{3,13}
\end{aligned}$$

The right-hand side of the above equation can be transformed into its disjoint quadratic form with the method presented in Sect. 3.

$$S^0_{2,28} \cdot S^0_{3,28} \oplus S^0_{1,28} \cdot S^0_{2,28} \oplus S^0_{3,28} \oplus S^0_{1,28}$$

$$
\begin{aligned}
&\oplus S^1_{2,9} \cdot S^1_{3,9} \oplus S^1_{1,9} \cdot S^1_{2,9} \oplus S^1_{3,9} \oplus S^1_{1,9} \\
&\oplus S^1_{2,27} \cdot S^1_{3,27} \oplus S^1_{1,27} \cdot S^1_{2,27} \oplus S^1_{3,27} \oplus S^1_{1,27} \\
&\oplus S^2_{2,8} \cdot S^2_{3,8} \oplus S^2_{1,8} \cdot S^2_{2,8} \oplus S^2_{3,8} \oplus S^2_{1,8} \\
&\oplus S^2_{0,1} \cdot S^2_{1,1} \oplus S^2_{0,1} \oplus S^2_{1,1} \\
&\oplus S^2_{2,1} \cdot S^2_{3,1} \oplus S^2_{2,1} \\
&\oplus S^2_{2,0} \cdot S^2_{3,0} \\
&\oplus S^3_{2,13} \cdot S^3_{3,13} \\
&= (S^0_{2,28} \oplus 1)(S^0_{1,28} \oplus S^0_{3,28}) \\
&\oplus (S^1_{2,9} \oplus 1)(S^1_{1,9} \oplus S^1_{3,9}) \\
&\oplus (S^1_{2,27} \oplus 1)(S^1_{1,27} \oplus S^1_{3,27}) \\
&\oplus (S^2_{2,8} \oplus 1)(S^2_{1,8} \oplus S^2_{3,8}) \\
&\oplus (S^2_{0,1} \oplus 1)(S^2_{1,1} \oplus 1) \\
&\oplus S^2_{2,1}(S^2_{3,1} \oplus 1) \\
&\oplus S^2_{2,0} \cdot S^2_{3,0} \\
&\oplus S^3_{2,13} \cdot S^3_{3,13} \oplus 1
\end{aligned}
$$

