New General Framework for Algebraic Degree Evaluation of NFSR-Based Cryptosystems

Lin Ding† and Zheng Wu

PLA SSF Information Engneering University, Zhengzhou 450001, China
{dinglin_cipher@163.com}

Abstract. At CRYPTO 2017, Liu presented a general framework of iterative estimation of algebraic degree for NFSR-based cryptosystems, by exploiting a technique, called *numeric mapping*, and gave distinguishing attacks on Trivium-like ciphers, including Trivium, Kreyvium and TriviA-SC. This paper aims at further investigating algebraic degree estimation of NFSR-based cryptosystems from a new perspective. A new general framework for algebraic degree estimation of NFSR-based cryptosystems is formalized to exploit a new way of constructing distinguishing attacks. This illustrates that our new framework is more accurate than Liu's when estimating the upper bound on algebraic degree of NFSR-based cryptosystems. As result, the best known attack on the full simplified variant of TriviA-SC v2 is presented.

Keywords: Cryptanalysis; Nonlinear feedback shift register; Distinguishing attack; Trivium; Kreyvium; TriviA-SC.

1 Introduction

In recent years, for constrained environments like RFID tags or sensor networks, a number of lightweight cryptographic primitives have been developed to provide security and privacy. Nonlinear Feedback Shift Register (NFSR) is widely used in the design of modern lightweight ciphers, such as Trivium [1], Grain v1 [2] and MICKEY 2.0 [3] which have been selected in the eSTREAM [4] portfolio of promising stream ciphers for small hardware, the authenticated cipher ACORN v3 [5] which has been selected as one of seven finalists in the CAESAR competition, the block cipher family KATAN/KTANTAN [6], and the hash function family Quark [7, 8].

The well-known Trivium stream cipher, designed by De Cannière and Preneel in 2005, is a bit-oriented stream cipher in the eSTREAM project portfolio for hardware implementation, and has an exceptional structure which leads to good performance in both hardware and software. It has been studied extensively and shows good resistance to cryptanalysis, even after more than a decade of effort by cryptanalysts. Inspired by Trivium, some Trivium-like ciphers have been successfully developed, e.g., Kreyvium [9] developed at FSE 2016 for the efficient homomorphic-ciphertext compression and TriviA-SC [10, 11] developed as a component of the authenticated encryption cipher TriviA-ck. Trivium uses

an 80-bit key and an 80-bit IV, while Kreyvium and TriviA-SC both use a 128-bit key and a 128-bit IV. These three ciphers all have 1152 rounds of initialization.

At CRYPTO 2017, Liu [12] presented a general framework of iterative estimation of algebraic degree for NFSR-based cryptosystems, by exploiting a new technique, called *numeric mapping*, and gave distinguishing attacks on Trivium-like ciphers, including Trivium, Kreyvium and TriviA-SC. The key idea is based on a simple fact. The advantage of this method is that it has linear time complexity and needs a negligible amount of memory. Nevertheless, the estimation bias of *numeric mapping* probably becomes larger as the number of iterated rounds increases, this estimation method still requires to be further investigated. In this work, Liu presented distinguishing attacks on 1035-round TriviA-SC v1, 1047-round TriviA-SC v2 and the full simplified variant of TriviA-SC, with time complexities of 2^{63}, 2^{61} and 2^{63}, respectively. After then, some new applications of the numeric mapping technique to other NFSR-based ciphers were published in [13–16].

Our Contributions. In this paper, we focus on formalizing a new general framework for algebraic degree evaluation of NFSR-based cryptosystems, to provide a new way of constructing distinguishing attacks. We first introduce a new notion, called *algebraic degree tuple*, which describes the algebraic degree of a multivariate Boolean function in each one of all variables. Based on this notion, a new technique, called *composite numeric mapping*, is proposed, which gives a new general idea for iteratively estimating the upper bound on algebraic degree of an NFSR. We prove that *composite numeric mapping* is at least as good as *numeric mapping*, and most likely better than it demonstrated by applications. Then a new general framework for algebraic degree estimation of NFSR-based cryptosystems is formalized, and an efficient algorithm is proposed and applied to Trivium-like ciphers. The effectiveness and accuracy of our algorithm is confirmed by the experimental results. More importantly, to the best of our knowledge, this is the first time that algebraic degree tuple is defined and used to exploit new cryptanalytic techniques, which gives a new view on cryptanalysis of NFSR-based cryptosystems. Our new framework is also potentially useful in the future applications to other cryptosystems that are not built on NFSR.

For a NFSR-based cryptosystem, our algorithm can give an upper bound on algebraic degree of any one internal state bit or output bit over a given set of initial input variables with any size, e.g., all the key and IV bits, or all the IV bits. It has practical time complexity and requires a negligible amount of memory. By using our algorithm, we first investigate the mixing efficiency of Trivium-like ciphers, when taking all the key and IV bits as initial input variables. The results show that the maximum numbers of initialization rounds of Kreyvium, TriviA-SC v1 and TriviA-SC v2 such that the generated keystream bit does not achieve maximum algebraic degree are at least 983, 1109 and 1110 (out of 1152), rather than 982, 1108 and 1108 obtained in [12], respectively. When taking all the IV bits as initial input variables, the result shows that the maximum number of initialization rounds of TriviA-SC v2 such that the generated keystream bit does not achieve maximum algebraic degree is at least 988 (out of 1152), rather

than 987 obtained in [12]. In other cases, although we do not improve [12] in terms of the number of initialization rounds, tighter upper bounds on algebraic degree are mostly obtained. Take 793 rounds of Trivium as example, when taking all the IV bits as initial input variables, the upper bound on algebraic degree of the first output bit evaluated by our algorithm is 78, which is tighter than 79 by [12]. There results show that our new framework is more accurate than [12] when estimating the upper bound on algebraic degree of NFSR-based cryptosystems. All the results above are obtained on a common PC with 2.5 GHz Intel Pentium 4 processor within one second.

When taking a subset of all the IV bits as initial input variables, we apply our new framework to Trivium-like ciphers. As results, some new cubes which are as good as the results of [12] and can not be found by [12] are obtained. As for the full simplified variant of TriviA-SC v2, new distinguishing attack is found with time complexity of 2^{59}, which is the best known attack on the cipher. The result is listed in Table 1, and comparisons with previous works are made. It further illustrates that our new framework is more accurate than [12] when estimating the upper bound on algebraic degree of NFSR-based cryptosystems.

Table 1. Attacks on simplified variant of TriviA-SC v2

Cipher	# Rounds	Attack	Time compleixity	Reference
Simplified variant of TriviA-SC v2	Full	Distinguishing attack	2^{120}	[17]
	Full	Distinguishing attack	2^{63}	[12]
	Full	**Distinguishing attack**	$\mathbf{2^{59}}$	**Sect. 3.2**

To verify these cryptanalytic results, we make an amount of experiments on round reduced variants of Trivium-like ciphers. The experimental results show that our distinguishing attacks are always consistent with our evaluated results. They are strong evidences of high accuracy of our new framework. To facilitate the reader to verify our results, the supplementary materials are submitted together with our paper. They consist of the experimental results obtained by applying our algorithm to Trivium-like ciphers, including all the results on Trivium, Kreyvium, TriviA-SC v1 and TriviA-SC v2.

