The welch-gong stream cipher: evolutionary path

Zidaric, N.; Mandal, K.; Gong, G.; Aagaard, M.

Cryptography and Communications (2024) 16:129–165
https://doi.org/10.1007/s12095-023-00656-0

RESEARCH

The welch-gong stream cipher - evolutionary path

N. Zidarič[1] · K. Mandal[2] · G. Gong[3] · M. Aagaard[3]

Received: 2 November 2022 / Accepted: 3 June 2023 / Published online: 5 August 2023

Abstract
This survey presents the rich history of the Welch-Gong (WG) Stream cipher family. It has been a long journey that lead the WG stream ciphers to become practical. The evolutionary path is a combination of mathematical endeavour and engineering striving to transfer pure mathematical functions to practical encryption algorithms for various applications. This path began as the pioneering work on WG transformation sequences with 2-level autocorrelation, leading to important breakthroughs in the early 2000's, such as the submission of the first WG stream cipher to the eSTREAM competition in 2005 and the subsequent introduction of the WG stream cipher family $WG(m, l)$, followed by extensive work on particular instances proposed for various (mostly lightweight) applications. A recent construction using a WG permutation is the authenticated encryption WAGE, submitted to the NIST LWC competition in 2019. The story of the WG stream cipher is by far not finished. The future opens numerous possibilities for WG stream ciphers and WAGE, with applications in both lightweight environments and in high-performance computing. We conclude the survey with new ideas and open problems.

Keywords The welch-gong stream cipher · Pseudorandom sequences · Hardware

Mathematics Subject Classification (2010) 11T71

The work was done when the authors were in University of Waterloo.

✉ N. Zidarič
n.zidaric@liacs.leidenuniv.nl

✉ K. Mandal
kmandal@unb.ca

G. Gong
ggong@uwaterloo.ca

M. Aagaard
maagaard@uwaterloo.ca

[1] Leiden Institute of Advanced Computer Science, Leiden University, Leiden 2311 EZ, The Netherlands

[2] Faculty of Computer Science, University of New Brunswick, Fredericton New Brusnwick E3B 5A3, Canada

[3] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo Ontario N2L 3G1, Canada

1 Introduction

This work is presenting the rich and long history of the Welch-Gong (WG) Stream cipher family, imperative for the design of the authenticated encryption scheme WAGE, a Round 2 NIST LWC Candidate. The WG stream ciphers are based on the WG transformations on m-sequences and generate keystreams with proven randomness and cryptographic properties. The theoretical foundations for the WG transformations were laid in the 90's and early 2000's. The WG stream cipher was first proposed by Nawaz and Gong in 2005 and the candidate WG-29 reached the phase 2 of the eSTREAM competition. $WG(m, l)$ stream ciphers are composed of a linear feedback shift register (LFSR) of degree l and a (decimated) WG transformation, defined over the same extension field $\mathbb{F}(2^m)$. WG cipher's security depends on the length of the LFSR and the cryptographic strength of the WG transformation used in the cipher. In order to achieve the highest security against existing generic passive attacks, such as algebraic attacks, DFT attacks and distinguishing attacks, we studied the cryptographic and randomness properties for $7 \leq m \leq 16$ and presented criteria for optimal selection of parameters. In the past decade, the lightweight variants WG-5, WG-7, and WG-8, suitable for constrained environments, such as RFID, were proposed and subjected to rigorous cryptanalysis. The bigger instance WG-16 was designed for use in confidentiality and integrity algorithms in mobile communications, such as 4G-LTE networks.

In 2017, we answered the NIST LWC call for submissions and designed a new lightweight authenticated encryption WAGE. The name WAGE is a permutation of WG-AE. The WAGE permutation is based on a 37-stage Galois NLFSR over $\mathbb{F}(2^7)$ with decimated WG permutations and newly designed 7-bit Sboxes. We use the WAGE permutation in the unified sponge-duplex mode to achieve the authenticated encryption functionality that provides 128-bit security. Our security analysis shows that WAGE resists diffusion, algebraic, differential, linear, and meet-in-the-middle distinguishers. Finally, we present possible future directions for WAGE in modern applications using multi-party computation, fully homomorphic encryption schemes, and zero-knowledge proofs.

This survey is organized as follows. In Section 2 we present basic concepts and definitions. In Section 3 we introduce pioneering work on WG transformation sequences, and in Section 4 the important breakthroughs from the early 2000's, such as the WG stream cipher family and submission to the eSTREAM competition. In Section 5, we present instances WG-5, WG-7, WG-8, WG-16 and WG-29, their applications, hardware implementations and some cryptanalysis. Then we present the authenticated encryption WAGE in Section 6. In Section 7, we discuss cryptographic primitives with polynomial Sboxes and propose to use the WAGE structure in MiMC.

2 Some basic concepts and definitions

This section provides basic concepts and definitions that are important for understanding, analyzing, and constructing sequences and WG stream ciphers. We introduce feedback shift register sequences (Section 2.1), properties of periodic binary sequences, such as linear span (Section 2.2), and correlation of sequences (Section 2.3). Then we introduce Boolean functions and vector Boolean functions in polynomial form, and present nonlinearity and resiliency (Section 2.4), followed by differential uniformity (Section 2.5), propagation (Section 2.6) and finally algebraic immunity (Section 2.7).

The following notation will be used throughout the paper.

- For a positive integer N, $\mathbb{Z}_N = \{0, 1, \cdots, N-1\}$ is the residue class ring modulo N.
- For positive integers $n, r, q = 2^r$, $\mathbb{F}_{q^n} = GF(q^n)$ denotes a finite field with q^n elements; $\mathbb{F}_{q^n}^*$, the multiplicative group of $\mathbb{F}_{q^n}$.
- For positive integers m, n, such that $m \mid n$, the trace function from $\mathbb{F}_{q^n}$ to $\mathbb{F}_{q^m}$ is given as $Tr_m^n(x) = \sum_{i=0}^{n/m-1} x^{q^{mi}}$, $x \in \mathbb{F}_{q^n}$. For $m = 1$, we use the notation $Tr(x)$ when the parameter n is clear, i.e., the trace function from $\mathbb{F}_{q^n}$ to $\mathbb{F}_q$ is defined as $Tr(x) = x + x^q + \cdots + x^{q^{n-1}}$.
- For a positive integer n, $\mathbb{F}_2^n = \{\mathbf{x} = (x_0, x_1, \cdots, x_{n-1}) | x_i \in \mathbb{F}_2\}$ is a vector space over $\mathbb{F}_2$ of dimension n.
- $\mathbf{a} = \{a_i\}$, $a_i \in \mathbb{F}_2$, a sequence over $\mathbb{F}_2$, is called a *b*inary sequence. If $\mathbf{a}$ is a periodic sequence with period v, then we also denote $\mathbf{a} = (a_0, a_1, \cdots, a_{v-1})$, an element in $\mathbb{F}_2^v$.
- Decimated sequences: Let $b_i = a_{id}$, $i = 0, 1, \cdots$. Then $\mathbf{b} = \{b_i\}$ is called a d-decimation of $\mathbf{a}$. For example, $\mathbf{a} = 1001011$ and 3-decimation of $\mathbf{a}$ is given by $\mathbf{b} = 1110100$.
- The Hamming weight of a binary vector $\mathbf{a} = (a_0, a_1, \cdots, a_{n-1})$, or a binary number $\overline{a} = \sum_{i=0}^{n-1} a_i 2^i$, or an element in $\mathbb{F}_{2^n}$ $a = \sum_{i=0}^{n-1} a_i \alpha_i$, where $\{\alpha_0, \cdots, \alpha_{n-1}\}$ is a basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$, is given by $w(\mathbf{a}) = w(\overline{a}) = w(a) = |\{i \mid 0 \le i < n, a_i = 1\}|$.
- Let A be a matrix, and A^T be its transpose.

2.1 Feedback shift register (FSR) sequences over $\mathbb{F}_q$

A feedback shift register contains n memory cells, a feedback function from $\mathbb{F}_{q^n}$ to $\mathbb{F}_q$, as shown in Fig. 1 [36]. The n-tuple $(a_0, a_1, \cdots, a_{n-1})$ is referred to as *an initial state* of the FSR, the state transition is given as

$$(a_0, a_1, \cdots, a_{n-1}) \longrightarrow (a_1, a_2, \cdots, a_n)$$

where the feedback element is computed by

$$a_n = f(a_0, a_1, \cdots, a_{n-1}),$$

and the output sequence is $a_0, a_1, \cdots, a_n, \cdots$ which satisfies the following recursive relation:

$$\begin{array}{c} a_{k+n} = f(a_k, a_{k+1}, \cdots, a_{k+n-1}), k = 0, 1, \cdots \\ (a_k, a_{k+1}, \cdots, a_{k+n-1}), \ \text{the} k\text{th state.} \end{array}$$

When f is an nonlinear function, the output $\mathbf{a} = \{a_i\}$ is referred to as an *nonlinear feedback shift register (NFLSR) sequence*, otherwise, as a *linear feedback shift register (LFSR) sequence*.

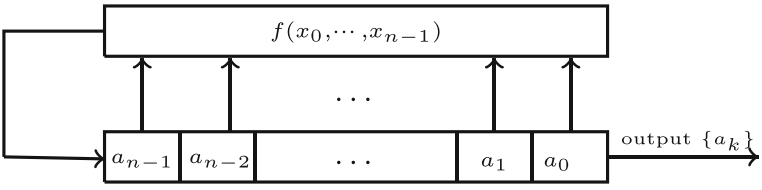


Fig. 1 A general model of FSR

In the LFSR case, we have

$$f(x_0, \cdots, x_{n-1}) = \sum_{i=0}^{n-1} c_i x_i, c_i \in \mathbb{F}_q \rightarrow f(x) = x^n - \sum_{i=0}^{n-1} c_i x^i,$$

where $f(x)$ is referred to as the characteristic polynomial of the LFSR. The $n \times n$ matrix given by

$$A = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & & & & \\ c_0 & c_1 & c_2 & \cdots & c_{n-1} \end{pmatrix} \tag{1}$$

is called a state transition matrix. We have

$$(a_1, \cdots, a_n)^T = A \cdot (a_0, \cdots, a_{n-1})^T,$$
$$(a_k, \cdots, a_{k+n})^T = A^k \cdot (a_0, \cdots, a_{n-1})^T.$$

In this survey, we only consider $q = 2^m$ where m is a positive integer. Let $n = m\ell$. When $m = 1$, i.e., the binary FSR, $n = \ell$. So, we will use n for the binary case. Thus, the feedback function f is a Boolean function in n variables when $m = 1$ (i.e., $q = 2$).

2.2 Properties of periodic binary sequences

2.2.1 Cyclotomic cosets modulo $2^n - 1$

Let $C = \{1, 2, 2^2, \cdots, 2^{n-1}\}$. Then C is a subgroup of the multiplicative group of $\mathbb{Z}_{2^n-1}$. In terms of C, we define a relation, say $\sim$, on $\mathbb{Z}_{2^n-1}$ as follows: for any $a, b \in \mathbb{Z}_N$ $(N = 2^n - 1)$

$$a \sim b \Longleftrightarrow a \equiv 2^i b \pmod{N} \text{ for some} i : 0 \leq i < n.$$

The relation $\sim$ is an equivalence relation, which induces a partition on $\mathbb{Z}_N$. We denote the equivalence class containing s by C_s. It can be represented as

$$C_s = \{s, s2, s2^2, \cdots, s2^{n_s-1}\},$$

where n_s is the smallest positive integer such that

$$2^{n_s} s \equiv s \pmod{N}. \tag{2}$$

Definition 1 *The set C_s is called a (cyclotomic) coset modulo N, the smallest number in C_s is called a coset leader modulo N, and n_s, the size of the coset C_s, is called the order of s* with respect to 2 modulo N.

Example 1 For $n = 4$, we have the following cosets modulo 15:

$$\begin{aligned} &C_0 = \{1\}, \\ &C_1 = \{1, 2, 4, 8\}, \\ &C_3 = \{3, 6, 12, 9\}, \\ &C_5 = \{5, 10\}, \\ &C_7 = \{7, 14, 13, 11\}, \\ &\mathbb{Z}_{15} = C_0 \cup C_1 \cup C_3 \cup C_5 \cup C_7, \text{ and} \\ &\text{the coset leaders: } \{0, 1, 3, 5, 7\}. \end{aligned} \tag{3}$$

For the theory of sequences, the reader is referred to [37] for details and the references therein.

2.2.2 Trace representation and linear span of sequences and functions

Let $\mathbf{a} = (a_0, a_1, a_2, \cdots)$ be a binary sequence satisfying the following recursive relation:

$$a_{k+n} = \sum_{i=0}^{n-1} c_i a_{k+i}, k = 0, 1, \cdots . \tag{4}$$

The polynomial $t(x) = x^n + \sum_{i=0}^{n-1} c_i x^i$, where $c_i \in \mathbb{F}_2$, is called a *characteristic polynomial* of $\mathbf{a}$ and n is the *length or order* of the LFSR. If $t(x)$ is the characteristic polynomial with *minimal degree*, then $t(x)$ is called the *minimal polynomial* of $\mathbf{a}$. In this case, the LFSR given by $t(x)$ has the shortest length. The *linear span or linear complexity* of a sequence $\mathbf{a}$, denoted $LS(\mathbf{a})$, is defined as the length of the shortest LFSR which generates this sequence.