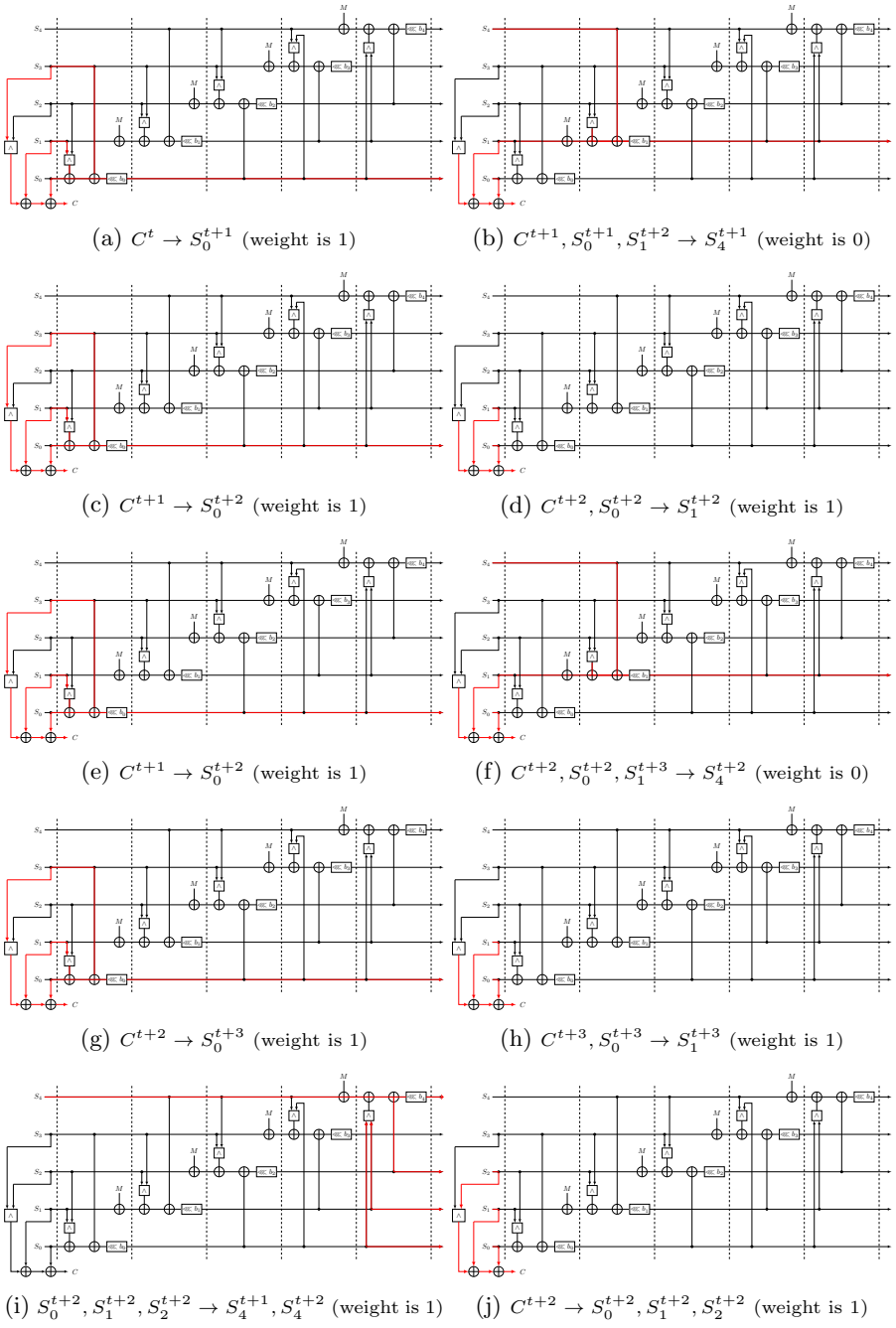
Therefore, the correlation of $C^0_{28} \oplus C^1_{27} \oplus C^1_9 \oplus C^1_1 \oplus C^2_{14} \oplus C^2_8 \oplus C^2_1 \oplus C^2_0 \oplus C^3_{13}$ is -2^{-8}. Similarly, we can compute the correlations of the trails of MORUS-640, MiniMORUS-1280, and MORUS-1280.

Before going any further, we would like to give some insight into the trails of MiniMORUS to show how the linear approximations covering different parts of the cipher eventually eliminate all internal variables, leading to approximations involving only ciphertext variables. The following discussion is similar to the Section 4 of [2]. Several fragments are common between [2] and ours. We recommend the reader to review the Figure 2 of [2] before reading the following part.

We can use the variables of C^t to approximate the variables of S^{t+1}_0, denoted by $C^t \rightarrow S^{t+1}_0$. At the same time, $C^{t+1}, S^{t+1}_0, S^{t+2}_1 \rightarrow S^{t+1}_4$. These approximations are visualized in Fig. 7a and Fig. 7b, Note that two AND operations are involved in Fig. 7a, in which one is approximated to S_3 and the other is approximated to S_1. This seems to require weight 2. This trail fragment is the same as one of the fragments in [2], and [2] explains that there is another way of approximating those two AND operations: one is approximated to $S_3 \oplus S_2$ and the other is approximated to $S_2 \oplus S_1$. Two ways of the approximation form a hull effect, which makes its weight 1.

Fig. 7b was also used in [2], which involves two AND operations. Those AND operations take the same input variables, S_2 and S_3. Hence those two deterministically cancel each other, which makes the weight of this fragment 0.

Basically, by combining the fragments in Fig. 7a to Fig. 7d, 1 bit of S^{t+1}_4 is approximated from the ciphertext bits. We do the same to approximate 1 bit of S^{t+2}_4 by sliding the steps by 1. Fig. 7e to Fig. 7h are for this approximation.