This paper is organized as follows. A new general framework for algebraic degree evaluation of NFSR-Based cryptosystems is formalized in Section 2. In Sections 3, algebraic degree tuple evaluations of Trivium-like ciphers are given as applications to prove the effectiveness of our new framework. The paper is concluded in Section 4.

2 New General Framework for Algebraic Degree Evaluation of NFSR-Based Cryptosystems

2.1 A New Notion: *Algebraic Degree Tuple*

Denote $\mathbb{F}_2^n$ the n-dimension vector space over $\mathbb{F}_2$. Let $\mathbb{B}_n$ be the set of all functions mapping $\mathbb{F}_2^n$ to $\mathbb{F}_2$, and let $f \in \mathbb{B}_n$. The Algebraic Normal Form (ANF) of given Boolean function f over variables $x_1, x_2, \cdots, x_n$ can be uniquely expressed as

$$f(x_1, x_2, \cdots, x_n) = \bigoplus_{c=(c_1,c_2,\cdots,c_n)\in\mathbb{F}_2^n} a_c \prod_{i=1}^{n} {x_i}^{c_i}$$

where the coefficient a_c is a constant in $\mathbb{F}_2$, and c_i denotes the i-th digit of the binary encoding of c (and so the sum spans all monomials in $x_1, x_2, \cdots, x_n$). The algebraic degree of f, denoted by $\deg(f)$, is defined as $\max\{wt(c) \,|a_c = 1\}$, where $wt(c)$ is the Hamming weight of c. Thus, for a multivariate Boolean function, the degree of a term is the sum of the exponents of the variables in the term, and then the algebraic degree of the multivariate Boolean function is the maximum of the degrees of all terms in the Boolean function. Now, we define a new notion, called *univariate algebraic degree*, which describes the algebraic degree of a multivariate Boolean function in one of all variables.

Definition 1. Let $f(X) = \bigoplus_{c=(c_1,c_2,\cdots,c_n)\in\mathbb{F}_2^n} a_c \prod_{i=1}^{n} {x_i}^{c_i}$ be a multivariate Boolean function over n variables $X = (x_1, x_2, \cdots, x_n)$, the algebraic degree of f in one variable $x_i\,(i = 1, 2, \cdots, n)$, called ***univariate algebraic degree***, is denoted by $\deg(f, x_i)$ and defined by

$$\deg(f, x_i) = \max_{a_c=1}\{wt(c) \,|c_i = 1\}$$

In particular, denote $\deg(0, x_i) = -\infty$ and $\deg(f, x_i) = 0$ if the variable x_i does not appear in nonzero f.

Example 1. Let $x_t = x_{t-2}x_{t-7} \oplus x_{t-4}x_{t-5} \oplus x_{t-8}\,(t \geq 9)$ be the update function of an NFSR with size 8. Then, iteratively compute

$$\begin{aligned}
&x_9 = x_2x_7 \oplus x_4x_5 \oplus x_1,\\
&x_{11} = x_2x_4x_7 \oplus x_1x_4 \oplus x_4x_5 \oplus x_6x_7 \oplus x_3,\\
&x_{12} = x_3x_5x_8 \oplus x_2x_5 \oplus x_5x_6 \oplus x_7x_8 \oplus x_4,\\
&x_{14} = x_2x_3x_7x_8 \oplus x_2x_5x_6x_7 \oplus x_3x_4x_5x_8 \oplus x_3x_5x_7x_8 \oplus x_1x_3x_8 \oplus x_1x_5x_6 \oplus\\
&\qquad x_2x_4x_5 \oplus x_2x_5x_7 \oplus x_4x_5x_6 \oplus x_5x_6x_7 \oplus x_1x_2 \oplus x_2x_7 \oplus x_4x_7 \oplus x_7x_8 \oplus x_6
\end{aligned}$$

It is easy to see that $\deg(x_9, x_1) = 1$, $\deg(x_{11}, x_1) = 2$, $\deg(x_{12}, x_1) = 0$, $\deg(x_{14}, x_1) = 3$ in Example 1.

Based on Definition 1, we define a new notion, called *algebraic degree tuple* as follows.

Definition 2. Let $f(X) = \underset{c=(c_1,c_2,\cdots,c_n)\in\mathbb{F}_2^n}{\oplus} a_c \prod_{i=1}^{n} {x_i}^{c_i}$ be a multivariate Boolean function over n variables $X = (x_1, x_2, \cdots, x_n)$, the ***algebraic degree tuple*** of f, denoted by $Tdeg(f, X)$, is defined by

$$Tdeg(f, X) = (\deg(f, x_1), \deg(f, x_2), \cdots, \deg(f, x_n))$$

Obviously, the algebraic degree of f is equal to the highest univariate numeric degree in the algebraic degree tuple $Tdeg(f, X)$, i.e.,

$$deg(f) = \max\{\deg(f, x_1), \deg(f, x_2), \cdots, \deg(f, x_n)\}$$

Recall Example 1, it is easy to see that $\deg(x_{14}) = 4$, while $Tdeg(x_{14}, X) = (3, 4, 4, 4, 4, 4, 4, 4)$. In fact, the algebraic degree of a multivariate Boolean function is the same as the degree of its term or terms having the highest degree and non-zero coefficient. However, in one variable, its corresponding univariate algebraic degree does not necessarily achieve the highest degree. Thus, it is clear that the multivariate Boolean function can be better characterized by its algebraic degree tuple than by the algebraic degree.

2.2 A New Technique: *Composite Numeric Mapping*

In this subsection, a new technique, called *composite numeric mapping*, is introduced, and a new general idea for iteratively estimating the upper bound on algebraic degree of an NFSR is given. It is a theoretical tool for algebraic degree estimation.

Before introducing our new technique, we have to define new computation models of upper bound on algebraic degree for $\oplus$ and $\cdot$ operations, since they are two fundamental operations in most NFSR-based cryptosystems, where the XOR of two internal state bits is computed in $\oplus$ and the AND of two internal state bits is computed in $\cdot$. Until now, the most classical computation models of upper bound on algebraic degree for $\oplus$ and $\cdot$ operations are $\deg(f \oplus g) \leq \max\{\deg(f), \deg(g)\}$ and $\deg(f \cdot g) \leq \deg(f) + \deg(g)$, respectively. They are the foundation of *numeric mapping* in [12]. More specifically, under *numeric mapping*, the algebraic degree of $f \oplus g$ has an upper bound $\mathbf{DEG}(f \oplus g) = \max\{\deg(f), \deg(g)\}$, and the algebraic degree of $f \cdot g$ has an upper bound $\mathbf{DEG}(f \cdot g) = \deg(f) + \deg(g)$. For simplicity, $\mathbf{DEG}(h)$ is called a super *numeric degree* of the composite function h in [12]. However, as for algebraic degree tuple, the trivial computation models have to be modified. The new computation models are described as follows.