If $t(x)$ is primitive, then $\mathbf{a}$ has period $2^n - 1$ and $\mathbf{a}$ is called an *m-sequence*. For example, for $n = 3$, $t(x) = x^3 + x + 1$, then $\mathbf{a} = 1001011$ is an m-sequence of period 7.

For any binary sequence $\{a_i\}$ of period p with $p|2^n - 1$, we have the following trace representation:

$$a_i = f(\alpha^i), i = 0, 1, \ldots,$$

where

$$f(x) = \sum_{s \in I} Tr_1^{n_s}(\beta_s x^s), \beta_s \in \mathbb{F}_{2^{n_s}}^*, \tag{5}$$

s is the coset leader of C_s with order n_s, and I is the subset of the set consisting all the coset leaders modulo $2^n - 1$. The linear span of $\mathbf{a}$, can be determined as follows:

$$LS(\mathbf{a}) = \sum_{s \in I} n_s.$$

Any function mapping from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$ has the polynomial form of (5). The algebraic degree of f is defined as

$$deg(f) = \max_{s \in I} w(s).$$

If $I = \{d\}$ with $\gcd(d, 2^n - 1) = 1$, then we have

$$a_i = Tr(\beta \alpha^{di}), i = 0, 1, \cdots, \beta \in \mathbb{F}_{2^n}^*,$$

where $f(x) = Tr(\beta x^d)$, which produces an m-sequence of period $2^n - 1$.

2.2.3 *m*-sequences over extension fields and filtering sequences

We now define m-sequences over an extension field. Let $p(x) = x^l + \sum_{i=0}^{l-1} p_i x^i$, $p_i \in \mathbb{F}_{2^m}$, a primitive polynomial over $\mathbb{F}_{2^m}$ and $\mathbf{b} = \{b_i\}$ with

$$b_{k+l} = \sum_{i=0}^{l-1} p_i b_{k+i}, k = 0, 1, \cdots,$$

then $\mathbf{b}$ is an m-sequence over $\mathbb{F}_{2^m}$ of degree l with period $q^l - 1$ ($q = 2^m$). The elements of the m-sequence $\mathbf{b}$ can be represented by the following trace representation

$$b_i = Tr_m^n(\beta \alpha^i), \beta \in \mathbb{F}_{q^l}, \tag{6}$$

where α is a root of $p(x)$ in $\mathbb{F}_{q^l}$.

Let $g(x)$ be a polynomial function from $\mathbb{F}_{2^m}$ to $\mathbb{F}_2$ with the trace representation given by (5) where n is replaced by m and the other parameters are also changed accordingly, i.e.,

$$g(x) = \sum_{s \in I} Tr_1^{m_s}(\beta_s x^s), \beta_s \in \mathbb{F}_{2^{m_s}}^*, \tag{7}$$

where $m_s = |C_s|$, the coset size of the coset C_s with the coset leader s modulo $2^m - 1$, so $m_s \mid m$. In the following, for simplicity, we may set $\beta = 1$ in (6). We define a binary sequence $\mathbf{u} = \{u_i\}$ whose elements given by

$$u_i = g(b_i) = g(Tr_m^n(\alpha^i)), i = 0, 1, \cdots .$$

The sequence $\mathbf{u}$ is a binary sequence with period $2^n - 1$, which is referred to as a *geometric* sequences in [49].

Proposition 1 *(Linear span of geometric sequences) With the above notation, the linear span of* $\mathbf{u}$ *can be computed as follows:*

$$LS(\mathbf{u}) = \sum_{s \in I} m_s l^{w(s)}. \tag{8}$$

Remark 1 Note that for a binary geometric sequence the linear span is determined by the sizes of the cosets modulo $2^m - 1$ and ℓ for $n = m\ell$.

2.3 Correlation of sequences

Let $\mathbf{a} = \{a_i\}$ and $\mathbf{b} = \{b_i\}$ be two binary sequences with period v. The (periodic) cross correlation function of two sequences $\mathbf{a}$ and $\mathbf{b}$ is defined as

$$C_{\mathbf{a},\mathbf{b}}(\tau) = \sum_{i=0}^{v-1} (-1)^{a_i + b_{i+\tau}}, \tau = 0, 1, \cdots$$

where $i + \tau$ is reduced by modulo v. When $\mathbf{a} = \mathbf{b}$, it becomes the (periodic) autocorrelation function of $\mathbf{a}$:

$$C_{\mathbf{a}}(\tau) = \sum_{i=0}^{v-1} (-1)^{a_i + a_{i+\tau}}, \tau = 0, 1, \cdots .$$

If

$$C_{\mathbf{a}}(\tau) = \begin{cases} v & \text{if } \tau \equiv 0 \bmod v \\ -1 & \text{otherwise,} \end{cases}$$

then we say that the sequence $\mathbf{a}$ has *(ideal) 2-level autocorrelation function*, shortened as $C(\tau)$ if the context is clear.

Any binary m-sequence has 2-level autocorrelation. For example, $\mathbf{a} = 1001011$ is an m-sequence of period 7, whose autocorrelation function is given by $C(\tau) = 7$ for $\tau \equiv 0 \bmod 7$ and $C(\tau) = -1$ otherwise. All known constructions of binary sequences with 2-level autocorrelation were collectively introduced in [37], and no new binary sequences with 2-level autocorrelation have been found since. Note that binary 2-level autocorrelation sequences of period v correspond to the cyclic Hadamard difference sets in combinatorics.

There are only a few known constructions of 2-level autocorrelation sequences. These include the number theory based constructions and the finite field based constructions. The

number theory based 2-level autocorrelation sequences are Legendre sequences, twin prime sequences, and Hall's sextic residue sequence. For the finite field based constructions, there are four types of constructions: m-sequences and GMW sequences using subfields, hyperoval constructions, the Welch-Gong (WG) transformation construction, and the Dillon-Dobbertin Kasami power function construction, including 3-term and 5-term sequences. Dillon and Dobbertin also established that the three constructions of WG sequences, given in Section 3 are the same class.

Notes for Table 1

- We omit the linear span of twin prime sequences.
- GMW*: we only list one type of the GMW sequences. For other types of GMW or generalized GMW and subfield constructions, the reader is referred to Section 8.4 in [37].

In Subsections 2.4-2.7 we will introduce some important definitions of cryptographic properties for functions (either in Boolean form or in polynomial form), i.e., nonlinearity, resiliency, differential property, propagation and algebraic immunity in general designs of cipher algorithms.

2.4 Nonlinearity and resiliency of Boolean functions and vector Boolean functions in polynomial form

2.4.1 Polynomial functions and Boolean functions

Note that for any polynomial function from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$, there is a one-to-one correspondence to a Boolean function in n variables. In this survey, we will use polynomial representations of functions from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$ to define cryptographic terms. Let $\Gamma = \{\gamma_0, \gamma_1, \cdots, \gamma_{n-1}\}$ be a basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$. If $Tr(\gamma_i \gamma_j) = \delta_{ij}$, where δ_{ij} is the Kronecker delta function, defined as

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases}$$

then Γ is a self-dual basis.

For two elements $x, y \in \mathbb{F}_{2^n}$ and the basis $\Gamma = \{\gamma_0, \gamma_1, \cdots, \gamma_{n-1}\}$ of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$, where $x = \sum_{i=0}^{n-1} x_i \gamma_i$ and $y = \sum_{i=0}^{n-1} y_i \gamma_i$, and where $x_i, y_i \in \mathbb{F}_2$, we have the following relation:

Lemma 1 *With the above notation, denoting* $\mathbf{x} = (x_0, \cdots, x_{n-1})$ *and* $\mathbf{y} = (y_0, \cdots, y_{n-1})$, *then*

$$Tr(xy) = \sum_{0 \le i,j < n} x_i y_j Tr(\gamma_i \gamma_j).$$

Especially, when Γ *is self-dual,*

$$Tr(xy) = \mathbf{x} \cdot \mathbf{y} = \sum_{i=0}^{n-1} x_i y_i,$$

where $\mathbf{x} \cdot \mathbf{y}$ *is the inner product of* $\mathbf{x}$ *and* $\mathbf{y}$.

Table 1 All known binary sequences with (ideal) 2-level autocorrelation

	Sequences with 2-level Autocorrelation	Year	Period p	Linear span
Number theory	Legendre sequence [75]	1932	p is prime, $p = 3 \mod 4$	$L_p = \begin{cases} \frac{p+1}{2} \text{ if } p = -1 \mod 8 \\ \frac{p-1}{2} \text{ if } p = 1 \mod 8 \\ p \text{ if } p = 3 \mod 8 \\ p-1 \text{ if } p = 5 \mod 8 \end{cases}$
	Hall's sextic residue sequences [13, 48]	1957	$p = 4a^2 + 27$	$L_p = \begin{cases} 1 + \frac{p-1}{6} \text{ if } p = -1 \mod 8 \\ p \text{ if } p = 3 \mod 8 \end{cases}$
			$p = 2^n - 1$	
Hyperoval	Segre sequences [17]	1955	$n \geq 5, n$ odd	$L_n = n\ell_n,\ \ell_n = \ell_{n-2} + \ell_{n-4} + 1,$ $\ell_2 = 0, \ell_3 = \ell_4 = 1, \ell_5 = 3, n \geq 7$
	Glynn I sequences [29]	1983	$n \geq 7, n$ odd	$L_n = n\ell_n,\ \ell_n = \ell_{n-2} + \ell_{n-4} + \ell_{n-6} + \ell_{n-8} + 1,$ $\ell_5 = 1, \ell_7 = 3, \ell_9 = 7, \ell_{11} = 13, n \geq 13$
	Glynn II sequences [29]	1983	$n \geq 11, n$ odd	$L_n = n\ell_n,\ \ell_n = \ell_{n-2} + 3\ell_{n-4} - \ell_{n-6} - \ell_{n-8} + 1,$ $\ell_3 = 1, \ell_5 = 1, \ell_7 = 5, \ell_9 = 7, n \geq 11$
Finite field	m-sequences [36]	1938	any n	$L_n = n$
		1954		
	GMW sequences* [45, 86]	1962	$n \geq 6$ with $n = m \cdot \ell$	$L_n = m\ell^{w(s)}$
	3-term sequences [39, 74]	1997	n odd integer	$L_n = 3n$
	5-term sequences [74]	1998	$n \neq 0 \mod 3$	$L_n = 5n$
	WG sequences [74]	1998	$n \neq 0 \mod 3$	$L_n = n(2^{\lceil \frac{n}{3} \rceil} - 3)$
	Dillon-Dobbertin Kasami power function sequences [21]	2004	$n \geq 6$	$L_n = n(2F_{k_1} - 1), k_1 = \min(k', n - k')$ $kk' = 1 \mod n, F_{i+2} = F_i + F_{i+1}, F_0 = F_1 = 1$

2.4.2 Hadamard transform and nonlinearity

Let $f(x)$ be a function from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$. The *H*adamard (or Walsh or Fourier) transform of f is defined by

$$\hat{f}(\lambda) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{f(x)+Tr(\lambda x)}, \lambda \in \mathbb{F}_{2^n}.$$

For a polynomial function $f(x)$ from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$, we also use f to represent the evaluation of f in $\mathbb{F}_{2^n}$ under some order of the elements in $\mathbb{F}_{2^n}$. For example, we can write $f = (f(0), f(1), f(\alpha), \cdots, f(\alpha^{2^n-2}))$. Recall that α is a primitive element in $\mathbb{F}_{2^n}$.

The *distance* between two binary vectors $\mathbf{a} = (a_0, \cdots, a_{v-1})$ and $\mathbf{b} = (b_0, \cdots, b_{v-1})$, denoted by $d(\mathbf{a}, \mathbf{b})$, is defined as the number of disagreements of terms of $\mathbf{a}$ and $\mathbf{b}$, i.e.,

$$d(\mathbf{a}, \mathbf{b}) = |\{i \mid a_i \neq b_i, 1 \leq i < n\}| \text{ or equivalently}$$
$$d(\mathbf{a}, \mathbf{b}) = w(\mathbf{a} + \mathbf{b}),$$

where $w(\mathbf{x})$ is the Hamming weight of $\mathbf{x}$.

The *nonlinearity* of f, denoted as N_f, is defined by the *minimum distance* between f and all affine functions. In other words,

$$N_f = \min_{\lambda \in \mathbb{F}_{2^n}, c \in \mathbb{F}_2} d(f, Tr(\lambda x) + c)$$

or equivalently

$$N_f = 2^{n-1} - \frac{1}{2}\hat{f}_{\max},$$

where

$$\hat{f}_{\max} = \max_{\lambda \in \mathbb{F}_{2^n}} |\widehat{f}(\lambda)|.$$

The function f is said to have *k-order correlation immunity* if $\widehat{f}(\lambda) = 0$ for all $1 \leq w(\lambda) \leq k$. If $f(x)$ is *balanced*, i.e., there are 2^{n-1} zeros in the evaluation of $f(x)$, and is k-order correlation immune, then we say that f is *k-order resilient*.