(a) $C^t \to S_0^{t+1}$ (weight is 1)

(b) $C^{t+1}, S_0^{t+1}, S_1^{t+2} \to S_4^{t+1}$ (weight is 0)

(c) $C^{t+1} \to S_0^{t+2}$ (weight is 1)

(d) $C^{t+2}, S_0^{t+2} \to S_1^{t+2}$ (weight is 1)

(e) $C^{t+1} \to S_0^{t+2}$ (weight is 1)

(f) $C^{t+2}, S_0^{t+2}, S_1^{t+3} \to S_4^{t+2}$ (weight is 0)

(g) $C^{t+2} \to S_0^{t+3}$ (weight is 1)

(h) $C^{t+3}, S_0^{t+3} \to S_1^{t+3}$ (weight is 1)

(i) $S_0^{t+2}, S_1^{t+2}, S_2^{t+2} \to S_4^{t+1}, S_4^{t+2}$ (weight is 1)

(j) $C^{t+2} \to S_0^{t+2}, S_1^{t+2}, S_2^{t+2}$ (weight is 1)

Fig. 7: MiniMORUS linear trail fragments

Hence by removing the step indices, Fig. 7e to Fig. 7h are exact copies of Fig. 7a to Fig. 7d.

Note that the linear trail up to here, which has weight 6, is identical with [2]. Ashur et al. [2] iterated this approximation twice and added 4 more approximations, which makes the weight of their trail $(6 \times 2) + 4 = 16$. The core of our improvement lies in the detection of a rather complicated new approximation that approximates S_4^{t+1} and S_4^{t+2} by ciphertext bits only with weight 2. The new approximations are shown in Fig. 7i to Fig. 7j, in which S_4^{t+1} and S_4^{t+2} are approximated to the 3-bit sum of S_0^{t+2}, S_1^{t+2} and S_2^{t+2}, and C^{t+2} are also approximated to the 3-bit sum of $S_0^{t+2}, S_1^{t+2}, S_2^{t+2}$. The previous work [2] found the attack by hand thus the most of the approximations are simple such that 2 internal state bits are approximated to 1-bit of another state. thanks to the generic model in Sect. 4, we could detect this efficient approximation

We stress that the trail fragments are only used to shed insight on the full trails, and the verification of these trail fragments are only used to provided additional evidence of the validity of the analysis. *We never use trail fragments to compute the correlation. The correlation must be computed on the full trail as whole.*

Remark. we would like to make a remark on the effect of the in-word rotation ($\lll_w$) offsets (b_i, $i \in \{0, \cdots, 4\}$) of MORUS on the linear trails we find. In [2], Ashur et al. assumes that the trails work for any choice of b_i without any concrete discussion of the actual effect. We randomly choose 50 different $(b_0, b_1, b_2, b_3, b_4)$'s and generate 50 MILP models to search for their trails. We do observe slight variance of the correlations of the trails we find for different choices of $(b_0, b_1, b_2, b_3, b_4)$. For example, in the case of $(b_0, b_1, b_2, b_3, b_4) = (16, 31, 23, 3, 17)$, we identify a trail of MORUS-640 with correlation 2^{-34}, meaning that under our current cryptanalysis technique, this version is weaker than the original design.

5.1 Distinguishing Attack and Message-recovery Attack on MORUS

So far, for the sake of simplicity, we have assumed that all message blocks are zero. As already pointed out in [2], message variables only contribute linearly to the trails.

Therefore, under the condition that the involved message bits are kept constants, the trails we identified can be employed to mount two types of attacks. The first one is a (partially) known-plaintext distinguishing attack, where a large number of partially known plaintexts are encrypted, and then we can detect the bias from the ciphertexts. The second one is a message-recovery attack, in which we can recover some unknown plaintext bits if the same plaintext is encrypted for many times. The scenario in which the message-recovery attack can be applied does happen in practice. For example, the same message can be encrypted with different IVs and potentially different keys in the so-called broadcast setting [20,1].