Proposition 1. (New Computation Model of Upper Bound on Algebraic Degree Tuple for $\oplus$, denoted by $\widetilde{\oplus}$) Let $f(X)$ and $g(X)$ be two multivariate Boolean functions over n variables $X = (x_1, x_2, \cdots, x_n)$, $Tdeg\left(f\widetilde{\oplus}g, X\right) = \left(\deg\left(f\widetilde{\oplus}g, x_1\right), \deg\left(f\widetilde{\oplus}g, x_2\right), \cdots, \deg\left(f\widetilde{\oplus}g, x_n\right)\right)$ gives an upper bound on algebraic degree tuple $Tdeg(f \oplus g, X)$, when the computation of $\deg\left(f\widetilde{\oplus}g, x_i\right)$ follows $\deg\left(f\widetilde{\oplus}g, x_i\right) = \max\{\deg(f, x_i), \deg(g, x_i)\}$.

It is clear to see that

$$\begin{aligned}\deg(f \oplus g, x_i) &\leq \max\{\deg(f, x_i), \deg(g, x_i)\} \\ &= \deg\left(f \widetilde{\oplus} g, x_i\right)\end{aligned}$$

Since $\deg(f, x_i) \leq \deg(f)$ and $\deg(g, x_i) \leq \deg(g)$, we have

$$\begin{aligned}\deg\left(f \widetilde{\oplus} g, x_i\right) &= \max\{\deg(f, x_i), \deg(g, x_i)\} \\ &\leq \max\{\deg(f), \deg(g)\} = \mathbf{DEG}(f \oplus g)\end{aligned}$$

Hence, the new computation model of upper bound on algebraic degree tuple for $\oplus$ operation ensures that the exact algebraic degree tuple of $Tdeg(f \oplus g, X)$ is always less than or equal to $Tdeg\left(f \widetilde{\oplus} g, X\right)$, i.e., $Tdeg(f \oplus g, X) \preccurlyeq Tdeg\left(f \widetilde{\oplus} g, X\right)$, where $Tdeg(f \oplus g, X) \preccurlyeq Tdeg\left(f \widetilde{\oplus} g, X\right)$ if $\deg(f \oplus g, x_i) \leq \deg\left(f \widetilde{\oplus} g, x_i\right)$ for all $1 \leq i \leq n$. Simultaneously, it also ensures that $Tdeg\left(f \widetilde{\oplus} g, x_i\right) \leq \mathbf{DEG}(f \oplus g)$ for all $1 \leq i \leq n$.

Denote $u_{\max} = \max\{\deg(f, x_i) \,|\, 1 \leq i \leq n\}$ and $v_{\max} = \max\{\deg(g, x_i) \,|\, 1 \leq i \leq n\}$. The new computation model of upper bound on algebraic degree tuple for $\cdot$ operation is described as follows.

Proposition 2. (New Computation Model of Upper Bound on Algebraic Degree Tuple for $\cdot$, denoted by $\widetilde{\cdot}$). Let $f(X)$ and $g(X)$ be two multivariate Boolean functions over n variables $X = (x_1, x_2, \cdots, x_n)$, $Tdeg(f \widetilde{\cdot} g, X) = (\deg(f \widetilde{\cdot} g, x_1), \deg(f \widetilde{\cdot} g, x_2), \cdots, \deg(f \widetilde{\cdot} g, x_n))$ gives an upper bound on algebraic degree tuple $Tdeg(f \cdot g, X)$, when the computation of $\deg(f \widetilde{\cdot} g, x_i)$ follows Table 2.

Table 2. The computation model of $\deg(f \widetilde{\cdot} g, x_i)$

-	$\deg(g, x_i) = -\infty$	$\deg(g, x_i) = 0$	$\deg(g, x_i) = v\,(v > 0)$
$\deg(f, x_i) = -\infty$	$-\infty$	$-\infty$	$-\infty$
$\deg(f, x_i) = 0$	$-\infty$	0	$v + u_{\max}$
$\deg(f, x_i) = u\,(u > 0)$	$-\infty$	$u + v_{\max}$	$\max\{u + v_{\max}, v + u_{\max}\}$

All cases in Table 2 can be easily checked, except one case when $\deg(f, x_i) = u\,(u > 0)$ and $\deg(g, x_i) = v\,(v > 0)$ are simultaneously satisfied. To verify the correctness of this case, a corollary is given and proved as follows.

Corollary 1. Let $f(X)$ and $g(X)$ be two multivariate Boolean functions over n variables $X = (x_1, x_2, \cdots, x_n)$, denote $u_{\max} = \max\{\deg(f, x_i) \,|\, 1 \leq i \leq n\}$ and $v_{\max} = \max\{\deg(g, x_i) \,|\, 1 \leq i \leq n\}$, if $\deg(f, x_i) = u\,(u > 0)$ and $\deg(g, x_i) = v\,(v > 0)$, then $\deg(f \cdot g, x_i) \leq \max\{u + v_{\max}, v + u_{\max}\}$.

Proof. Rewrite the function f as $f = x_i \cdot f_0 \oplus f_1$, where f_0 and f_1 denotes two derived functions over the remaining $n - 1$ variables except x_i with $\deg(f_0) =$

$u-1$ and $\deg(f_1) \leq u_{\max}$. Similarly, rewrite the function g as $g = x_i \cdot g_0 \oplus g_1$, where g_0 denotes two derived functions over the remaining $n-1$ variables except x_i with $\deg(g_0) = v-1$ and $\deg(g_1) \leq v_{\max}$. Thus, we have

$$f \cdot g = (x_i \cdot f_0 \oplus f_1) \cdot (x_i \cdot g_0 \oplus g_1) = x_i \cdot f_0 \cdot g_0 \oplus x_i \cdot f_0 \cdot g_1 \oplus x_i \cdot f_1 \cdot g_0 \oplus f_1 \cdot g_1$$

It is easy to see that

$$\begin{gathered}\deg(x_i \cdot f_0 \cdot g_0, x_i) \leq 1+u-1+v-1 = u+v-1 \\ \deg(x_i \cdot f_0 \cdot g_1, x_i) \leq 1+u-1+v_{\max} = u+v_{\max} \\ \deg(x_i \cdot f_1 \cdot g_0, x_i) \leq 1+u_{\max}+v-1 = u_{\max}+v \\ \deg(f_1 \cdot g_1, x_i) \leq 0\end{gathered}$$

Hence, we have

$$\begin{aligned}&\deg(f \cdot g, x_i) \\ &\leq \max\{\deg(x_i \cdot f_0 \cdot g_0, x_i), \deg(x_i \cdot f_0 \cdot g_1, x_i), \deg(x_i \cdot f_1 \cdot g_0, x_i), \deg(f_1 \cdot g_1, x_i)\} \\ &= \max\{u+v_{\max}, v+u_{\max}\}\end{aligned}$$

Since $deg(f) = u_{\max} = \max\{\deg(f, x_i) | 1 \leq i \leq n\}$ and $deg(g) = v_{\max} = \max\{\deg(g, x_i) | 1 \leq i \leq n\}$, we have

$$\begin{aligned}\deg(f \tilde{\cdot} g, x_i) &= \max\{u+v_{\max}, v+u_{\max}\} \\ &\leq \deg(f) + \deg(g) = \mathbf{DEG}(f \cdot g)\end{aligned}$$

Thus, the new computation model of upper bound on algebraic degree tuple for $\cdot$ ensures that the exact algebraic degree tuple of $Tdeg(f \cdot g, X)$ is always less than or equal to $Tdeg(f \tilde{\cdot} g, X)$, i.e., $Tdeg(f \cdot g, X) \preccurlyeq Tdeg(f \tilde{\cdot} g, X)$, where $Tdeg(f \cdot g, X) \preccurlyeq Tdeg(f \tilde{\cdot} g, X)$ if $\deg(f \cdot g, x_i) \leq \deg(f \tilde{\cdot} g, x_i)$ for all $1 \leq i \leq n$. Simultaneously, it also ensures that $Tdeg(f \tilde{\cdot} g, x_i) \leq \mathbf{DEG}(f \cdot g)$ for all $1 \leq i \leq n$.