2.4.3 Vectorial Boolean functions in polynomial form and their nonlinearity

We say that F is an (n, m)-*vectorial Boolean function in polynomial form* or simply an (n, m)-function if it is a function mapping from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^m}$. Let $\Gamma = \{\alpha_0, \alpha_1, \cdots, \alpha_{m-1}\}$ be a basis of $\mathbb{F}_{2^m}$ over $\mathbb{F}_2$ (not necessary self-dual). For

$$\mathbf{b} = (b_0, \cdots, b_{m-1}) \in \mathbb{F}_2^m \leftrightarrow \beta = \sum_{i=0}^{m-1} b_i \alpha_i \in \mathbb{F}_{2^m},$$

an (n, m)-function F can be written as

$$F(x) = (f_0(x), \cdots, f_{m-1}(x)) \leftrightarrow F(x) = \sum_{i=0}^{m-1} f_i(x)\alpha_i, x \in \mathbb{F}_{2^n},$$

where f_i's are polynomial functions from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$. According to Lemma 1, we may have $\mathbf{b} \cdot F = Tr_1^m(\xi F)$ for some $\xi \in \mathbb{F}_{2^m}^*$. In particular, $\mathbf{b} \cdot F = Tr_1^m(\beta F)$ when Γ is self-dual. (Here we misuse the notation F as both an m dimensional vector and an element in $\mathbb{F}_{2^m}$.)

The *nonlinearity* of F, denoted as N_F, is defined by

$$N_F = \min_{\xi \in \mathbb{F}_{2^m}^*} N_{Tr_1^m(\xi F)}.$$

Equivalently,

$$N_F = 2^{n-1} - \frac{1}{2}\hat{F}_{\max},$$

where

$$\hat{F}_{\max} = \max_{\lambda \in \mathbb{F}_{2^n}, \xi \in \mathbb{F}_{2^m}^*} |\widehat{Tr_1^m}(\xi F)(\lambda)|.$$

Example 2 For $n = 4$ and $m = 2$, let $\mathbb{F}_{2^4}$ be defined by $t_4(x) = x^4 + x + 1$ with $t_4(\gamma) = 0$ and $\mathbb{F}_{2^2}$, defined by $t_2(x) = x^2 + x + 1$ with $t_2(\alpha) = 0$. Thus $\{\alpha, \alpha^2\}$ is a self-dual basis of $\mathbb{F}_4$ over $\mathbb{F}_2$. Let

$$F(x) = Tr_1^4(x^3)\alpha + Tr_1^4(x^7)\alpha^2$$

which is a (4, 2)-vectorial Boolean function in the polynomial form. For $\beta \in \mathbb{F}_4$,

$$g_\beta(x) = Tr_1^2(\beta F) = Tr_1^4(x^3)Tr_1^2(\beta\alpha) + Tr_1^4(x^7)Tr_1^2(\beta\alpha^2), x \in \mathbb{F}_{16}$$

a function from $\mathbb{F}_{16}$ to $\mathbb{F}_2$. The Hadamard transform of $g_\beta(x)$ is given by

$$\widehat{g_\beta}(\lambda) = \sum_{x \in \mathbb{F}_{16}} (-1)^{g_\beta(x) + Tr_1^4(\lambda x)}.$$

For $\lambda = \gamma$ and $\beta = \alpha$, we have $g_\beta(x) = Tr_1^4(x^3)$ and

$$\widehat{g_\beta}(\lambda) = \sum_{x \in \mathbb{F}_{16}} (-1)^{Tr_1^4(x^3) + Tr_1^4(\lambda x)} = 0.$$

When $\beta \in \{1, \alpha, \alpha^2\}$, the nonlinearity of $g_\beta(x)$ is equal to 4 which is maximum.

2.5 Differential *k*-uniform

Let F be an (n, m)-vectorial boolean function in the polynomial form. For any $a \in \mathbb{F}_{2^n}^*$, $b \in \mathbb{F}_{2^m}$, we denote

$$\Delta(a, b) = |\{x | x \in \mathbb{F}_{2^n}, F(x) + F(x + a) = b\}|.$$

We say that F is *differentially k-uniform distributed* if there are at most k solutions in $\mathbb{F}_{2^n}$ of the equation

$$F(x) + F(x + a) = b$$

i.e., $\Delta(a, b) \leq k$.

2.6 Propagation of boolean functions

Let f be a function from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$. The *additive autocorrelation* of f is defined as

$$A_f(a) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{f(x) + f(x+a)}, \ a \in \mathbb{F}_{2^n}.$$

We say that f has *k-order propagation* if $A_f(a) = 0$ for $1 \leq w(a) \leq k$.

2.7 Algebraic immunity of functions

Let B_n be the set consisting of all functions mapping from $\mathbb{F}_{2^n}$ to $\mathbb{F}_2$, i.e., of all Boolean functions in n variables. The *algebraic immunity* of f, denoted by $AI(f)$, is defined as

$$AI(f) = \min_{g \in B_n} \{\deg(g) \mid f \cdot g = 0 \text{ or } (f+1) \cdot g = 0\},$$

where $f \cdot g$ is the multiplication of two Boolean functions f and g. The algebraic immunity is upper bounded by $\lceil \frac{n}{2} \rceil$ [19].

3 WG transformation sequences with 2-level autocorrelation (1998 - 2004)

The Welch-Gong (WG) transformation sequences are binary sequences of period $2^n - 1$ with 2-level autocorrelation. They were discovered by Golomb, Gong, and Gaal in 1998 [74]. The work in [74] presents five new classes of binary sequences of period $2^n - 1$ with ideal 2-level autocorrelation. According to the historic development of those conjectures, Dr. Golomb named the sequences conjectured in Conjectures 4 and 5, Welch-Gong transformation sequences. All of the conjectured sequences correspond to new cyclic Hadamard difference sets. The 2-level autocorrelation was verified for $5 \leq n \leq 20$ in the case $n = 3k-1$, and for $5 \leq n \leq 19$ in the case $n = 3k-2$. Shortly after, the authors of [73] gave another construction of WG transformation sequences and verified the autocorrelation for $5 \leq n \leq 23$. In 1999 [20], Dillon proved WG sequences for odd n using the representation from [73]. Finally, in 2004, Dillon and Dobbertin provided a new general construction which includes all the conjectured sequences in their milestone work presented in [21].

In this section we present the first, the second and the third definition of WG sequences in Sections 3.1, 3.2, and 3.3, respectively. We conclude with a brief note on unified study of sequences, Boolean functions and univariate polynomials in Section 3.4.

3.1 First definition of WG sequences

The first definition of Welch-Gong transformation was presented in [74] as follows.

Definition 2 *Let α be a primitive element in $\mathbb{F}_{2^n}$, $\mathbf{a} = \{a_i\}$, a binary sequence of period $2^n - 1$ with 2-level autocorrelation, and $f(x)$, the trace representation of* $\mathbf{a}$*, i.e., $a_i = f(\alpha^i)$. Then, $WGT(x) = f(x+1) + Tr(1)$ is called a* Welch-Gong transformation *and a sequence $\mathbf{b} = \{b_i\}$, $b_i = WGT(\alpha^i) = f(\alpha^i + 1) + Tr(1)$, a* Welch-Gong transformation sequence*, WG sequence for short.*

Unfortunately, when $\mathbf{a}$ is a 2-level autocorrelation sequence, there exists only one case that the WG transformation produces the sequence with 2-level autocorrelation.

Fact 1 *(WG sequences conjectured, 1998 [74]) Let $f(x) = Tr(x+x^{q_1}+x^{q_2}+x^{q_3}+x^{q_4})$, $x \in \mathbb{F}_{2^n}$, where the q_i's are defined by*

$n = 3k-1$	$n = 3k-2$
$q_1 = 2^k + 1$	$q_1 = 2^{k-1} + 1$
$q_2 = 2^{2k-1} + 2^{k-1} + 1$	$q_2 = 2^{2k-2} + 2^{k-1} + 1$
$q_3 = 2^{2k-1} - 2^{k-1} + 1$	$q_3 = 2^{2k-2} - 2^{k-1} + 1$
$q_4 = 2^{2k-1} + 2^k - 1$	$q_4 = 2^{2k-1} - 2^{k-1} + 1$

and $WGT(x)$, the WG transformation of f, i.e., $WGT(x) = f(x+1) + Tr(1)$. Then two sequences $\mathbf{a} = \{a_i\}$ and $\mathbf{b} = \{b_i\}$ defined by

$$a_i = f(\alpha^i), i = 0, 1, \cdots, \tag{9}$$
$$b_i = WGT(\alpha^i) = f(\alpha^i + 1) + Tr(1), i = 0, 1, \cdots. \tag{10}$$

have 2-level autocorrelation, which is verified for $5 \leq n \leq 20$.

Proposition 2 *(1998 [74]) The trace representation of the WG sequences is given by*

$$WGT(x) = \sum_{s \in I} Tr(x^s),$$

where $I = I_1 \cup I_2$ and where for $m = 3k - 1$,

$$I_1 = \{2^{2k-1} + 2^{k-1} + 2 + i | 0 \leq i \leq 2^{k-1} - 3\}$$
$$I_2 = \{2^{2k} + 3 + 2i | 0 \leq i \leq 2^{k-1} - 2\} \tag{11}$$

and for $m = 3k - 2$,

$$I_1 = \{2^{k-1} + 2 + i | 0 \leq i \leq 2^{k-1} - 3\}$$
$$I_2 = \{2^{2k-1} + 2^{k-1} + 2 + i | 0 \leq i \leq 2^{k-1} - 3\}. \tag{12}$$

In order to introduce the other forms of the definitions of WG sequences, we first introduce the following well known exponents.

Definition 3 *(Kasami, 1971 [52]) The power function x^d on $\mathbb{F}_q$ ($q = 2^n$) is called a* Kasami power function *and d a* Kasami exponent *when*

$$d = 2^{2k} - 2^k + 1 \textit{where } k < n \textit{ and } \gcd(k, n) = 1.$$

3.2 Second definition of WG sequences

Fact 2 *(Conjecture, 1998 [73]) Let*

$$\sigma(x) = (x+1)^d + x^d, \ 3k \equiv 1 \pmod{n}.$$

Then $\sigma(x)$ is a 2-to-1 map on $\mathbb{F}_{2^n}$. Let

$$N = \begin{cases} Im(\sigma) = \{\sigma(x) \mid x \in \mathbb{F}_q\} & \text{if } n \text{ is even} \\ F_q \setminus Im(\sigma) & \text{if } n \text{ is odd}, \end{cases} \tag{13}$$

and $\mathbf{b} = \{b_i\}$ where

$$b_i = \begin{cases} 0 & \text{if } \alpha^i \in N \\ 1 & \text{otherwise}. \end{cases} \tag{14}$$

Then $\mathbf{b}$ has 2-level autocorrelation, which is verified for $5 \leq n \leq 23$.

The following assertion has been also experimentally verified in [73]. Shortly after, Dobbertin proved that in [25].

Property 1 (Dobbertin, 1999 [25]) The sequence constructed in Fact 2 is the same as the WG sequence class constructed in Fact 1.

3.3 Third definition of WG sequences

Fact 3 *(Dillon and Dobbertin, 2004 [21]) Let* $\gcd(k, n) = 1$, $k < n$ *and*

$$\Delta_k(x) = (x+1)^d + x^d + 1, x \in \mathbb{F}_{2^n}, \tag{15}$$

where d *is the Kasami exponent. Let*

$$B_k = Im(\Delta_k) = \{\Delta_k(x) | x \in \mathbb{F}_{2^n}\}$$

and $\mathbf{a} = \{a_i\}$ *where*

$$a_i = \begin{cases} 0, & \alpha^i \in B_k \\ 1, & otherwise \end{cases} \tag{16}$$

Then $\mathbf{a}$ *is a 2-level autocorrelation. Furthermore, the trace representation of* $\mathbf{a}$ *has the form* $Tr(t(x))$*, where* $t(x)$ *is a permutation on* $\mathbb{F}_{2^n}$.

Lemma 2 *(Dillon and Dobbertin, 2004 [21]) With* $3k \equiv 1 \bmod n$*, let*

$$\begin{aligned} r_1 &= 2^k + 1 \\ r_2 &= 2^{2k} + 2^k + 1 \\ r_3 &= 2^{2k} - 2^k + 1 \\ r_4 &= 2^{2k} + 2^k - 1. \end{aligned}$$

and

$$t(x) = x + x^{r_1} + x^{r_2} + x^{r_3} + x^{r_4}.$$

Then $t(x)$ *is a permutation on* $\mathbb{F}_{2^n}$*. In this case,* $\mathbf{a}$ *is a 5-term sequence with the trace representation* $Tr(t(x))$*, and* $WGT(x) = Tr(t(x+1)+1)$*, the WG transformation, produces the WG sequence.*

Definition 4 *We call* $t(x+1)+1$*, a WG permutation, denoted as* $WGP(x)$*:*

$$WGP(x) = t(x+1) + 1.$$

From Lemma 2, this set of $\{r_i\}_{i=1}^4$ unifies the two cases given in Fact 1. Although they have different exponents, they are identical after applying the trace function because they belong to the same cosets modulo $2^n - 1$. In WG cipher design, we will use the Dillon and Dobbertin set, since $WGP(x)$ is a permutation, which will be used in the key initialization algorithm of the WG stream cipher family. These exponents are in the same cosets as those in first representation.

In 2014, Mandal *et al.* found a set of new exponents from Lemma 2, along with the algebraic degree and linear span [63, 64].