For the message-recovery attacks, we rely on the approach proposed by Matsui [22]. For example, if the correlation of the trail employed in our message-recovery attack is 2ρ, we would encrypt a (unknown) message approximately n times with different nonces or keys. Let T_b be the number of encryptions such that the linear combination (derived from the trail) of the ciphertext bits is equal to $b \in \{0, 1\}$. Then we guess the value of the linear combination $\mathcal{L}(M)$ of the message M according to the following rule:

$$\mathcal{L}(M) = \begin{cases} 0, & \text{if } T_0 > T_1 \text{ and } \rho > 0, \\ 1, & \text{if } T_0 > T_1 \text{ and } \rho < 0, \\ 1, & \text{if } T_1 > T_0 \text{ and } \rho > 0, \\ 0, & \text{if } T_1 > T_0 \text{ and } \rho < 0. \end{cases}$$

The success probability of the procedure can be estimated as $\int_{-2\sqrt{n}|\rho|}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx$, which would be greater than 84.1% if we set $n > \frac{1}{4}|\rho|^{-2}$ [22]. Therefore, if the correlation of the underlying approximation is 2^{-c}, we need about 2^{2c} encryptions to mount the attack.

On the data complexity. As pointed out by Ashur et al. [2], the data complexities of the attacks could be slightly lowered by using multiple linear trails [16,15,6]. Actually, given any trail found in this paper, we can derive another trail with the same correlation by rotating the masks within words by a common offset. If we assume independency, we could run $q/4$ (the word size) copies of the trail in parallel on the same encrypted blocks, which would save a factor of 2^5 on the data complexity for MORUS-640, and 2^6 for MORUS-1280.

5.2 Verification of the Attacks

To confirm the validity of our analysis, we experimentally verify the trails or trail fragments. For MiniMORUS, we are able to fully verify the correlations. Experiments show that the weights of the correlations of

$$C_{28}^0 \oplus C_{27}^1 \oplus C_9^1 \oplus C_1^1 \oplus C_{14}^2 \oplus C_8^2 \oplus C_1^2 \oplus C_0^2 \oplus C_{13}^3$$

and

$$C_{16}^0 \oplus C_{62}^1 \oplus C_{29}^1 \oplus C_{20}^1 \oplus C_{33}^2 \oplus C_{29}^2 \oplus C_{11}^2 \oplus C_2^2 \oplus C_{15}^3$$

for MiniMORUS-640 and MiniMORUS-1280 are 7.7919 and 8.1528 respectively, which are quite close to 8, the theoretically predicted correlation.

For MORUS, the correlation of the best trails we find is 2^{-38}, indicating that about 2^{76} encryptions have to be performed to verify the full trail, which is out of our reach. Following the approach presented in [2], we decompose the full trail into trail fragments according to Fig. 7, and every fragment is verified independently.

For MORUS-640 and MORUS-1280, the full trails can be divided into five trail fragments shown in Table 7 and Table 8, respectively. We independently

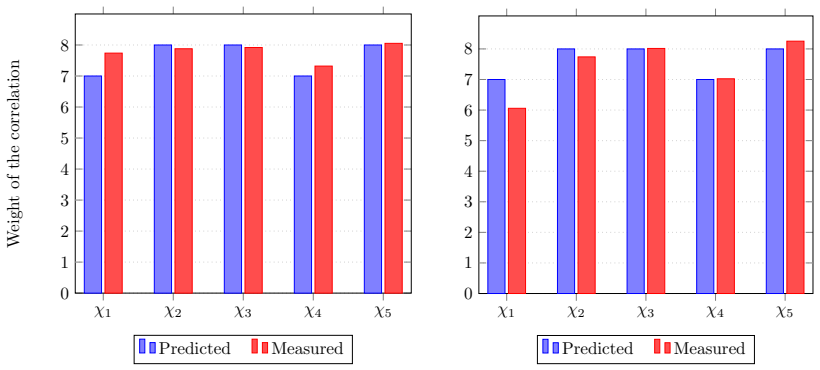
verify these trail fragments and the results are given in Fig. 8a and Fig. 8b. Again, the results fit the theoretical analysis very well.