Based on these new computation models of upper bound on algebraic degree tuple for $\oplus$ and $\cdot$ operations, the upper bound on algebraic degree tuple of a composite function can be estimated by using the following new technique, called *composite numeric mapping*, and then estimating the upper bound on algebraic degree tuples of an NFSR can be easily done by iterative computation.

Let $\mathbb{B}_n$ be the set of all functions mapping $\mathbb{F}_2^n$ to $\mathbb{F}_2$, let $f \in \mathbb{F}_2^n$ and $f(S) = \bigoplus_{c=(c_1,c_2,\cdots,c_m)\in\mathbb{F}_2^m} a_c \prod_{i=1}^{m} {s_i}^{c_i}$ be a multivariate Boolean function over m internal variables $S = (s_1, s_2, \cdots, s_m)$, let $X = (x_1, x_2, \cdots, x_n)$ be n initial input variables, let $D = (d_{i,j})_{i=1,2,\cdots,m, j=1,2,\cdots,n} \in \mathbb{Z}^{m\times n}$ satisfying $d_{i,j} \geq \deg(s_i, x_j)$ for all $1 \leq i \leq m$ and $1 \leq j \leq n$, we define the following mapping, called ***composite numeric mapping*** and denoted by **CDEG**,

$$\begin{aligned}&\mathbf{CDEG} : \mathbb{B}_m \times \mathbb{Z}^{m\times n} \to \mathbb{Z}^n, \\ &\qquad (f, D) \mapsto Tdeg\left(\bigwidetilde{\bigoplus}_{a_c=1, c=(c_1,c_2,\cdots,c_m)\in\mathbb{F}_2^m} ({s_1}^{c_1} \tilde{\cdot} {s_2}^{c_2} \tilde{\cdot} \cdots \tilde{\cdot} {s_m}^{c_m}), X\right)\end{aligned}$$

$\mathbf{CDEG}(f, D)$ gives an upper bound on algebraic degree tuple $Tdeg(f, X)$, called *composite numeric degree tuple* of f over variables X and denoted by $\mathbf{CNDT}(f, X) = (\mathbf{CNDT}(f, x_1), \mathbf{CNDT}(f, x_2), \cdots, \mathbf{CNDT}(f, x_n)) = \mathbf{CDEG}(f, D)$. Hereinafter, $\mathbf{CNDT}(f, x_i)$ is called *univariate composite numeric degree* of f in one variable x_i for simplicity. Usually, we are more concerned with the smallest *univariate composite numeric degree* in the *composite numeric degree tuple* $\mathbf{CNDT}(f, X)$, denoted by $\mathbf{minCNDT}(f, X) = \min\{\mathbf{CNDT}(f, x_i), 1 \leq i \leq n\}$, since it implies the best distinguishing attack on NFSR-based cryptosystems.

As demonstrated above, $\deg(f \oplus g, x_i) \leq \deg\left(f \widetilde{\oplus} g, x_i\right)$ and $\deg(f \cdot g, x_i) \leq \deg(f \mathbin{\tilde{\cdot}} g, x_i)$ simultaneously hold for all $1 \leq i \leq n$, and then we certainly have $Tdeg(h, X) \preccurlyeq \mathbf{CNDT}(h, X)$, where $Tdeg(h, X) \preccurlyeq \mathbf{CNDT}(h, X)$ if $\deg(h, x_i) \leq \mathbf{CNDT}(h, x_i)$ for all $1 \leq i \leq n$. Similarly, since $\mathbf{CNDT}(f \oplus g, x_i) \leq \mathbf{DEG}(f \oplus g)$ and $\mathbf{CNDT}(f \cdot g, x_i) \leq \mathbf{DEG}(f \cdot g)$ simultaneously hold for all $1 \leq i \leq n$, and then we certainly have $\mathbf{CNDT}(h, x_i) \leq \mathbf{DEG}(h)$ for all $1 \leq i \leq n$.

Corollary 2. *The algebraic degree tuple of a multivariate Boolean function is always less than or equal to its composite numeric degree tuple. Simultaneously, each univariate composite numeric degree in the composite numeric degree tuple is always less than or equal to its numeric degree.*

Recall the Example 1. For $t = 16$, it is easy to compute $x_{16} = x_{14}x_9 \oplus x_{12}x_{11} \oplus x_8$. Then according to Definition 2, it has

$$Tdeg(x_{14}, X) = (\deg(x_{14}, x_1), \deg(x_{14}, x_2), \cdots, \deg(x_{14}, x_8)) = (3, 4, 4, 4, 4, 4, 4, 4)$$
$$Tdeg(x_9, X) = (\deg(x_9, x_1), \deg(x_9, x_2), \cdots, \deg(x_9, x_8)) = (1, 2, 0, 2, 2, 0, 2, 0)$$
$$Tdeg(x_{12}, X) = (\deg(x_{12}, x_1), \deg(x_{12}, x_2), \cdots, \deg(x_{12}, x_8)) = (0, 2, 3, 1, 3, 2, 2, 3)$$
$$Tdeg(x_{11}, X) = (\deg(x_{11}, x_1), \deg(x_{11}, x_2), \cdots, \deg(x_{11}, x_8)) = (2, 3, 1, 3, 2, 2, 3, 0)$$
$$Tdeg(x_8, X) = (\deg(x_8, x_1), \deg(x_8, x_2), \cdots, \deg(x_8, x_8)) = (0, 0, 0, 0, 0, 0, 0, 1)$$

Using *composite numeric mapping*, it has

$$\mathbf{CNDT}(x_{16}, X) = Tdeg\left((x_{14} \mathbin{\tilde{\cdot}} x_9) \mathbin{\widetilde{\oplus}} (x_{12} \mathbin{\tilde{\cdot}} x_{11}) \mathbin{\widetilde{\oplus}} x_8, X\right) = (5, 6, 6, 6, 6, 6, 6, 6)$$

It is easy to verify that $\mathbf{minCNDT}(x_{16}, X) = \mathbf{CNDT}(x_{16}, x_1) = 5$ and $\mathbf{CNDT}(x_{16}, x_1) < \deg(x_{16}) = 6$. This demonstrates that *composite numeric mapping* can exploit more useful information which is ignored by *numeric mapping*, and give more accurate estimation of the upper bound on the algebraic degree of the internal states of an NFSR.