Proposition 3 *(Mandal* et al.*, 2014 [64]) The trace representation of the WG sequences given in Lemma 2 is shown as*

$$\begin{aligned} WGP(x) &= \textstyle\sum_{i \in J} x^i, x \in \mathbb{F}_{2^n} \\ WGT(x) &= Tr(WGP(x)) = \textstyle\sum_{i \in J} Tr(x^i), x \in \mathbb{F}_{2^n}, \end{aligned} \tag{17}$$

where $J = J_1 \cup J_2$ *and for* $n = 3k - 1$

$$\begin{aligned} J_1 &= \{2^{2k-1} + 2^{k-1} + 2 + i | 0 \le i \le 2^{k-1} - 3\} \\ J_2 &= \{2^{2k} + 3 + 2i | 0 \le i \le 2^{k-1} - 2\} \end{aligned}$$

and for $n = 3k - 2$

$$J_1 = \{2^{k-1} + 2 + i | 0 \le i \le 2^{k-1} - 3\} \cup \{1\}$$
$$J_2 = \{2^{2k-1} + 2^{k-1} + 2 + i | 0 \le i \le 2^{k-1} - 3\}.$$

The total number of exponents in J *for both cases is equal to* $(2^{\lceil \frac{n}{3} \rceil} - 3)$.

Property 2 (2014 [64]) The algebraic degree of $WGT(x)$ in (17), denoted as $\deg(WGT(x))$, is given by $\deg(WGT(x)) = \lceil \frac{n}{3} \rceil + 1$ and the linear span of WG sequences is $n(2^{\lceil \frac{n}{3} \rceil} - 3)$.

Note that the exponents in J in Proposition 3 and I in Proposition 2 belong to the same cosets modulo $2^n - 1$. However, $\sum_{i \in I} x^i$ is not a permutation polynomial on $\mathbb{F}_{2^n}$.

To achieve the maximum level of security of a WG cipher against different attacks, the authors in [64] presented all optimal parameter instances of the (decimated) WG transformations over $\mathbb{F}_{2^n}$ for $7 \le n \le 16$. The optimality is defined with respect to the cryptographic properties such as low Hamming weight decimations, high algebraic immunity, high algebraic degree, high nonlinearity, and high resiliency.

3.4 Unified study of sequences, Boolean functions and univariate polynomials

Theoretical results on WG sequences rely on earlier work, such as [18], where the authors established the first connection between

$$\begin{array}{l} \text{binary sequences with period } 2^n - 1, \\ \text{polynomial functions } \mathbb{F}_{2^n} \mapsto \mathbb{F}_2, \text{ and} \\ \text{Boolean functions in } n \text{ variables.} \end{array} \tag{18}$$

They also provided theoretical results on the linear structure, strict avalanche condition, and the nonlinearity of exponential Boolean functions and exponential permutations on $\mathbb{F}(2)^n$. In [40] Gong and Golomb successfully used the connection (18), together with the tools for pseudorandom sequence analysis, to analyze DES S-boxes. When they considered the relationship between sequences and functions, they realized that monomials, which correspond to m-sequences, are not secure when used as component functions in block ciphers. This lead to a concept of linear span for polynomial functions introduced in [40].

4 Cryptographic properties of WG sequences and WG stream cipher in eSTREAM (2000 - 2008)

This section is divided into three major parts. First, the WG transformation sequence generators, formalized in the year 2000 (Section 4.1). Second, the generators matured to the WG stream cipher, introduced and submitted to the eSTREAM competition in 2005 (Section 4.3). And third, the WG stream cipher family, introduced in 2008 (Section 4.2). The WG stream cipher submitted to eSTREAM competition is an instance of the WG stream cipher family, which is why in this work, we introduce the family first and this instance later, instead of following a chronological order.

4.1 WG transformation sequence generators

The WG sequence generators were formalized by Gong and Youssef [43] in the year 2000. Their work provides the definition of WG transformation sequences and presents their randomness properties, such as (ideal) 2-level auto correlation, cross correlation with m-sequences, and the balance property, and their cryptographic properties, such as nonlinearity, resilience property, linear span and degree when regarded as Boolean functions.

It is well-known that a sequence can also be viewed as a Boolean function [37], and the WG transformation can be viewed as a WG sequence as well as a Boolean function. Table 2 provides a summary of the WG transformation's sequence properties and Boolean function properties. As shown in the first two columns of Table 2, a WG sequence has the following randomness properties: period $2^n - 1$, it is balanced, it has 2-tuple distribution, 2-level autocorrelation, 3-level (optimal) cross-correlation with respect to an m-sequence, and a high linear span $n(2^{\lceil \frac{n}{3} \rceil} - 3)$. When the WG transformation is viewed as an n-variable Boolean function, it is balanced, it has an algebraic degree $(\lceil \frac{n}{3} \rceil + 1)$, 3-valued Hadamard transform $\{0, \pm 2^{\frac{n+1}{2}}\}$, high nonlinearity $2^{n-1} - 2^{\frac{n-1}{2}}$ and r-th order resiliency with $1 \leq r \leq n - \lceil \frac{n}{3} \rceil$ for a suitable decimation, as shown in the last two columns of Table 2.

The authors of [43] note that the WG sequence **b** can be obtained using the decimation property: the 5-term sequence **a** can be generated by using five linear feedback shift registers and one AND gate. This property of the WG sequences allows them to have an efficient implementation for small n by operating decimation on **a** together with a table look-up.

4.2 WG stream cipher family

4.2.1 Stream cipher design principles

A stream cipher is an analogue of one-time-pad encryption scheme where the key with uniform distribution is replaced by a key stream generator. A key stream generator is imple-

Table 2 Profiles of WG transformations [43, 44]

WG sequences profile	WG sequence		WG Transformation as Booleamn func.	WG Transformation Boolean profile
$2^n - 1$	Period	$\leftrightarrow$	Boolean	n variables
✓	Balance	$\leftrightarrow$	Balance	✓
✓	2-tuple distribution		NC	
2-level	Auto correlation		NC	
$0, \pm 2^{(n+1)/2}$ ‡	Hadamard transform spectrum			$0, \pm 2^{(n+1)/2}$ ‡
$\{-1, -1 \pm 2^{(n+1)/2}\}$ ‡ optimal w.r.t. the Welch bound.	Cross correlation with m-sequences	$\leftrightarrow$	Non-linearity	$2^{n-1} - 2^{(n-1)/2}$ ‡
$n(2^{\lceil n/3 \rceil} - 3)$ †	Linear span	$\leftrightarrow$	Linear span	$n(2^{\lceil n/3 \rceil} - 3)$ †
	NC		Degree	$\lceil n/3 \rceil + 1$
	NC		1-resilient	

‡ for n odd † increases exponentially in n NC - no corresponding concept

mented by a pseudorandom bit or sequence generator (PSG). The attacker's goal is to recover a secret key (called seed) used in the key stream generator.

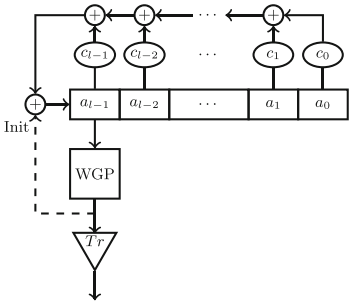
In stream cipher design there are two phases. One is the key initialization algorithm (KIA), and the other is the pseudorandom sequence generation (PSG). The KIA takes two inputs: one is an initial vector (IV), a public information, and the other is a secret key k, a pre-shared encryption key. The goal of KIA is to scramble key bits with the IV in order to get a bit stream as random as possible. The output of KIA is provided as an initial value to the PSG. The KIA is executed only once for each encryption session. After the key initialization, the PSG starts to output a key stream which is used in encryption. For the message $m = (m_0, m_1, \cdots, m_{N-1}), m_i \in \mathbb{F}_2$ and the output of PSG $(s_0, s_1, \cdots, s_{N-1}), s_i \in \mathbb{F}_2$, the ciphertext is given by

$$c_i = m_i + s_i, i = 0, 1, \cdots, N-1.$$

In the WG stream cipher family, the KIA (initialization phase) is a nonlinear feedback shift register and the PSG (running phase) is a filtering generator.

4.2.2 WG stream cipher family *WG*(*m*, *l*)

The WG stream cipher family was first introduced in 2008 [72]. The authors show how to rewrite the exponents of the polynomial $t(x)$ in order to avoid implementing finite field inversion modules, which are very expensive in hardware. This trick was used in subsequent implementations of WG-29 and in implementations of instances WG-8, WG-16.



Mathematical parameters	
m	Bit-width of LFSR
$g(x)$	defining $GF(2^m)$
$p(x) = \sum_{i=0}^{\ell} c_i x^i$	CP for LFSR

Find k such that $3k \equiv 1 \mod m$.

$$WGP(x) = t(x+1) + 1$$
$$t(x) = x + x^{r_1} + x^{r_2} + x^{r_3} + x^{r_4}$$
$$WGT(x) = Tr(WGP(x))$$

$$r_1 = 2^k + 1$$
$$r_2 = 2^{2k} + 2^k + 1$$
$$r_3 = 2^{2k} - 2^k + 1$$
$$r_4 = 2^{2k} + 2^k - 1$$

The KIA and PSG are specified as follows.

Updating process

$$a_{k+\ell} = \begin{cases} \sum_{i=0}^{\ell-1} c_i a_{i+k} + WGP(a_{k+\ell-1}), & 0 \le k < 2\ell \\ & \text{(in KIA phase, NLFSR)} \\ \sum_{i=0}^{\ell-1} c_i a_{i+k} & k \ge 2\ell \\ & \text{(in PSG running phase)} \end{cases}$$

Output: $s_k = WGT(a_{k+2\ell+\ell-1}), k = 0, 1, \cdots$ (filtering generator)

This is referred to as a WG(m, l) *stream cipher family*, shortened as a WG-m stream cipher family.

We list the randomness properties of WG keystreams as follows.

1. An output sequence has period $2^n - 1$ and is balanced.
2. It has the ideal 2-level autocorrelation function.
3. The output sequence has ideal t-tuple distribution, i.e., each t-tuple is equally likely distributed for $1 \leq t \leq \ell$.
4. The linear span exponentially increases with m, which can be determined exactly as

$$LS = \sum_{s \in T} m l^{w(s)}$$

where $T = I$ or J which is given in Proposition 2 or 3, respectively.

The following cryptographic properties of WG transformations for odd m are known.

1. They are 1-order resilient.
2. Their algebraic degree is equal to $\lfloor m/3 \rfloor$.
3. Their nonlinearity is given by $2^{m-1} - 2^{(m-1)/2}$.
4. Their additive autocorrelation between $f(x+a)$ and $f(x)$ has three values: $0, \pm 2^{(m+1)/2}$.
5. They have 1-order propagation property.

In terms of security analysis, we have the following general features for WG stream ciphers.

1. They have guaranteed keystream randomness properties.
2. They are secure against time/memory/data tradeoff attacks, algebraic and correlation attacks.
3. They can be implemented in hardware with reasonable complexity.

4.3 The eSTREAM submission

In 2005, Nawaz and Gong submitted the WG stream cipher [71] as a Profile 2 (stream ciphers for hardware applications with highly restricted resources) candidate to the eSTREAM competition. The submitted instance WG-29, defined over $\mathbb{F}_{2^{29}}$ and using an 11-stage LFSR, reached Phase 2 of the competition. A normal basis was given to reduce hardware implementation area: terms x^{2^i} can be implemented using simple cyclic shifts. The schematic of WG-29 transformation is shown in Fig. 2. According to our notation, WG-29 is $WG(29, 11)$.

The initial state of the LFSR contains 319 bits. For a 128-bit key and 128-bit IV, the rule for loading the LFSR, where each register holds 29 bits, is as follows.

Registers	29-bit Format
0,2,4,6,8	16 bits from the key||8 bits from IV||padding zeros
1,3,5,7,9	8 bits from the key||16 bits from IV||padding zeros
10	8 bits from the key||8 bits from IV||padding zeros

The WG-29 stream cipher has the following randomness properties for keystream.

– Period $2^{319} - 1$

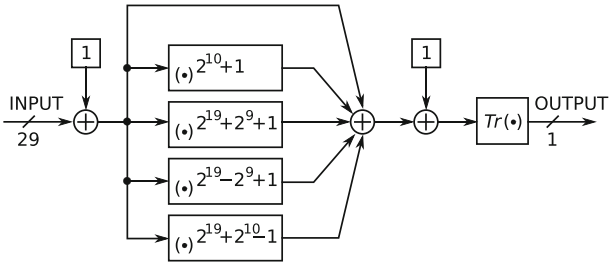
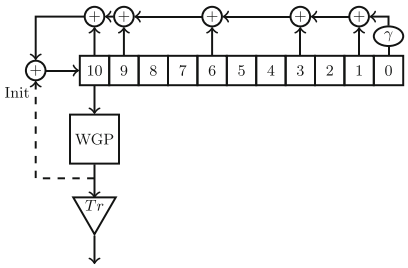


Fig. 2 Block diagram of WG transformation: $\mathbb{F}_{2^{29}} \rightarrow \mathbb{F}_2$[71]

- Balanced
- 2-level autocorrelation
- Ideal t-tupledistribution ($1 \leq t \leq 11$)
- Linear complexity $\approx 2^{45.04}$

Cryptographic properties of the WG transformation over $\mathbb{F}_{2^{29}}$ are listed as follows.