Table 7: The five trail fragments of MORUS-640

	Trail fragment	Weight
χ_1	$C^0_{\{124,92,60,28\}} \oplus C^1_{\{97,65,33,1\}} = S^1_{4,\{97,65,33,1\}} \oplus S^2_{1,\{96,64,32,0\}}$	7
χ_2	$C^1_{\{123,91,59,27\}} \oplus C^2_{\{96,64,32,0\}} = S^2_{1,\{96,64,32,0\}}$	8
χ_3	$C^2_{\{104,72,40,8\}} \oplus C^3_{\{109,77,45,13\}} = S^3_{1,\{109,77,45,13\}}$	8
χ_4	$C^1_{\{105,73,41,9\}} \oplus C^2_{\{110,78,46,14\}} = S^3_{1,\{109,77,45,13\}} \oplus S^2_{4,\{110,78,46,14\}}$	7
χ_5	$C^2_{\{97,65,33,1\}} = S^1_{4,\{97,65,33,1\}} \oplus S^2_{4,\{110,78,46,14\}}$	8

Table 8: The five trail fragments of MORUS-1280

	Trail fragment	Weight
χ_1	$C^0_{\{208,144,80,16\}} \oplus C^1_{\{221,157,93,29\}} = S^1_{4,\{221,157,93,29\}} \oplus S^2_{1,\{203,139,75,11\}}$	7
χ_2	$C^1_{\{254,190,126,62\}} \oplus C^2_{\{203,139,75,11\}} = S^2_{1,\{203,139,75,11\}}$	8
χ_3	$C^2_{\{194,130,66,2\}} \oplus C^3_{\{207,143,79,15\}} = S^3_{1,\{207,143,79,15\}}$	8
χ_4	$C^1_{\{212,148,84,20\}} \oplus C^2_{\{225,161,97,33\}} = S^3_{1,\{207,143,79,15\}} \oplus S^2_{4,\{225,161,97,33\}}$	7
χ_5	$C^2_{\{221,157,93,29\}} = S^1_{4,\{221,157,93,29\}} \oplus S^2_{4,\{225,161,97,33\}}$	8



(a) MORUS-640 (b) MORUS-1280

Fig. 8: Experimental verification of the trail fragments of MORUS-640 and MORUS-1280

6 Searching for Trails with Smaller Spans

In [2], it is said that ciphertext correlations like those presented in previous sections can be exploited only when the same message is encrypted enough times:

> " *... they can be leveraged to mount an attack in the broadcast setting, where the same message is encrypted multiple times with different IVs and potentially different keys [21]. In particular, the broadcast setting appears in practice in man-in-the-browser attacks against HTTPS connections following the BEAST model [10].* "

However, we find that this strong condition can be relaxed. Let us recall Fig. 4, and consider a trail with a 4-block span. If we encrypt a set of n-block ($n > 4$) messages sharing a common 4-block prefix $M^0 \parallel M^1 \parallel M^2 \parallel M^3$, then our analysis presented in previous sections is *completely irrelevant* with those message blocks beyond this common prefix. In fact, if we encrypt $M^0 \parallel M^1 \parallel M^2 \parallel M^3$ and $M^0 \parallel M^1 \parallel M^2 \parallel M^3 \parallel \cdots \parallel M^{n-1}$ with the same key, nonce, and associated data, the same intermediate values and ciphertexts will be produced within the 4-block span. Therefore, we can draw the conclusion that *the correlations involving k-block ciphertext can be leveraged to mount an attack if enough messages with a k-block common prefix are encrypted with different IV $\parallel$ key.*

Note that the above condition is strictly weaker than that presented in ASIACRYPT 2018 [2], and this setting does occur in practice. For example, when ARP packets are encrypted in WPA2-AES enabled WIFI networks, they share a 16-byte common prefix (8-byte LLC header and 8-byte ARP request header) [9]. This 16-byte common prefix extends to 22 bytes if the attacker is able to control the following 6-byte MAC address, which is not difficult to carry out [33]. Therefore, trails with smaller spans are more preferable, which motivates us to search for linear trails with smaller spans. The best trail with respect to the number of ciphertext blocks involved (span) we find is a trail of MORUS-640 with correlation 2^{-79}, whose span is 3 (see Table 9). However, the correlation is too low to be used in an attack.