2.3 A New General Framework for Algebraic Degree Evaluation of NFSR-Based Cryptosystems

A NFSR-based cryptosystem with internal state size of L usually consists of the update functions $G = (g_1, g_2, \cdots, g_L)$ and an output function f, where G are used to update the internal states and f is used to generate the output bit after a sufficient number of iterative rounds. By Corollary 2, the upper bound

on algebraic degree tuple of each internal state bit or output bit of NFSR-based cryptosystems can be estimated, described in Corollary 3.

Corollary 3. *Denote by X the initial input variables of a NFSR-based cryptosystem, and let $G = (g_1, g_2, \cdots, g_L)$ and f be the update functions and output function respectively. Then the algebraic degree tuples of the updated bits and output bit are respectively less than or equal to their composite numeric degree tuples, i.e., $Tdeg\,(g_j, X) \preccurlyeq \boldsymbol{CNDT}(g_j, X)$, $1 \le j \le L$ and $Tdeg\,(f, X) \preccurlyeq \boldsymbol{CNDT}(f, X)$. Simultaneously, each univariate composite numeric degree in the composite numeric degree tuples of the updated bits and output bit is always less than or equal to the corresponding numeric degree, i.e., $\boldsymbol{CNDT}(g_j, x_i) \le \boldsymbol{DEG}(g_j)$ and $\boldsymbol{CNDT}(f, x_i) \le \boldsymbol{DEG}(f)$ for all $1 \le i \le n$ and $1 \le j \le L$.*

As shown in Corollary 3, *composite numeric mapping* gives a new general idea for iteratively estimating the upper bound on algebraic degree tuple of NFSR-based cryptosystems, and is at least as good as *numeric mapping*, and most likely better than it. The iterative estimation procedure is depicted in Algorithm 1.

In this algorithm, $S^{(t)} = \left(s_1^{(t)}, s_2^{(t)}, \cdots, s_L^{(t)}\right)$ denotes the internal state at time t with size L. The update functions $G = (g_1, g_2, \cdots, g_L)$ is written as vectorial Boolean functions from $\mathbb{F}_2^L$ to $\mathbb{F}_2^L$, where a few bits of input are updated and the rest bits are shifted. **CNMDegEst** is a procedure for estimating the upper bound on algebraic degree tuple using *composite numeric mapping*. The algorithm gives a composite numeric degree tuple $\mathbf{CNDT}\,(f, X)$ as output, which is an upper bound on algebraic degree tuple $Tdeg\,(f, X)$ of the output function f. This is based on the fact that, $Tdeg\,(g_j, X) \preccurlyeq \mathbf{CNDT}\,(g_j, X)$, $1 \le j \le L$ and $Tdeg\,(f, X) \preccurlyeq \mathbf{CNDT}\,(f, X)$.

Algorithm 1. CNMDegEst: Iterative Estimation of Upper Bound on Algebraic Degree Tuple of NFSR-Based Cryptosystems Using *Composite Numeric Mapping*

Require: Given the ANFs of the initial state $S^{(0)} = \left(s_1^{(0)}, s_2^{(0)}, \cdots, s_L^{(0)}\right)$, the ANFs of the update functions $G = (g_1, g_2, \cdots, g_L)$ and the output function f, and the set of initial input variables $X = (x_1, x_2, \cdots, x_n)$.

1: Set $D^{(0)}$ to $\deg\left(S^{(0)}, X\right)$;
2: For t from 1 to N do:
3: Compute $\mathbf{CNDT}\left(g_j\left(S^{(t-1)}\right), X\right) \leftarrow \mathbf{CDEG}\left(g_j\left(S^{(t-1)}\right), D^{(t-1)}\right)$ for all $1 \le j \le L$;
4: $D^{(t)} \leftarrow \mathbf{CNDT}\left(G\left(S^{(t-1)}\right), X\right)$;
5: $\mathbf{CNDT}\,(f, X) \leftarrow \mathbf{CDEG}\left(f\left(S^{(N)}\right), D^{(N)}\right)$;
6: Return $\mathbf{CNDT}\,(f, X)$.

Complexity of Algorithm 1. Since the sizes of the ANFs of the update functions $G = (g_1, g_2, \cdots, g_L)$ and the output function f are constant and thus $\mathbf{CDEG}\left(g_j\left(S^{(t-1)}\right), D^{(t-1)}\right)$ can be calculated in constant time, the time complexity of Algorithm 1 mainly depends on the values of N and L. Therefore, Algorithm 1 has a time complexity of $\mathcal{O}(N \cdot L)$. In the algorithm, it requires to store $D^{(t)}$ for $t = 1, 2, \cdots, N$, which leads to a memory complexity of $\mathcal{O}(L \cdot n \cdot N)$. When $\mathbf{CNDT}(f, X)$ is obtained, zero-sum distinguishers can be easily constructed.

Actually, we are more concerned with the smallest univariate composite numeric degree in the outputted composite numeric degree tuple $\mathbf{CNDT}(f, X)$, i.e., $\mathbf{minCNDT}(f, X) = \min\{\mathbf{CNDT}(f, x_i), 1 \leq i \leq n\}$, since it implies the best distinguishing attack on NFSR-based cryptosystems with time complexity of $2^{1+\mathbf{minCNDT}(f,X)}$.

3 Algebraic Degree Tuple Evaluation of Trivium-Like Ciphers

In this section, we will refine and apply our new framework to Trivium-like ciphers, i.e., Trivium, Kreyvium and TriviA-SC to analyze the mixing efficiency of them and exploit distinguishing attacks on them.

3.1 The Algorithm for Algebraic Degree Tuple Estimation of Trivium-like Ciphers

In this subsection, we will present an efficient algorithm for algebraic degree tuple estimation of Trivium-like ciphers using *composite numeric mapping*, as depicted in Algorithm 2.

Algorithm 2. Algebraic Degree Tuple Estimation of Trivium-like Ciphers Using *Composite Numeric Mapping*

Require: Given the ANFs of the initial state $S^{(0)} = \left(A^{(0)}, B^{(0)}, C^{(0)}\right)$ (or $S^{(t)} = \left(A^{(t)}, B^{(t)}, C^{(t)}, K^*, IV^*\right)$), the ANFs of the update functions $G = (g_1, g_2, \cdots, g_L)$ and the keystream output function f, and the set of initial input variables $X = (x_1, x_2, \cdots, x_n)$.

1: Set $D^{(0)}$ to $\deg\left(S^{(0)}, X\right)$;
2: For t from 1 to N do:
3: For δ in $\{A, B, C\}$ do:
4: $\mathbf{CNDT}\left(g_\delta\left(S^{(t-1)}\right), X\right) \leftarrow \mathbf{CDEG}_{Tri}\left(g_\delta\left(S^{(t-1)}\right), D^{(t-1)}\right)$;
5: Set $D^{(t)}$ using $\mathbf{CNDT}\left(g_\delta\left(S^{(t-1)}\right), X\right)$ $(\delta \in \{A, B, C\})$ and $D^{(t-1)}$;
6: $\mathbf{CNDT}(f, X) \leftarrow \mathbf{CDEG}\left(f\left(S^{(N)}\right), D^{(N)}\right)$;
7: Return $\mathbf{CNDT}(f, X)$.