- 1-order resilient
- Algebraic degree 11
- Nonlinearity $= 2^{28} - 2^{14} = 268419072$
- Additive autocorrelation between $f(x+a)$ and $f(x)$ has three values: $0, \pm 2^{15}$
- 1-order propagation



Mathematical parameters

29, Bit-width of LFSR

$g(x) = x^{29} + x^{28} + x^{24} + x^{21} + x^{20} + x^{19}$
$+x^{18} + x^{17} + x^{14} + x^{12} + x^{11} + x^{10}$
$+x^7 + x^6 + x^4 + x + 1$, defining $GF(2^m)$

$g(\alpha) = 0$

$p(x) = x^{11} + x^{10} + x^9 + x^6 + x^3 + x + \gamma$
where $\gamma = \alpha^{464730077}$

$$WGP(x) = t(x+1) + 1$$
$$t(x) = x + x^{r_1} + x^{r_2} + x^{r_3} + x^{r_4}$$
where $x \in GF(2^m)$
$$WGT(x) = Tr(WGP(x))$$

$$r_1 = 2^{10} + 1$$
$$r_2 = 2^{20} + 2^{10} + 1$$
$$r_3 = 2^{20} - 2^{10} + 1$$
$$r_4 = 2^{20} + 2^{10} - 1$$

$$\text{Update} \quad a_{k+11} = \begin{cases} \sum_{i=0}^{10} c_i a_{i+k} + WGP(a_{k+10}) & 0 \leq k < 22 \\ & \text{(in KIA phase, NLFSR)} \\ \sum_{i=0}^{10} c_i a_{i+k} & k \geq 22 \\ & \text{(in running phase)} \end{cases}$$

$$\text{Output} : s_k = WGT(a_{k+32}), k = 0, 1, \cdots$$

The attack complexities against some known attacks are summarized as follows.

- Time/Memory/Data tradeoff attacks: size of internal state is 2^{319}.

– Algebraic attacks: the number of linear equations $\approx \binom{319}{11}$; and the attack complexity $\approx 2^{182}$.
– Correlation attacks: WG transformation is 1-order resilient, and nonlinearity is very high $2^{28} - 2^{14}$.
– Some detailed security analysis has been done in the literature, say [41, 43, 44, 70, 88, 95, 96]

5 WG instances for lightweight cryptography (2008 -)

In this section we present instances of the WG stream cipher family with focus on their hardware implementations and applications for odd m (Section 5.1) and even m (Section 5.2). Details about randomness properties of generated keystreams, cryptographic properties and known-attack complexities for each particular instance are omitted from this work, however, we summarize some newly proposed attacks and countermeasures (Section 5.3). Then we present applications of WG transformations to generate span n sequences (Section 5.4) and conclude this section with design automation (Section 5.5).

5.1 WG-5, WG-7, WG-11, and WG-29

The first implementations of WG stream cipher instance WG-29, with hardware optimizations, such as pipelining and component reuse, were presented in [55]; the WG-29 implementations were subjected to various formal verification techniques. In 2009, the hardware implementations of Multi-Output WG cipher (MOWG) were reported by Lam et al. [56]. The MOWG ciphers use only WGP. The authors implemented instances MOWG-7, MOWG-11, and MOWG-29 in several different ways, ranging from designs using exponent rewrites (Fig. 3), combinational logic, ROM, and finite field multipliers, which were pipelined, superpipelined, and superpipelined with reuse. The multi-output bit-width depends on the instance and the hardware design; for example the MOWG-29 was designed for 17 output bits. Furthermore, the authors identified performance-area differences between implementation technologies used (ASIC and FPGA).

In 2014, El-Razouk et al. [27] presented hardware implementations of WG-29 using an optimal normal basis for $\mathbb{F}_{2^{29}}$. This work is an extension of their preliminary results reported in [26]. After using properties of the trace function, the WGT requires only five multiplications (instead of nine), four of which are needed to compute exponentiation to $2^{10} - 1$ (Fig. 4). In 2015, the same group published further optimized hardware designs of WG-29 using polynomial basis [28]. They used traditional and Karatsuba polynomial basis finite field multipliers. The authors also presented a serialized implementation with a single multiplier for the area-constrained environments, and a pipelined design for the

$$
\begin{aligned}
& 2^{10} - 1 \\
&= \sum_{i=0}^{9} 2^i \\
&= \left(1 + 2^5\right)\left(\left(1 + 2^2\right)\left(\left(1 + 2^1\right)2^0\right) + 2^4\right) \\
&= \left(1 + 2^5\right)\left(\left(1 + 2^2\right)\left(1 + 2^1\right) + 2^4\right)
\end{aligned}
$$

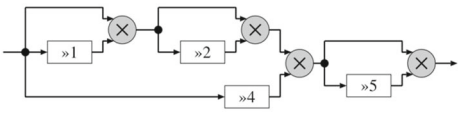


Fig. 3 MOWG-29: computation of x^{2^i-1} in [56], using the trick from [71, 72]

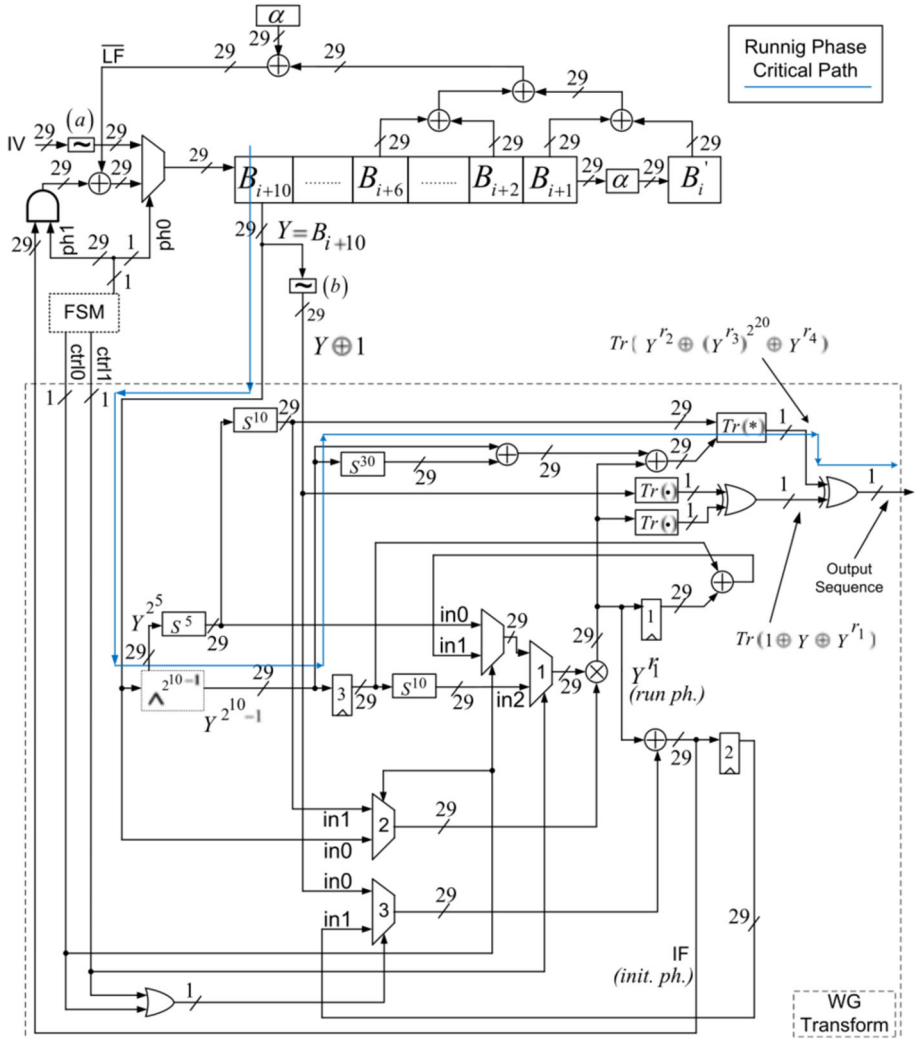


Fig. 4 WG-29 hardware architecture [28], using the trick from [71, 72]

high-throughput environments. Additional designs for WG-29 were presented, but never implemented, in [54] and [11].

In 2010, the authors of [16, 57] proposed the use of the instance WG-7 for encryption and authentication in RFID applications. They provide software implementation results of WG-7 on 4-bit and 8-bit microcontrollers, showing that WG-7 outperforms the state-of-the-art ultra-lightweight ciphers, such as PRESENT, Hummingbird and Grain. Another instance of the WG stream cipher family, explored for the use in RFID systems by Aagaard et al. in [3], is the smallest family member WG-5. The hardware implementations of WG-5 and WG-7 use the cyclotomic coset leaders optimization [3, 69, 91]. The work in [3] presents hardware implementations of WG-5, Grain and Trivium using the same technology and design flow: the results show that WG-5 outperforms both in terms of area and power consumption. The

work in [91] designed a cryptographic engine WGLCE for generation of pseudorandom numbers and for data confidentiality in passive RFID systems. The smallest post place-and-route area for the CMOS 65nm implementation of WG-5 is 890GE. WG-7 was used to build WG-NLFSR, a pseudorandom number generator for securing RFID applications [61]. This work investigates the cycle structure, period, and randomness properties of the composited recurrence relation and its sequences over $\mathbb{F}_{2^7}$ and $\mathbb{F}_{2^5}$. In 2021, WG-5 was used as a part of an optimized application-specific instruction set processor (ASIP) for ultralight Hardware Security Module (HSM) [12]. An EPC RFID tag was chosen as the prototype device for the HSM. The HSM includes custom instructions and dedicated hardware for the WG-5 cipher.

WG(7,23) parameters [57]

$m = 7$, Bit-width of LFSR
$g(x) = x^7 + x + 1$, defining $\mathbb{F}(2^m)$
$g(\alpha) = 0$, α is a root of $g(x)$
$p(x) = x^{23} + x^{11} + \alpha$

$WGP(x^3) = t(x^3 + 1) + 1$, where $x \in \mathbb{F}(2^m)$ and decimation $d = 3$

$r_1 = 2^5 + 1 = 33$
$r_2 = 2^3 + 2^5 + 1 = 41$
$r_3 = 2^3 - 2^5 + 1 = 104$
$r_4 = 2^3 + 2^5 - 1 = 39$
$t(x) = x + x^{33} + x^{39} + x^{41} + x^{104}$
$WGT(x^3) = Tr(WGP(x^3))$
$= Tr(x^3 + x^9 + x^{21} + x^{57} + x^{87})$

WG(5,32) parameters [3]

$m = 5$, Bit-width of LFSR
$g(x) = x^5 + x^4 + x^2 + x + 1$, defining $\mathbb{F}(2^m)$
$g(\alpha) = 0$, α is a root of $g(x)$
$p(x) = x^{32} + x^7 + x^6 + x^4 + x^3 + x^2 + \omega$, where $\omega = \alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1$,
$WGP(x^{11}) = t(x^{11} + 1) + 1$, where $x \in \mathbb{F}(2^m)$ and decimation $d = 11$

$r_1 = 2^2 + 1 = 5$
$r_2 = 2^4 + 2^2 + 1 = 21$
$r_3 = 2^4 - 2^2 + 1 = 13$
$r_4 = 2^4 + 2^2 - 1 = 19$
$t(x) = x + x^5 + x^{13} + x^{19} + x^{21}$
$WGT(x^{11}) = Tr(WGP(x^{11}))$
$= Tr(x^{15})$

WG(5,32) parameters [91]

$m = 5$, Bit-width of LFSR
$g(x) = x^5 + x^4 + x^3 + x + 1$, defining $\mathbb{F}(2^m)$
$g(\alpha) = 0$, α is a root of $g(x)$
$p(x) = x^{32} + x^7 + x^6 + x^5 + x^4 + x + \omega$, where $\omega = \alpha^{15}$,
$WGP(x) = t(x + 1) + 1$, where $x \in \mathbb{F}(2^m)$

$r_1 = 2^2 + 1 = 5$
$r_2 = 2^4 + 2^2 + 1 = 21$
$r_3 = 2^4 - 2^2 + 1 = 13$
$r_4 = 2^4 + 2^2 - 1 = 19$
$t(x) = x + x^5 + x^{13} + x^{19} + x^{21}$
$WGT(x) = Tr(WGP(x))$
$= Tr(x^7)$

5.2 WG-8 and WG-16

In 2013, Fan et al. proposed the lightweight instance WG-8 for the use in embedded devices, RFID, smart cards, and wireless sensor nodes [31]. The authors presented efficient implementations of WG-8 on low-power 8-bit and 16-bit microcontrollers, demonstrating high throughput and energy efficiency. They used three different implementation methods for implementation of the WG-8 permutation: direct look-up table (DLT) with polynomial basis, coset leader based look-up table (CLT) with normal basis, and tower-field arithmetic based approach (TFA) with look-up tables for the subfield $\mathbb{F}_{2^4}$. On both microcontrollers, the DLT approach outperformed both other methods. Moreover, the software implementations on the low-power microcontrollers demonstrated high performance and low energy consumption of the WG-8 stream cipher, when compared to the most previous block ciphers and stream

ciphers. The instance WG-8 was also recognized as a suitable lightweight cipher for IoT [79].