Table 9: A linear trail of MORUS-640 with correlation 2^{-79} whose span is 3

Round		Linear masks
0	α_0	00002520*4 00002520*4 00002020*4 00002520*4 00000000*4
		0004a400*4 00000000*4 00002020*4 00000000*4 00000000*4
		0004a400*4 00000000*4 00002020*4 00000000*4 00000000*4
		00048420*4 00000000*4 00101000*4 00000020*4 00000020*4
		00048400*4 00000020*4 00101000*4 08000000*4 00000020*4
	β_0	00048420*4 00000000*4 00101020*4 08000000*4 00040000*4
	γ_0	00002520*4 00002520*4 00002020*4 00002520*4 00000000*4
	λ_0	00002520*4
1	α_1	00009420*4 00041000*4 00140020*4 08041000*4 00040000*4
		00128400*4 00040000*4 00140400*4 08048420*4 00040000*4
		00128400*4 00020000*4 00100400*4 08008420*4 00000000*4
		00028000*4 00020000*4 08020000*4 08008020*4 00000000*4
		08028020*4 08028020*4 08020000*4 08020020*4 00000000*4
	β_1	08028020*4 08028020*4 08020000*4 08020020*4 00000000*4
	γ_1	00041000*4 00041000*4 00041000*4 00041000*4 00000000*4
	λ_1	00041000*4
2	γ_2	08028020*4 08028020*4 08020000*4 08020020*4 00000000*4
	λ_2	08028020*4

The discussion of this section also indicates that the trails we find are superior to the ones presented in [2] in terms of both correlation and span. Moreover, since given a trail found in this paper, we can derive another trail with the same correlation by rotationally shift the masks within words by a common offset, we can identify the shifting offset minimizing the number of trailing zeros in the masks of the last block, which may further reduce the size of the common prefix. For example, by shifting the trail of MORUS-640-1280 shown in Table 6, we obtain a trail shown in Table 10 requiring only 481-bit common prefix when used in an attack.

To take it one step further, the positions of the identical message blocks required in the attack do not need to be located at the beginning. A common suffix works as well as a common prefix, and any four consecutive common blocks work.

Table 10: A linear trail of MORUS-640 with correlation 2^{-38}

Round		Linear masks
0	α_0	00004000*4 00004000*4 00000000*4 00004000*4 00000000*4
		00080000*4 00000000*4 00000000*4 00000000*4 00000000*4
		00080000*4 00000000*4 00000000*4 00000000*4 00000000*4
		00080000*4 00000000*4 00000000*4 00000000*4 00000000*4
		00080000*4 00000000*4 00000000*4 00000000*4 00000000*4
	β_0	00080000*4 00000000*4 00000000*4 00000000*4 00000000*4
	γ_0	00004000*4 00004000*4 00000000*4 00004000*4 00000000*4
	λ_0	00004000*4
1	α_1	08002000*4 08082000*4 00000000*4 08082000*4 00000000*4
		00040001*4 00080000*4 00000000*4 00080000*4 00000000*4
		00040001*4 00040000*4 00000000*4 00000000*4 00080000*4
		00040001*4 00040000*4 00000000*4 00000000*4 00080000*4
		00040001*4 00040000*4 00000000*4 00000000*4 00080000*4
	β_1	000c0001*4 000c0000*4 00080000*4 00000000*4 00000001*4
	γ_1	08082000*4 08082000*4 00000000*4 08082000*4 00000000*4
	λ_1	08082000*4
2	α_2	04000000*4 04000001*4 00000000*4 04000000*4 00000001*4
		80000000*4 00000001*4 00000000*4 00000000*4 00000001*4
		80000000*4 80000000*4 00000000*4 00000000*4 00000000*4
		80000000*4 80000000*4 00000000*4 00000000*4 00000000*4
		80000000*4 80000000*4 00000000*4 00000000*4 00000000*4
	β_2	80000000*4 80000000*4 00000000*4 00000000*4 00000000*4
	γ_2	040c0001*4 040c0001*4 00080000*4 04000000*4 00000000*4
	λ_2	040c0001*4
3	γ_3	80000000*4 80000000*4 00000000*4 00000000*4 00000000*4
	λ_3	80000000*4

7 Conclusion and Open Problems

In this work, we propose a polynomial-time algorithm for computing the correlation of a quadratic Boolean function based on its disjoint quadratic form. This method is employed to determine the correlations of the linear trails of

MiniMORUS and MORUS we find by solving MILP problems derived from a generic helper model for MORUS-like key-stream generators.