Algorithm 2 is an application of Algorithm 1 to Trivium-like ciphers. Thus, its main process is similar to Algorithm 1, except two points required to be highlighted due to the special structure of Trivium-like ciphers. Firstly, all of the update functions $G = (g_1, g_2, \cdots, g_L)$, which are used to update all L internal state bits, are shifting operations except three quadratic functions. In other words, $D^{(t)}$ can be easily set when $\mathbf{CNDT}\left(g_\delta\left(S^{(t-1)}\right), X\right)$ $(\delta \in \{A, B, C\})$ and $D^{(t-1)}$ are simultaneously obtained. Secondly, for estimating the upper bound on algebraic degree tuple more accurately, we exploit a new procedure $\mathbf{CDEG}_{Tri}$, a variant of **CDEG** specifically aiming at Trivium-like ciphers, to compute the composite numeric degree tuple of $g_\delta\left(S^{(t-1)}\right)$ $(\delta \in \{A, B, C\})$ over initial input variables X. Note that the basic idea of procedure $\mathbf{CDEG}_{Tri}$ is similar to Lemma 4 of [12]. However, $\mathbf{CDEG}_{Tri}$ is based on *composite numeric mapping*, rather than *numeric mapping* used in [12]. An instance of $\mathbf{CDEG}_{Tri}$ for $\delta = A$ is depicted in Algorithm 3. The other two cases, i.e., $\delta = B$ and $\delta = C$, are similar and given in Algorithm 4 and 5 respectively in Appendix. The procedure $\mathbf{CDEG}_{Tri}$ is based on simple derivations, and can be easily verified.

Take $\delta = A$ for example. For $t \geq r_C + 1$, it has

$$
\begin{aligned}
& g_A\left(S^{(t)}\right) \\
&= \left(c_{r_C}^{(t)} \cdot c_{r_C+1}^{(t)}\right) \oplus \mathcal{L}_A\left(S^{(t)}\right) \\
&= c_1^{(t-r_C+1)} \cdot c_1^{(t-r_C)} \oplus \mathcal{L}_A\left(S^{(t)}\right) \\
&= \left[\left(b_{r_B}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C)}\right) \oplus \mathcal{L}_C\left(S^{(t-r_C)}\right)\right] \cdot \left[\left(b_{r_B}^{(t-r_C-1)} \cdot b_{r_B+1}^{(t-r_C-1)}\right) \oplus \mathcal{L}_C\left(S^{(t-r_C-1)}\right)\right] \oplus \mathcal{L}_A\left(S^{(t)}\right) \\
&= \left[\left(b_{r_B}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C)}\right) \oplus \mathcal{L}_C\left(S^{(t-r_C)}\right)\right] \cdot \left[\left(b_{r_B+1}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C-1)}\right) \oplus \mathcal{L}_C\left(S^{(t-r_C-1)}\right)\right] \oplus \mathcal{L}_A\left(S^{(t)}\right) \\
&= \left(b_{r_B}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C-1)}\right) \oplus \left(b_{r_B}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C)} \cdot \mathcal{L}_C\left(S^{(t-r_C-1)}\right)\right) \\
&\quad \oplus \left(b_{r_B+1}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C-1)} \cdot \mathcal{L}_C\left(S^{(t-r_C)}\right)\right) \oplus \left(\mathcal{L}_C\left(S^{(t-r_C)}\right) \cdot \mathcal{L}_C\left(S^{(t-r_C-1)}\right)\right) \oplus \mathcal{L}_A\left(S^{(t)}\right)
\end{aligned}
$$

Algorithm 3 $\mathbf{CDEG}_{Tri}\left(g_\delta\left(S^{(t)}\right), D^{(t)}\right)$ for $\delta = A$

If $t < r_C + 1$ then:
  Return $\mathbf{CDEG}\left(\left(c_{r_C}^{(t)} \cdot c_{r_C+1}^{(t)}\right) \oplus \mathcal{L}_A\left(S^{(t)}\right), D^{(t)}\right)$;
If $t \geq r_C + 1$ then:
  Return $\mathbf{CDEG}\left(h, D^{(t)}\right)$, where $h = \left(b_{r_B}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C-1)}\right)$
  $\oplus \left(b_{r_B}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C)} \cdot \mathcal{L}_C\left(S^{(t-r_C-1)}\right)\right) \oplus \left(b_{r_B+1}^{(t-r_C)} \cdot b_{r_B+1}^{(t-r_C-1)} \cdot \mathcal{L}_C\left(S^{(t-r_C)}\right)\right)$
  $\oplus \left(\mathcal{L}_C\left(S^{(t-r_C)}\right) \cdot \mathcal{L}_C\left(S^{(t-r_C-1)}\right)\right) \oplus \mathcal{L}_A\left(S^{(t)}\right)$.

Algorithm 2 gives a composite numeric degree tuple $\mathbf{CNDT}\left(f, X\right)$ of the output function f after N rounds over initial input variables $X = (x_1, x_2, \cdots, x_n)$ as output, which gives upper bounds on all n univariate algebraic degrees of the first output bit after N rounds of a Trivium-like cipher. When $\mathbf{CNDT}\left(f, X\right)$ is obtained, zero-sum distinguishers can be easily constructed, and then a distinguishing attack with time complexity of $2^{1+\mathbf{minCNDT}(f,X)}$ can be obtained.

Similar to Algorithm 1, the time complexity of Algorithm 2 mainly depends on the values of N and L. Since all of the update functions $G = (g_1, g_2, \cdots, g_L)$ are shifting operations except three quadratic functions for Trivium-like ciphers, Algorithm 2 has a time complexity of $\mathcal{O}(N)$. Similar to Algorithm 1, Algorithm 2 requires to store $D^{(t)}$ for $t = 1, 2, \cdots, N$. Since the number of initial input variables is constant for a given Trivium-like cipher, it leads to a negligible memory complexity of $\mathcal{O}(N)$.

3.2 Experimental Results

In general, the key and the IV are mapped to the internal state of the cipher by an initialization function which consists of a well-chosen number of iterative rounds without generating keystream bits. The security of the initialization function relies on its mixing properties, i.e., each key and IV bit should affect each internal state bit in a complex way. If the mixing is not perfect, then the initialization function may be distinguished from a uniformly random Boolean function. By using our algorithm, we will investigate the mixing efficiency of Trivium-like ciphers.