WG(8,20) parameters [31, 91, 92]

$m = 8$, Bit-width of LFSR
$g(x) = x^8 + x^4 + x^3 + x^2 + 1$, defining $\mathbb{F}(2^m)$
$g(\alpha) = 0$, α is a root of $g(x)$
$p(x) = x^{20} + x^9 + x^8 + x^7 + x^4 + x^3 + x^2 + x + \alpha$
$WGP(x^{19}) = t(x^{19} + 1) + 1$, where $x \in \mathbb{F}(2^m)$ and decimation $d = 19$

$r_1 = 2^3 + 1 = 9$
$r_2 = 2^6 + 2^3 + 1 = 73$
$r_3 = 2^6 - 2^3 + 1 = 57$
$r_4 = 2^6 + 2^3 - 1 = 71$
$t(x) = x + x^9 + x^{57} + x^{71} + x^{73}$
$WGT(x^{19}) = Tr(WGP(x^{19}))$
$= Tr(x^9 + x^{37} + x^{53} + x^{63} + x^{127})$ using coset leaders

Later on, Yang et al. presented FPGA and ASIC hardware design and implementation results [91, 92]. They use a look-up table design (LUT) and three different tower field constructions, one tailored to the FPGA cells (TF 1), one using optimal normal basis (TF 2), and one exploiting the algebraic properties of the WG permutation and the trace function (TF 3). All four methods use a parallel LFSR to provide data rates from 1 to 11 bits per clock cycle. Implementation results using FPGA and ASIC technologies show that the LUT method is best suitable for a small WG instance like WG-8. The TF 3 $\mathbb{F}_{((2^2)^2)^2}$ tower field construction uses a normal basis for all three extensions. The exponentiations to the powers of two x^{2^i} are performed as cyclic shifts in the isomorphic field $\mathbb{F}_{2^8}$ with the normal basis representation, but some overhead in area and delay comes from the basis transition matrices. The multipliers and the squarers are implemented with dedicated arithmetic circuits for each level of the tower. For the construction $\mathbb{F}_{((2^2)^2)^2}$, properties of the trace function allow significant optimizations of the WGT module at the cost of reduced throughput for the WGP computation (Fig. 5).

The instance WG-16 was proposed by [30] in 2013. A plain lookup table design is no longer feasible for bigger instances, such as WG-16. A method similar to TF 3 in [92] was used by Fan et al. in [32] for the implementation of WG-16. The design uses dedicated multipliers, an Itoh-Tsujii inversion module, and properties of the trace function in $\mathbb{F}_{(((2^2)^2)^2)^2}$ for the optimization of the WG transformation. The integrated datapath for WGP and WGT was pipelined to increase the performance. Different bases and field constructions were also explored in [98]. The WG-16 instance was not designed for lightweight applications, but rather to be used in the confidentiality and integrity algorithms for 4G-LTE telecommunication systems [30, 89]. WG-16 was used to construct WGIA-128, a linear forgery attack resistant variant of EIA1 [89]. Later on, El-Razouk et al. [28] used the polynomial basis for the implementation of the instance WG-16. Their hardware design follows the same approach as their polynomial basis implementation of WG-29. In summary, the hardware architectures in [28] contain a small number of finite field multipliers and no finite field inversion modules.

In 2019, Zidaric et al. [100] explored tower field constructions and hardware optimizations for the WG-16 stream cipher. The constructions $\mathbb{F}_{(((2^2)^2)^2)^2}$ and $\mathbb{F}_{(2^4)^4}$ were chosen because their small subfields enable high speed arithmetic implementations, and because their regularity provides flexibility in pipeline granularity. The authors presented a design methodology where the tower field constructions guide how to proceed systematically from the algebraic optimizations, through initial hardware implementation, selection of submodules, pipelining, and finally through the fine-tuning hardware optimizations to increase the

clock speed. Eventually, the LFSR became the frequency bottleneck and retiming was applied to increase the LFSR clock speed at no cost in area (Fig. 6). With the exception of speedup, all metric ratios favor the $\mathbb{F}_{(((2^2)^2)^2)^2}$ designs, while the $\mathbb{F}_{(2^4)^4}$ implementations are preferable for the high-frequency applications. Speedups of 3.94× for ASIC and 1.6× for FPGA were achieved w.r.t. the implementations in [32], and a speedup of 1.82× was achieved w.r.t. the implementations in [28].

WG(16,32) parameters [32]

$m = 16$, Bit-width of LFSR
$g(x) = x^{16} + x^5 + x^3 + x^2 + 1$, defining $\mathbb{F}(2^m)$
$g(\alpha) = 0$, α is a root of $g(x)$
$p(x) = x^{32} + x^{25} + x^{16} + x^7 + \omega$,
where $\omega = \alpha^{2743}$
$t(x) = x + x^{2049} + x^{2111} + x^{2113} + x^{63552}$
$WGP(x^{1057}) = t(x^{1057} + 1) + 1$, where $x \in \mathbb{F}(2^m)$ and decimation $d = 1057$
$WGT(x^{1057}) = Tr(WGP(x^{1057}))$

$r_1 = 2^{11} + 1 = 2049$
$r_2 = 2^6 + 2^{11} + 1 = 2113$
$r_3 = 2^6 - 2^{11} + 1 = 63552$
$r_4 = 2^6 + 2^{11} - 1 = 2111$

5.3 More cryptanalysis

In 2012, Orumiehchiha et al. proposed a distinguishing attack on WG-7 by applying Walsh-Hadamard transform to the WG transformation and deriving the linear equations for the LFSR state [77]. Gong et al. extended their work to a general distinguishing attack and suggested criteria to protect the WG stream cipher family from this attack [38]. The countermeasure requires a proper choice of the minimal polynomial of the LFSR. In 2019, a distinguishing attack on WG-8 and WG-16 was proposed [84]. The authors use a counting algorithm for

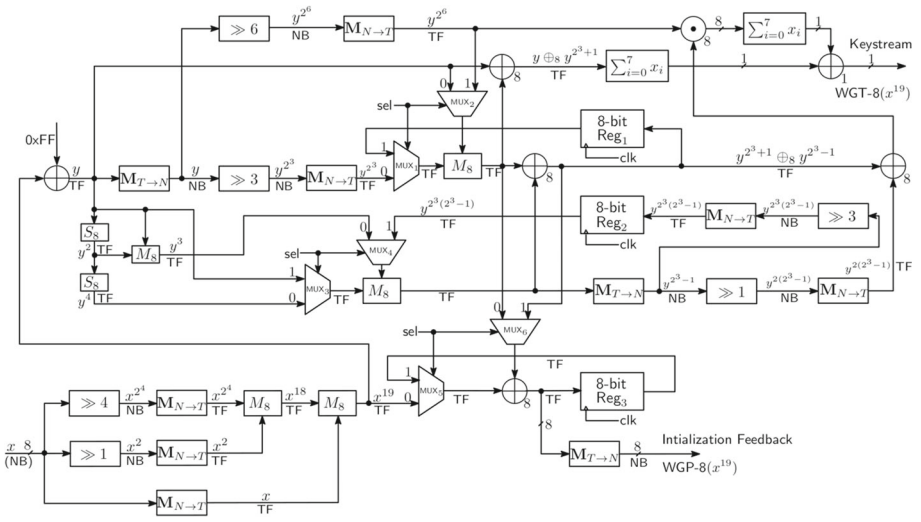


Fig. 5 The WG stream cipher WG-8: the integrated hardware architecture for WGP and WGT using the TF 3 field construction [92]

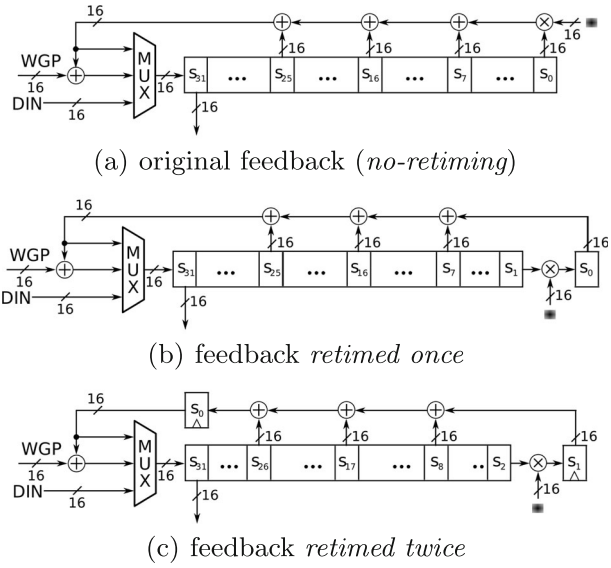


Fig. 6 WG-16 LFSR with feedback retiming [100]

the number of zeros and the number of ones of Boolean functions to increase the attack efficiency. The work in [23, 24] demonstrated the related key attacks on WG-8 and WG-29 based on sliding property, and proposed a new key/IV loading that provides resistance to the proposed attacks. Distinguishing attacks on other instances were reported in [22, 50]. The paper [97] reported the randomness tests of WG keystreams, and they passed all the NIST 15 randomness tests. The work in [82, 83] proposed improved algebraic attacks on stream ciphers based on linear feedback shift registers over binary extension fields, and provided new and improved bounds for the spectral immunity, proposed by Gong et al. [42], for this class of stream ciphers.

Rohit et al., analyzed the nonlinear feedback-based 24-round reduced initialization phase [80, 81]. The authors used a cube attack in a non-blackbox polynomial setting employing the division property. Furthermore, the use of Mixed Integer Linear Programming (MILP) models to theoretically bound the complexity of superpoly recovery allowed them to automate the attack. The authors were able to recover the secret key of WG-5 with data complexity $2^{6.32}$ and time complexity $2^{76.81}$. Their analysis also showed, that the design choices (the feedback and filtering tap positions) of WG-5 offer more resistance to cube attacks than Grain128a and Trivium, where the cube attacks can cover more than half of initialization rounds.

A differential fault attack on instances WG-7, WG-8, WG-16 and WG-29 was reported in [78]. The attack requires a precomputation phase to determine differential patters and an online matching phase, and works under the assumption, that the adversary has the access to the hardware implementation and is able to reset the cipher and re-run it using the same key and IV unlimited times. However, the attack was simulated on a PC, and not performed on the actual hardware.

5.4 Applications of WG to Generate (Modified) De Bruijn Sequences

In 2013, Mandal investigated generating span n sequences using WG transformations and other orthogonal functions in nonlinear feedback shift registers (NLFSRs), and designed the Warbler family of lightweight PRNGs with guaranteed randomness properties, such as period and linear span, using the WG transformations as feedback and filtering functions [58]. Later on, Mandal et al. showed how to generate a long period de Bruijn sequence (e.g., $2^{128/256/1024}$) using the WG transformations and the composited construction [60, 65, 90]. WG transformations used to generate span n sequences from NLFSR were also investigated in [51].

WG-5 used in Warbler-I and Warbler-II [58, 60, 65, 90, 91]

$WGP(x^d) = t(x^d + 1) + 1$, where $t(x) = x + x^5 + x^{13} + x^{19} + x^{21}$

$x \in \mathbb{F}(2^m)$ and decimation d $WGT(x^d) = Tr(WGP(x^d))$

$g(x) = x^5 + x^4 + x^3 + x + 1$, defining $\mathbb{F}(2^m)$

$g(\alpha) = 0$, α is root of $g(x)$ $p(x) = x^6 + x + \omega$, where $\omega = \alpha^{15}$

$d = 1$ $WGT(x) = Tr(x^7)$

$d = 3$ $WGT(x^3) = Tr(x^{11})$

$d = 11$ $WGT(x^{11}) = Tr(x^{15})$

$d = 11$ was also used with the following defining polynomials: $g(x) = x^5 + x^3 + 1$,

$g(x) = x^5 + x^4 + x^2 + x + 1$, $g(x) = x^5 + x^4 + x^3 + x^2 + 1$

In [62], Mandal and Gong introduced a new interesting property, called *WG invariance or invariant under the WG transform* of Boolean functions. It was theoretically shown that, for a specific decimation, the WG transformation on the 5-term and 3-term functions has the WG invariance property. It was proven that when such a WG invariance function is used as a filtering function in a filtering de Bruijn generator, the pseudorandom sequences have an ideal-tuple distribution of a larger length. The authors also studied the Gold and Quadratic functions, and were able to show that the WG transformations on these two classes have the WG invariance property. The authors performed an experiment over the WG transformation of the Kasami power function (KPF), and claimed that except the 5-term and 3-term function, no other KPF has the WG invariance property.

5.5 Design automation

The work in [101] presents a design automation toolkit for hardware implementations of linear and non-linear feedback shift registers (FSRs), and is partially available at [102]. The toolkit is implemented in the GAP computer algebra system [35], and generates both the executable GAP code and the VHDL for synthesizable hardware, testbenches and test-vectors. The primary FSR objects are LFSRs, NLFSRs and filtering functions (implemented as multivariate polynomials). The paper demonstrates the capabilities of the toolkit using the WG-7 and WG-8 keystream generators, among others. Two critical points in the design of this toolkit were (i.) recognition and exploitation of structural similarities between LFSRs, NLFSRs and filters, from both mathematical and hardware perspectives, and (ii.) modular thinking; a cipher can be implemented as a collection of basic modules. For this purpose, the toolkit implements all FSR objects with a regular step (for self-contained execution), and external step, which adds an external value to the feedback for (N)LFSRs, or as a mask to the

output of the filter. A WG keystream generator can be thus implemented as an (i.) LFSR with external step from a WGP filter during the initialization and (ii.) LFSR with a regular step, followed by two filters (WGP and trace), during the running mode. The extended toolkit presented in [99] has the ability to work with polynomial, normal, and their dual bases, and tower field bases. Furthermore, it can now generate synthesizable hardware modules for arbitrary expressions over arbitrary finite fields, including tower fields. Many extensions were added for the purposes of the algorithm design during the design stage of WAGE.