As a result, a set of trails involving four blocks of ciphertext with correlation 2^{-38} is identified for all versions of full MORUS, which leads to the first distinguishing and message-recovery attacks on MORUS-640-128 and MORUS-1280-128. We also observe that the condition specified in [2] to launch the attacks can be relaxed, and this relaxation shows that our trails are superior to those presented in previous work not only in terms of correlation, but also in terms of the numbers of ciphertext blocks involved.

At this point, it is natural to ask some open questions. Firstly, is it possible to compute the correlation of Boolean functions with degrees higher than two efficiently? We believe that an efficient algorithm solving this problem would have a significant effect for cryptanalysis. Secondly, can we find good trails for MORUS which are not rotationally invariant?

Acknowledgment. The authors thank the anonymous reviewers for many helpful comments. The work is supported by the National Key R&D Program of China (Grant No. 2018YFB0804402), the Chinese Major Program of National Cryptography Development Foundation (Grant No. MMJJ20180102), the National Natural Science Foundation of China (61732021, 61802400, 61772519, 61802399), and the Youth Innovation Promotion Association of Chinese Academy of Sciences. Chaoyun Li is supported by the Research Council KU Leuven: C16/15/058, OT/13/071, and by European Union's Horizon 2020 research and innovation programme (No. H2020-MSCA-ITN-2014-643161 ECRYPT-NET).

References

1. AlFardan, N.J., Bernstein, D.J., Paterson, K.G., Poettering, B., Schuldt, J.C.N.: On the security of RC4 in TLS. In: Proceedings of the 22th USENIX Security Symposium, Washington, DC, USA, August 14-16, 2013. pp. 305–320 (2013)
2. Ashur, T., Eichlseder, M., Lauridsen, M.M., Leurent, G., Minaud, B., Rotella, Y., Sasaki, Y., Viguier, B.: Cryptanalysis of MORUS. In: Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part II. pp. 35–64 (2018)
3. Bar-On, A., Dunkelman, O., Keller, N., Weizman, A.: DLCT: A new tool for differential-linear cryptanalysis. IACR Cryptology ePrint Archive **2019**, 256 (2019), https://eprint.iacr.org/2019/256, accepted to EUROCRYPT 2019
4. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings. pp. 531–545 (2000)
5. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. J. Cryptology **21**(4), 469–491 (2008)
6. Biryukov, A., Cannière, C.D., Quisquater, M.: On multiple linear approximations. In: Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptol-

ogyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. pp. 1–22 (2004)
7. CAESAR: Call for Submission: http://competitions.cr.yp.to/
8. Dobraunig, C., Eichlseder, M., Mendel, F.: Heuristic tool for linear cryptanalysis with applications to CAESAR candidates. In: Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II. pp. 490–509 (2015)
9. Domonkos, T.P., Lueg, L.: Taking a different approach to attack WPA2-AES, or the born of the CCMP known-plain-text attack (2010), https://www.hwsw.hu/kepek/hirek/2011/05/wpa2aes_ccmp_known_plaintext.pdf
10. Duong, T., Rizzo, J.: Here come the ⊕ ninjas. Ekoparty (2011)
11. Dwivedi, A.D., Kloucek, M., Morawiecki, P., Nikolic, I., Pieprzyk, J., Wójtowicz, S.: SAT-based cryptanalysis of authenticated ciphers from the CAESAR competition. In: Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) - Volume 4: SECRYPT, Madrid, Spain, July 24-26, 2017. pp. 237–246 (2017)
12. Dwivedi, A.D., Morawiecki, P., Wójtowicz, S.: Differential and rotational cryptanalysis of round-reduced MORUS. In: Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) - Volume 4: SECRYPT, Madrid, Spain, July 24-26, 2017. pp. 275–284 (2017)
13. Early Symmetric Crypto workshop (ESC 2013): https://www.cryptolux.org/mediawiki-esc2013/index.php/ESC_2013
14. Fu, K., Wang, M., Guo, Y., Sun, S., Hu, L.: MILP-based automatic search algorithms for differential and linear trails for Speck. In: Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers. pp. 268–288 (2016)
15. Jr., B.S.K., Robshaw, M.J.B.: Linear cryptanalysis using multiple approximations. In: Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings. pp. 26–39 (1994)
16. Jr., B.S.K., Robshaw, M.J.B.: Linear cryptanalysis using multiple approximations and FEAL. In: Fast Software Encryption: Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings. pp. 249–264 (1994)
17. Kales, D., Eichlseder, M., Mendel, F.: Note on the robustness of CAESAR candidates. IACR Cryptology ePrint Archive **2017**, 1137 (2017), http://eprint.iacr.org/2017/1137
18. Langford, S.K., Hellman, M.E.: Differential-linear cryptanalysis. In: Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings. pp. 17–25 (1994)
19. Li, Y., Wang, M.: Cryptanalysis of MORUS. Des. Codes Cryptography (2018), https://doi.org/10.1007/s10623-018-0501-6
20. Mantin, I., Shamir, A.: A practical attack on broadcast RC4. In: Fast Software Encryption, 8th International Workshop, FSE 2001 Yokohama, Japan, April 2-4, 2001, Revised Papers. pp. 152–164 (2001)
21. Mantin, I., Shamir, A.: A practical attack on broadcast RC4. In: Fast Software Encryption – FSE 2001. LNCS, vol. 2355, pp. 152–164. Springer (2001)
22. Matsui, M.: Linear cryptanalysis method for DES cipher. In: Helleseth, T. (ed.) Advances in Cryptology – EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer (1993)