Table 3. New lower bound on the maximum number of rounds of not achieving maximum degree for Trivium-like ciphers with all the key and IV bits as initial input variables ($X = (K, IV)$)

Cipher	# key + # IV	[12]		**This paper**	
		# Rounds	**DEG**(f)	# Rounds	**minCNDT** (f, X)
Trivium	160	907	158	907	**157**
Kreyvium	256	982	255	**983**	255
TriviA-SC v1	256	1108	254	**1109**	255
TriviA-SC v2	256	1108	254	**1110**	255

When Will the Key and IV Be Sufficiently Mixed? Taking all the key and IV bits as initial input variables X, we implement Algorithm 2 to Trivium-like ciphers including Trivium, Kreyvium and TriviA-SC. The results are listed in Table 3, and comparisons with previous results are made. The results show that the maximum numbers of initialization rounds of Kreyvium, TriviA-SC v1 and TriviA-SC v2 such that the generated keystream bit does not achieve maximum algebraic degree are at least 983, 1109 and 1110 (out of 1152), rather than 982, 1108 and 1108 obtained in [12], respectively. As for Trivium, although we do not improve the result of [12] in terms of the number of initialization rounds, tighter upper bound on algebraic degree is obtained. In detail, for 907 rounds of Trivium, the upper bound on algebraic degree of the first output bit evaluated by our algorithm is 157, which is tighter than 158 by [12]. There results show that our new framework is more accurate than [12] when estimating the upper bound on algebraic degree of NFSR-based cryptosystems. All results above are

obtained on a common PC with 2.5 GHz Intel Pentium 4 processor within one second.

When Will the IV Be Sufficiently Mixed? Taking all the IV bits as initial input variables, i.e., $X = IV$, we apply Algorithm 2 to Trivium-like ciphers, to give new distinguishing attacks on these ciphers in the chosen IV setting. Here, the key is taken as parameter, that is, $deg(k_i, X) = 0$ for any key bit k_i. This is consistent with a distinguisher in the setting of unknown key. The results are listed in Table 4, and comparisons with previous results are made. The result shows that the maximum number of initialization rounds of TriviA-SC v2 such that the generated keystream bit does not achieve maximum algebraic degree is at least 988 (out of 1152), rather than 987 obtained in [12]. In other cases except TriviA-SC v1, although we do not improve the results of [12] in terms of the number of initialization rounds, tighter upper bounds on algebraic degree are mostly obtained. For 793 rounds of Trivium, the upper bound on algebraic degree of the first output bit evaluated by our algorithm is 78, which is tighter than 79 by [12]. For 862 rounds of Kreyvium, the upper bound on algebraic degree of the first output bit evaluated by our algorithm is 126, which is tighter than 127 by [12]. There results also show that our new framework is more accurate than [12] when estimating the upper bound on algebraic degree of NFSR-based cryptosystems. All results above are obtained on a common PC within one second.

Table 4. New lower bound on the maximum number of rounds of not achieving maximum degree for Trivium-like ciphers with all IV bits as initial input variables ($X = IV$)

Cipher	# IV	[12]		**This paper**	
		# Rounds	**DEG**(f)	# Rounds	**minCNDT** (f, X)
Trivium	80	793	79	793	**78**
Kreyvium	128	862	127	862	**126**
TriviA-SC v1	128	987	126	987	126
TriviA-SC v2	128	987	126	**988**	127

Furthermore, similar to [12], we also take a subset of IV bits as initial input variables X, and apply Algorithm 2 to Trivium-like ciphers. We consider an exhaustive search on the sets of input variables X which have size of around half of $\#IV$ and contain no adjacent indexes. Using Algorithm 2, we have exhausted all the cubes of size $37 \leq n \leq 40$ for Trivium, which contain no adjacent indexes, within half an hour on a common PC. The amount of such cubes is $\sum_{n=37}^{40} \binom{81-n}{n} \approx 2^{25}$. As for Kreyvium and TriviA-SC, we have exhausted all the cubes of size $61 \leq n \leq 64$ for them, which contain no adjacent indexes, in a few hours on a common PC. The amount of such cubes is $\sum_{n=61}^{64} \binom{129-n}{n} \approx 2^{30}$. As results, some new cubes which are as good as the results of [12] and can not be found by [12] are obtained, as shown in Table 6 in Appendix.

Table 5. Distinguishing attacks on the full simplified variant of TriviA-SC v2

Cipher	# Rounds	Size of cube	Time compleixity	Reference
Simplified variant of TriviA-SC v2	Full	-	2^{120}	[17]
	Full	63	2^{63}	[12]
	Full	**59**	$\mathbf{2^{59}}$	**This paper**

As for the full simplified variants of TriviA-SC, we have exhausted all the cubes containing no adjacent indexes of size 60 in a week on a common PC. The amount of such cubes is $\binom{69}{60} \approx 2^{36}$. Then we have also exhausted approximately one twentieth of all the cubes containing no adjacent indexes of size 59 in half a month on a common PC. The amount of such cubes is $\binom{70}{59}/20 \approx 2^{37}$. As results, new distinguishing attack on the full simplified variant of TriviA-SC v2 is found with time complexity of 2^{59}, which improves the attack in [12] by a factor of 2^4 and is the best known attack on the cipher. The result is listed in Table 5, and comparisons with previous works are made. The corresponding cube is listed in Table 7 in Appendix. These results clearly illustrate the superiority of our new framework, compared with [12].

4 Conclusions

A new general framework for evaluating the algebraic degree tuple of NFSR-based cryptosystems is exploited and formalized to provide a new way of constructing distinguishing attacks. This is based on a new technique called *composite numeric mapping*, which gives a new general idea for iteratively estimating the upper bound on algebraic degree tuple of NFSR-based cryptosystems. We prove that *composite numeric mapping* is at least as good as *numeric mapping*, and most likely better than it demonstrated by applications. Based on this new technique, an efficient algorithm for algebraic degree tuple estimation of NFSR-Based cryptosystems is proposed and applied to Trivium-like ciphers to prove the effectiveness of our new framework. To the best of our knowledge, this is the first time that the idea of estimating the algebraic degree of a cryptographic primitive in one variable is considered and explored into cryptanalysis. The new framework may help to give the cryptosystem designers more insight to choose the required number of iterative rounds, and we expect that our new framework could become a new generic tool to measure the security of NFSR-based cryptosystems. In the future, it would be interesting to work on converting distinguishing attacks to key recovery attacks, and applications to other NFSR-based cryptosystems.

Acknowledgements. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported by the National Natural Science Foundation of China under Grant 61602514, 61802437, 61272488, 61202491, 61572516, 61272041, 61772547, National Cryptography De-

velopment Fund under Grant MMJJ20170125 and National Postdoctoral Program for Innovative Talents under Grant BX201700153.