The profiling work in [85] compares two different hardware design approaches, discrete component designs (using finite field arithmetic modules, such as multipliers, implemented as parametrized VHDL) and constant array designs (using look-up tables for the WGP and the WGT modules, generated with the toolkit [101]). The profiling explores the impact of polynomial bases used on the hardware design area for $m = 5, 7, 8, 10, 11, 13, 14, 16$.

6 WAGE - WG in Authenticated Encryption in NIST Lightweight Cryptography (2019 -)

WAGE is a hardware-friendly authenticated encryption based on the lightweight WAGE permutation in the unified sponge-duplex mode [1, 2, 7, 80]. It was a round 2 candidate of the NIST LWC competition [1, 2]. Inspired by the key initialization phase of the WG cipher, the 259-bit lightweight WAGE permutation was designed. The WAGE permutation (see Fig. 7) consists of an LFSR with 37 stages, a WG permutation with decimation $d = 13$, applied 2 times in parallel, and another s-box SB over $\mathbb{F}_{2^7}$, applied 4 times in parallel, and is iterated $111\times$ (Table 3). Specification parameters of WAGE are listed in Table 3, the cryptographic properties of WGP and SB are listed in Table 4, and WAGE security claims in Table 5. With a simple tweak in the control circuit of WAGE, it can be turned into a WG pseudorandom bit generator (PRBG) with proven randomness properties such as long period, balanced, ideal 2-level autocorrelation property and an ideal ℓ-tuple ($1 \leq \ell \leq 37$).

The design rationale for WAGE is outlined in [7, 80]. During the design of WAGE, the toolkit [99, 101] was used for parameter search and (automated) hardware implementations to obtain more accurate implementation costs. This profiling dictated the size of the finite field, basis, and candidates for the LFSR feedback polynomial. The choice of the LFSR polynomial was finalized together with the choice of the positions of s-boxes (WGP and SB) to provide the maximum resistance against differential attack [80]. This analysis was performed with a MILP model, which takes as input the tap positions, the positions of s-boxes (WGP and SB),

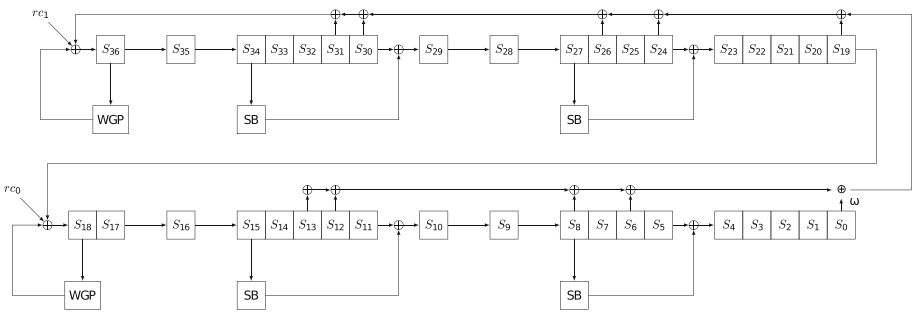


Fig. 7 One round of WAGE permutation

Table 3 Specification parameters of WAGE

$\mathbb{F}_{2^7}$	$f(x) = x^7 + x^3 + x^2 + x + 1$, $f(\omega) = 0$ polynomial basis: $\text{PB} = \{1, \omega, \ldots, \omega^6\}$		
$x \in \mathbb{F}_{2^7}$	$x = \sum_{i=0}^{6} x_i \omega^i, x_i \in \mathbb{F}_2$ vector representation: $[x]_{\text{PB}} = (x_0, x_1, x_2, x_3, x_4, x_5, x_6)$		
LFSR	$fb = S_{31} \oplus S_{30} \oplus S_{26} \oplus S_{24} \oplus S_{19} \oplus S_{13} \oplus S_{12} \oplus S_8 \oplus S_6 \oplus (\omega \otimes S_0)$		
	$\omega \otimes (x_0, x_1, x_2, x_3, x_4, x_5, x_6) \leftarrow (x_6, x_0 \oplus x_6, x_1 \oplus x_6, x_2 \oplus x_6, x_3, x_4, x_5)$		
WGP	$WGP(x^d) = x^d + (x^d + 1)^{33} + (x^d + 1)^{39} + (x^d + 1)^{41} + (x^d + 1)^{104}, d = 13$		
SB Q	$Q(x_0, x_1, x_2, x_3, x_4, x_5, x_6) = (x_0 \oplus (x_2 \odot x_3), x_1, x_2, \overline{x}_3 \oplus (x_5 \odot x_6), x_4, \overline{x}_5 \oplus (x_2 \odot x_4), x_6)$		
SB P	$P(x_0, x_1, x_2, x_3, x_4, x_5, x_6) = (x_6, x_3, x_0, x_4, x_2, x_5, x_1)$		
SB R	$R(x_0, x_1, x_2, x_3, x_4, x_5, x_6) = (x_6, \overline{x}_3 \oplus (x_5 \odot x_6), x_0 \oplus (x_2 \odot x_3), x_4, x_2, \overline{x}_5 \oplus (x_2 \odot x_4), x_1)$		
SB	$(x_0, x_1, x_2, x_3, x_4, x_5, x_6) \leftarrow R^5(x_0, x_1, x_2, x_3, x_4, x_5, x_6)$		
	$(x_0, x_1, x_2, x_3, x_4, x_5, x_6) \leftarrow Q\ (x_0, x_1, x_2, x_3, x_4, x_5, x_6)$		
	$(x_0, x_1, x_2, x_3, x_4, x_5, x_6) \leftarrow\ \ (\overline{x}_0, x_1, \overline{x}_2, x_3, x_4, x_5, x_6)$		
state update unction	$S_{24} \leftarrow S_{24} \oplus \text{SB}(S_{27})$	$S_5 \leftarrow S_5 \oplus \text{SB}(S_8)$	$S_j \leftarrow S_{j+1}$ for $0 \le j \le 35$
	$S_{30} \leftarrow S_{30} \oplus \text{SB}(S_{34})$	$S_{11} \leftarrow S_{11} \oplus \text{SB}(S_{15})$	
	$fb \leftarrow fb \oplus \text{WGP}(S_{36}) \oplus rc_1$	$S_{19} \leftarrow S_{19} \oplus \text{WGP}(S_{18}) \oplus rc_0$	$S_{36} \leftarrow fb$

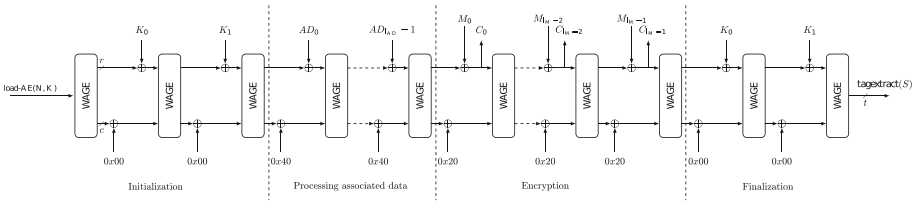


Fig. 8 WAGE: authenticated encryption

and the number of rounds, and returns a minimum number of active s-boxes. The work in [80] also presents the (generic) AEAD algorithm (Fig. 8) and the choice of domain separators and the rate and capacity of the state. Valuable insights were gained during earlier work on Sponge-specific designs [8], filtering NLFSRs using WG transformations [61], and many others [9, 10, 59, 87], to list only a few.

The hardware design of WAGE with the minimal interface is presented in [1, 2, 4]. The smallest post place-and-route area for the CMOS 65nm implementation of WAGE is 2.9kGE. The work in [4] also presents parallel implementations of WAGE with the throughput increases of up to 8×. Later, WAGE was adapted for the LWC hardware interface and evaluated together with other Round 2 candidates using various FPGA [67, 68] and ASIC [5] technologies. The authors of [33] analyzed the security of WAGE against the correlation power analysis. About 10000 power traces are required to recover the 128-bit secret key. As a countermeasure, an optimized masking scheme was proposed in the t-strong non-interference security model. Subsequent analysis of power traces obtained from the masked implementation did not reveal first-order leakage. The proposed masking scheme was also implemented in hardware, where we were able to observe 2.9×, 5.6× and 8.9× area increase for 1,2, and 3-order security, respectively.

The work in [66] proposed an algebraic attack on WG-PRBG using many annihilators simultaneously. The authors were able to recover the initial state using $2^{16.72}$ keystream bits and with the time complexity $2^{108.15}$. The WG-PRBG was proposed and analyzed in [7]: it has guaranteed properties, but limited bits, specifically, the number of consecutive output bits per seed should be less than 2^{18}. In response to [66] we propose to further restrict the number of WG-PRBG output bits, and wish to remind the readers that WAGE, not WG-PRBG, was submitted to the NIST LWC competition.

Recently, WAGE was used for KDF and MIC algorithms in WiFi and CoAP handshake authentication protocols [93, 94]. Both protocols were embedded in the IEEE802.11a OFDM communication protocol and implemented in software defined radio. The work in [53] presented error detection schemes for the nonlinear blocks SB and WGP.

Table 4 WGP and SB cryptographic properties

S-box	Differential uniformity	Nonlinearity	Minimum algebraic degree	Maximum algebraic degree	Fixed point
WGP	6	42	6	6	yes
SB	8	44	3	6	no

Table 5 WAGE security claims (in bits)

Confidentiality	Integrity	Authenticity	Data limit
128	128	128	2^{64}

7 Cryptographic primitives with polynomial Sboxes

In the literature, as early as 1995, Nyberg-Knudsen (1995 [76]) has presented the following DES-like block cipher design using the monomial x^3 over $\mathbb{F}_{2^{33}}$. Starting in the early 2010s, with increasing connected communication, the popularity of multi-party computation (MPC) and fully homomorphic encryption (FHE) for applications in cloud computing and privacy preserving machine learning has greatly advanced. More recently, zero-knowledge proofs found applications in protecting privacy of blockchain transactions. Those schemes are referred to as *zero-knowledge succinct non-interactive argument of knowledge (zkSNARK)* proof systems, and they require hash functions with minimal multiplicative complexity (MiMC). Especially for zkSNARK schemes for Rank-1 Constraint Satisfaction (R1CS), the prover/verifier's complexity only depends on the number of multiplication gates in a fan-in two circuit, where the size of the underlying finite field is not so relevant or it can be easily satisfied that condition without increasing the complexity (for this line of research, see more details in Stark (2018) [14], Aurora [15] (2019), and Polaris [34] (2022) and the references therein).

From this perspective, using a polynomial permutation in a large finite field could fulfill this requirement, since it does not need to be 'lightweight'. Here we briefly introduce some work along this line, i.e., MiMC [6] (Section 7.1), HadesMiMC [47] and Poseidon [46] as the instantiation of HadesMiMC (Section 7.2), and the relationship between WAGE and HadesMiMC, for which the contents are from those papers (Section 7.3).

7.1 MiMC

MiMC-n/n. We use the notation MiMC-m/n to represent MiMC with block size m and key size n. In the following, we introduce the MiMC-n/n block cipher defined over $\mathbb{F}_{2^n}$. Let $x, k \in \mathbb{F}_{2^n}$. The round function of MiMC-n/n is as follows:

$$\begin{aligned} &f(x) = x^3, \quad \text{more general } f(x) = x^d, \gcd(d, 2^n - 1) \\ &f_i(x) = f(x + k + t_i), t_i \in \mathbb{F}_{2^n}, i = 0, \cdots, r-1, t_0 = 0, \end{aligned} \tag{19}$$

where the computation is in $\mathbb{F}_{2^n}$ and t_i's are constant. The encryption function is iterated f_i r times. Note that in this case, the decryption will be a different circuit, since $f_i^{-1} = x^s$ where $3s \equiv 1 (\mathrm{mod} 2^n - 1)$.

The MiMC design can also use Feistel structure, i.e., an NLFSR structure. In this case, the round number will be doubled, compared to a substitution permutation network (SPN) block cipher. But the decryption is the same as encryption where only the order of the constants is reversed.

MiMC in $\mathbb{F}_p$. Both MiMC block cipher and MiMC Feistel/NLFSR can be generated to operate on $\mathbb{F}_p$.

MiMC hash. MiMC can also be used as a permutation in the sponge structure for instantiating a hash function (see Section 6 for a sponge structure). Let MiMCHash-q denote MiMC in the sponge structure to instantiate a hash function with q bits of output for a given permutation of size n, a desired security level s, and rate t, i.e., hash $t = n - 2s$ bits per call to

the permutation. For $n = 4q + 1$, it sets $s = q$, then $r = 2q + 1$. An instance given in [6] is MiMC-n/n for $q = 256$ and $n = 1025$ for 128 bit security. In this case, the rate and capacity are 513 and 512 respectively.

7.2 HadesMiMC and Poseidon

HadesMiMC is defined on $\mathbb{F}_p$. Let $n = lm$ for an n-bit plaintext block and key block, where $m = \lfloor \log p \rfloor$. HadesMiMC uses a general AES-like structure: with an $Sbox(x) = x^3$, where $x \in \mathbb{F}_p$, repeatedly used t times. Similarly as in the case of MiMC, it can use an $Sbox(x) = x^d$ with $\gcd(d, p - 1) = 1$ and good cryptographic properties (Table 6).