23. Matsui, M.: On correlation between the order of S-boxes and the strength of DES. In: Advances in Cryptology–EUROCRYPT 1994. pp. 366–375. Springer (1995)
24. Mileva, A., Dimitrova, V., Velichkov, V.: Analysis of the authenticated cipher MORUS (v1). In: Cryptography and Information Security in the Balkans - Second International Conference, BalkanCryptSec 2015, Koper, Slovenia, September 3-4, 2015, Revised Selected Papers. pp. 45–59 (2015)
25. Minaud, B.: Linear biases in AEGIS keystream. In: Joux, A., Youssef, A.M. (eds.) Selected Areas in Cryptography – SAC 2014. LNCS, vol. 8781, pp. 290–305. Springer (2014)
26. Rogaway, P.: Authenticated-encryption with associated-data. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS 2002, Washington, DC, USA, November 18-22, 2002. pp. 98–107 (2002)
27. Rogaway, P.: Nonce-based symmetric encryption. In: Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers. pp. 348–359 (2004)
28. Rogaway, P., Shrimpton, T.: A provable-security treatment of the key-wrap problem. In: Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings. pp. 373–390 (2006)
29. Salam, M.I., Simpson, L., Bartlett, H., Dawson, E., Pieprzyk, J., Wong, K.K.: Investigating cube attacks on the authenticated encryption stream cipher MORUS. In: 2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, Australia, August 1-4, 2017. pp. 961–966 (2017)
30. Shi, T., Guan, J., Li, J., Zhang, P.: Improved collision cryptanalysis of authenticated cipher MORUS. Artificial Intelligence and Industrial Engineering–AIIE pp. 429–432 (2016)
31. Strassen, V.: Gaussian elimination is not optimal. Numerische Mathematik **13**(4), 354–356 (1969)
32. Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I. pp. 158–178 (2014)
33. Tews, E., Weinmann, R., Pyshkin, A.: Breaking 104-bit WEP in less than 60 seconds. In: Information Security Applications, 8th International Workshop, WISA 2007, Jeju Island, Korea, August 27-29, 2007, Revised Selected Papers. pp. 188–202 (2007)
34. Todo, Y., Isobe, T., Meier, W., Aoki, K., Zhang, B.: Fast correlation attack revisited - cryptanalysis on full Grain-128a, Grain-128, and Grain-v1. In: Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II. pp. 129–159 (2018)
35. Vaudenay, S., Vizár, D.: Under pressure: Security of CAESAR candidates beyond their guarantees. IACR Cryptology ePrint Archive **2017**, 1147 (2017), http://eprint.iacr.org/2017/1147
36. Wu, H., Huang, T.: The authenticated cipher MORUS (v2). Submission to CAESAR: Competition for Authenticated Encryption. Security, Applicability, and Robustness (Round 3 and Finalist) (2016), https://competitions.cr.yp.to/round3/morusv2.pdf