References

1. Cannière, C.: Trivium: A stream cipher construction inspired by block cipher design principles. In: Katsikas, S.K., Lòpez, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 171-186. Springer, Heidelberg (2006).
2. Hell, M., Johansson, T., Maximov, A., Meier, W.: The Grain Family of Stream Ciphers. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 179-190. Springer, Heidelberg (2008).
3. Babbage, S., Dodd, M.: The MICKEY Stream Ciphers. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 191-209. Springer, Heidelberg (2008).
4. ECRYPT. The eSTREAM project, http://www.ecrypt.eu.org/stream/
5. Wu, H.: ACORN: a lightweight authenticated cipher (v3). CAESAR Submission, (2016). http://competitions.cr.yp.to/round3/acornv3.pdf
6. Cannière, C., Dunkelman, O., Knežević, M.: KATAN and KTANTAN-A family of small and efficient hardware-oriented block ciphers. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 272-288. Springer, Heidelberg (2009).
7. Aumasson, J.-P., Henzen, L., Meier, W., Naya-Plasencia,M.: Quark: A lightweight hash. In: Mangard, S., Standaert, F.-X. (eds.) CHES 2010. LNCS, vol. 6225, pp.1-15. Springer, Heidelberg (2010).
8. Aumasson, J., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A lightweight hash. J. Cryptol. 26(2), 313-339 (2013).
9. Canteaut, A., Carpov, S., Fontaine, C., Lepoint, T., Naya-Plasencia, M., Paillier, P., Sirdey, R.: Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. In: Peyrin, T. (eds.) FSE 2016. LNCS, vol. 9783, pp. 313-333. Springer, Heidelberg (2016).
10. Chakraborti, A., Chattopadhyay, A., Hassan, M., Nandi, M.: TriviA: A fast and secure authenticated encryption scheme. In: G*ü*neysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 330-353. Springer, Heidelberg (2015).
11. Chakraborti, A., Nandi, M.: TriviA-ck-v2. CAESAR Submission (2015). http://competitions.cr.yp.to/round2/triviackv2.pdf
12. Liu, M.: Degree evaluation of NFSR-based cryptosystems. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 227-249. Springer, Cham (2017).
13. Ding, L., Wang, L., Gu, D., Jin, C., Guan, J.: Algebraic degree estimation of A-CORN v3 using numeric mapping. Secur. Commun. Netw. 2019, 1-5 (2019). https://doi.org/10.1155/2019/7429320. Article ID 7429320
14. Yang, J., Liu, M., Lin, D.: Cube cryptanalysis of roundreduced ACORN. In: Lin, Z., Papamanthou, C., Polychronakis, M. (eds.) ISC 2019. LNCS, vol. 11723, pp. 44-64. Springer, Cham (2019).
15. Ding, L., Wang, L., Gu, D., Jin, C., Guan, J.: A New General Method of Searching for Cubes in Cube Attacks. In: Meng, W., Gollmann, D., Jensen, C.D., Zhou, J. (eds.) ICICS 2020. LNCS, vol. 12282, pp. 369-385. Springer, Cham (2020).
16. Kesarwani, A., Roy, D., Sarkar, S., Meier, W.: New cube distinguishers on NFSR-based stream ciphers. Des. Codes Cryptogr. 88, 173-199 (2020).
17. Xu, C., Zhang, B., Feng, D.: Linear cryptanalysis of FASER128/256 and TriviA-ck. In: Meier, W., Mukhopadhyay, D. (eds.) INDOCRYPT 2014. LNCS, vol. 8885, pp. 237-254. Springer, Cham (2014).

Appendix

A The Procedure $\mathbf{CDEG}_{Tri}$ for $\delta = B$ and $\delta = C$

The procedure $\mathbf{CDEG}_{Tri}$ in the two cases, $\delta = B$ and $\delta = C$, are described in Algorithm 4 and 5, respectively.

Algorithm 4 $\mathbf{CDEG}_{Tri}\left(g_\delta\left(S^{(t)}\right), D^{(t)}\right)$ for $\delta = B$

If $t < r_A + 1$ then:
Return $\mathbf{CDEG}\left(\left(a_{r_A}^{(t)} \cdot a_{r_A+1}^{(t)}\right) \oplus \mathcal{L}_B\left(S^{(t)}\right), D^{(t)}\right)$;
If $t \geq r_A + 1$ then:
Return $\mathbf{CDEG}\left(h, D^{(t)}\right)$, where $h = \left(b_{r_C}^{(t-r_A)} \cdot b_{r_C+1}^{(t-r_A)} \cdot b_{r_C+1}^{(t-r_A-1)}\right) \oplus \left(b_{r_C}^{(t-r_A)} \cdot b_{r_C+1}^{(t-r_A)} \cdot \mathcal{L}_A\left(S^{(t-r_A-1)}\right)\right) \oplus \left(b_{r_C+1}^{(t-r_A)} \cdot b_{r_C+1}^{(t-r_A-1)} \cdot \mathcal{L}_A\left(S^{(t-r_A)}\right)\right) \oplus \left(\mathcal{L}_A\left(S^{(t-r_A)}\right) \cdot \mathcal{L}_A\left(S^{(t-r_A-1)}\right)\right) \oplus \mathcal{L}_B\left(S^{(t)}\right)$.

Algorithm 5 $\mathbf{CDEG}_{Tri}\left(g_\delta\left(S^{(t)}\right), D^{(t)}\right)$ for $\delta = C$

If $t < r_B + 1$ then:
Return $\mathbf{CDEG}\left(\left(c_{r_B}^{(t)} \cdot c_{r_B+1}^{(t)}\right) \oplus \mathcal{L}_C\left(S^{(t)}\right), D^{(t)}\right)$;
If $t \geq r_B + 1$ then:
Return $\mathbf{CDEG}\left(h, D^{(t)}\right)$, where $h = \left(b_{r_A}^{(t-r_A)} \cdot b_{r_A+1}^{(t-r_A)} \cdot b_{r_A+1}^{(t-r_A-1)}\right) \oplus \left(b_{r_A}^{(t-r_A)} \cdot b_{r_A+1}^{(t-r_A)} \cdot \mathcal{L}_B\left(S^{(t-r_A-1)}\right)\right) \oplus \left(b_{r_A+1}^{(t-r_A)} \cdot b_{r_A+1}^{(t-r_A-1)} \cdot \mathcal{L}_B\left(S^{(t-r_A)}\right)\right) \oplus \left(\mathcal{L}_B\left(S^{(t-r_A)}\right) \cdot \mathcal{L}_B\left(S^{(t-r_A-1)}\right)\right) \oplus \mathcal{L}_C\left(S^{(t)}\right)$.

B The Cubes Used in Our Attacks

Table 6. Some new cubes found by Algorithm 2 for Trivium-like ciphers

Cipher	# Rounds	Cube size	Cube
TriviA-SC v2	1046	62	0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36, 38,40,42,44,46,48,50,52,54,56,58,60,62,64,66,68,70, 72,75,77,79,81,83,85,87,89,91,93,95,97,99,101,103, 105,107,109,111,115,117,121,123,125,127
The full simplified variants of TriviA-SC	1152	63	0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36, 38,40,42,44,46,48,50,52,54,56,58,60,62,64,66,68,70, 72,74,76,78,81,83,85,87,89,91,93,95,97,99,101,103, 105,107,109,111,113,115,117,121,123,125,127

Table 7. The cubes used in our distinguishing attack on the full simplified variant of TriviA-SC v2

Cipher	Cube size	Cube
The full simplified variant of TriviA-SC v2	59	0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38,40,42, 44,46,48,50,52,54,56,58,60,62,64,66,68,70,73,75,77,79,81,83, 85,87,89,91,93,95,97,99,101,103,105,107,109,121,123,125,127