The ith round function is defined as

$$G_i(x) = MF_i(x),\ F_i(x) = F(x + k_i) \in \mathbb{F}_p^l,\ x, k_i \in \mathbb{F}_p^l,\ i = 0, 1, \cdots, r - 1,$$

where k_i is a round key generated by a key scheduling algorithm, and $F \in \{F_f, F_p\}$ where both F_f, called a full SPN, and F_p, called a partial SPN, are vectorial functions over $\mathbb{F}_p^l$ where F_f's each component is identical $Sbox(x)$, defined above, and for F_p's component functions, the first component uses $Sbox(x)$ and the others are the identity function (in general, F_p may have one or more components that are the $Sbox$).

Compared with MiMC, HadesMiMC will be more efficient, since its Sbox is much smaller. HadesMiMC can also be used in the sponge structure to instantiate a hash function. Poseidon is such an example, as shown below.

Poseidon hash - an instantiation of HadesMiMC hash. The Poseidon hash function is designed to use the sponge structure. It uses a HadesMiMC structure where a subkey is replaced by a round constant. Here we introduce the Poseidon-128 case. As HadesMiMC, it works on $\mathbb{F}_p$ where $m = \lfloor \log p \rfloor = 255$. Two instances are given below with the following parameters for the $Sbox(x) = x^5$, where r_x is the number of rounds for F_x, $x \in \{f, p\}$.

The permutation polynomial is an $n = lm = 765$ for $l = 3$ and $n = 1275$ for $l = 5$ bit permutation, precisely, a permutation of $\mathbb{F}_p^l$, $l = 3, 5$.

The Poseidon hash function is designed to conjunct with BLS12-381, BN254, Ed25519 curves for zkSNARKs with trusted set-up. The above instance is for the BN254 curve, which has 128 bit security.

7.3 Relationship between WAGE and HadesMiMC

From Stark, Aurora, and Polaris [14, 15, 34], we understand that there is another class of zkSNARK schemes with transparent set-up, and the security depends on the hash functions which do not rely on any computationally hard problems. However, they work on a binary field, e.g., $\mathbb{F}_{2^{256}}$ in Polaris case for 128-bit security. Therefore, we could directly use an analog way of HadesMiMC over $\mathbb{F}_{2^n}$ with the monomial terms x^3, x^5 or x^{-1} as Sboxes. However, replacing these monomials by the WG permutations will result in a smaller number of rounds compared with these monomials. Another case is that x^3, x^5 are not permutations in some

Table 6 AES like structure

1. Add the subkey
2. An Sbox over $\mathbb{F}_p$
3. Mix affine layer: a $l \times l$ maximum distance separable (MDS) matrix over $\mathbb{F}_p$

$n = l\lceil \log p \rceil = lm$	l	r_f	r_p	rate r $bits$	security s bits
765	3	8	57	$2l = 510$	$\lfloor m/2 \rfloor = 127$
1275	5	8	60	$4l = 1020$	$\lfloor m/2 \rfloor = 127$

binary fields with optimal memory access structures, such as $\mathbb{F}_{2^m}$ for $m \in \{16, 32, 64, 128\}$. In order to demonstrate the solutions to those problems, we first look at the relationship between the WAGE structure and the HadesMiMC. Next, we present two types of WAGE related schemes with low multiplicative complexity.

From Section 6, with $(S_0, S_1, \cdots, S_{36})$ being the current state of the registers in WAGE, we have $n = lm = 259$ where $l = 37$ and $m = 7$. Then the next state, i.e, the round function can be represented as

$$x = (S_0, S_1, \cdots, S_{36}) \to MF(x + t_i), x, t_i \in \mathbb{F}_7^{37} \tag{20}$$

where

$$\begin{aligned} F(x) &= (\sigma_0(S_0), \sigma_1(S_1), \cdots, \sigma_{36}(S_{36})) \\ &\sigma_{16} = \sigma_{36} = WGP7 \\ &\sigma_8 = \sigma_{15} = \sigma_{27} = \sigma_{34} = SB \\ &\sigma_i = I \text{ for the rest of } i \end{aligned} \tag{21}$$

and M is a 37×37 matrix resulting from the state transition matrix (see Section 2.1) of the LFSR with degree 37, where a few entries are permuted. (Note that this matrix may not be a MDS matrix, but it has a good diffusion property.)

WGP in HadesMiMC. We now replace the Sbox in HadesMiMC by WGP. The resulted structure is referred to as *WAGE HadesMiMC*, or WHM for short , and list some parameter sets in Table 7.

Below we list some choices for such designs in order to have low number of multiplication gates. Furthermore, the number of constrains of the following WHM designs can be precisely obtained by the hardware implementation of WGPs as shown in Sections 4 and 5.

WAGE-like permutations. Let $F(x) = (\sigma_0(x_0), \cdots, \sigma_{l-1}(x_{l-1}))$ be a function over $\mathbb{F}_q^l$ $(q = 2^m)$, where $x = (x_0, \cdots, x_{l-1}) \in \mathbb{F}_q^l$ and $\sigma_i(x_i)$ is either a WGP on $\mathbb{F}_q$ or the identify function on $\mathbb{F}_q$. Note that $F(x)$ can be represented as an univariate polynomial on $\mathbb{F}_{q^l}$. Define

$$T_i(x) = F_i \circ F_{i-1} \circ \cdots \circ F_0$$

where

$$F_i(x) = A \circ L \circ F(x + t_i), i = 0, 1, \cdots, r - 1,$$

Table 7 Some parameter sets of WHM hash

$n = lm$	$\mathbb{F}_{2^m}$	l	rate r bits	security s bits	comments
$37 \times 7 = 259$	$\mathbb{F}_{2^7}$	37	32	113	WAGE in hash mode
$7 \times 37 = 259$	$\mathbb{F}_{2^{37}}$	7	37	111	WAGE reversed parameters
$5 \times 64 = 320$	$\mathbb{F}_{2^{64}}$	5	64	128	x^3, x^5 are not permutations
$3 \times 128 = 384$	$\mathbb{F}_{2^{128}}$	3	128	128	the same as above
$5 \times 128 = 640$	$\mathbb{F}_{2^{128}}$	5	384	128	

and where both $A(x)$ and $L(x)$ are linearized permutation polynomials on $\mathbb{F}_{q^l}$, and $A(x)$ resulting from the state transfer matrix of an LFSR of degree l, and where t_i are constants in $\mathbb{F}_{q^l}$. (This definition originated from (20) and (21) for WAGE's round function.) Similarly, like WAGE, we can have an authenticated encryption and hash function by placing T_r in the sponge structure.

Nevertheless, those two types of new permutations deserve substantial research on their security. For the WHM, due to the relationship between WAGE and HadesMiMC, the security can also be tackled from the WAGE structure. Furthermore, their implementations in both software and hardware will be much more efficient using the WAGE structure, thanks to the existing efficient implementation of the LFSR.

8 Concluding remarks and open problems

It has been a long journey that lead the WG stream ciphers to become practical. The evolutionary path is a combination of mathematical endeavour and engineering striving to transfer pure mathematical functions to practical encryption algorithms for various applications.

Figure 9 depicts only a few highlights of the quarter of a century long evolutionary path of the WG stream cipher. The evolution can roughly be summarized as the pioneering work on the WG transformation sequences with 2-level autocorrelation, important breakthroughs in the early 2000's, such as the submission of WG stream cipher to the eSTREAM competition and the introduction of the WG stream cipher family $WG(m, l)$, followed by the work on particular instances proposed for various (mostly lightweight) applications, and the most recent construction WAGE, submitted to the NIST LWC competition in 2019. While the story of the WG stream cipher is not finished, the future opens numerous possibilities, such as the possibility of expanding WAGE into a family of authenticated encryption schemes WAGE(m, l), suitable for applications in high-performance computing, and explor-

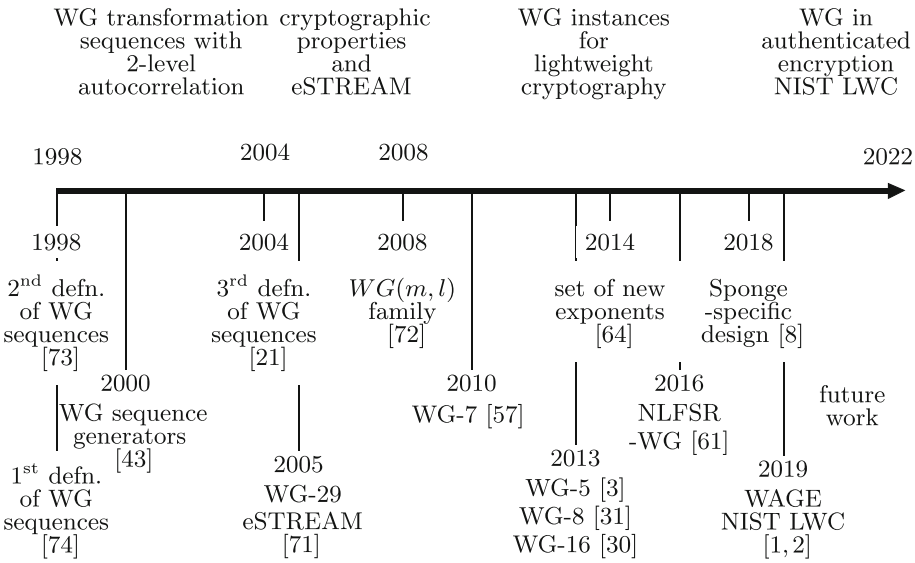


Fig. 9 The Welch-Gong Stream Cipher - Evolutionary Path

ing WAGE structure in MiMC structure for the applications in multiparty computation, fully homomorphic encryption, and zero-knowledge proofs.

As the reader may see, in theory, some cryptographic properties of WGP and WGT for the even case are unsolved. These parameters have important applications in MiMC design for MPC, FHE and zero-knowledge proofs.

In the following, we list the details of the open problems for those n where WGP/WGT exist, i.e., $n \pmod 3 \neq 0$.

1. The nonlinearity of WGP over $\mathbb{F}_{2^n}$ and the distribution of the Hadamard transform of WGP for any n.
2. The differential uniformity of WGP for any n.
3. The nonlinearity of WGT and the distribution of Hadamard transform of WGT for n even.
4. Cryptographic properties in the schemes given in Table 7 for WHM.
5. Progressive bounds for nonlinearity, differential uniformity and algebraic degree for the iterated function $T_i(x), i = 1, 2, \cdots$ in WAGE-like permutations for $m \in \{16, 32, 64, 128, 256\}$ and $l \in \{3, 5, 7\}$.

Acknowledgements The research was supported by NSERC SPG grant.

References

1. Aagaard, M., AlTawy, R., Gong, G., Mandal, K., Rohit, R., Zidaric, N.: WAGE: An authenticated cipher, round 1 submission to *nist lightweight cryptography standardization project* (2019). https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/wage-spec.pdf
2. Aagaard, M., AlTawy, R., Gong, G., Mandal, K., Rohit, R., Zidaric, N.: WAGE: An authenticated cipher, round 2 submission to nist lightweight cryptography standardization project (2019). https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-2/spec-doc-rnd2/wage-spec-round2.pdf
3. Aagaard, M.D., Gong, G., Mota, R.K.: Hardware implementations of the WG-5 cipher for passive RFID tags. In 2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp 29–34 (2013). IEEE
4. Aagaard, M.D., Sattarov, M., Zidaric, N.: Hardware design and analysis of the ACE and WAGE ciphers. NIST LWC workshop 2019, arXiv preprint (2019). arXiv:1909.12338
5. Aagaard, M.D., Zidaric, N.: ASIC benchmarking of round 2 candidates in the NIST lightweight cryptography standardization process. Cryptology ePrint Archive, Paper 2021/049 (2021). https://eprint.iacr.org/2021/049
6. Albrecht, M., Grassi, L., Rechberger, C., Roy, A., Tiessen, T.: MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In Cheon, J.H., Takagi, T. (eds) Advances in Cryptology – ASIACRYPT 2016. Berlin, Heidelberg. Springer Berlin Heidelberg, pp 191–219 (2016)
7. AlTawy, R., Gong, G., Mandal, K., Rohit, R.: WAGE: an authenticated encryption with a twist. IACR Transactions on Symmetric Cryptology – Special Issue on Designs for the NIST Lightweight Standardisation Process, p 132–159 (2020)
8. Altawy, R., Rohit, R., He, M., Mandal, K., Yang, G., Gong, G.: sLISCP-Light: Towards hardware optimized Sponge-specific cryptographic permutations. ACM Trans Embed Comput Syst **17**(4) (2018)

9. AlTawy, R., Rohit, R., He, M., Mandal, K., Yang, G., Gong, G.: sLiSCP: Simeck-based permutations for lightweight Sponge cryptographic primitives. In: Adams, C., Camenisch, J. (eds.) Selected Areas in Cryptography - SAC 2017, pp. 129–150. Springer International Publishing, Cham (2018)
10. AlTawy, R., Rohit, R., He, M., Mandal, K., Yang, G., Gong, G.: Towards a cryptographic minimal design: The sLiSCP family of permutations. IEEE Transactions on Computers **67**(9), 1341–1358 (2018)
11. Ashan, V.: Implementation of WG stream cipher with involution function. Procedia Technology **24**, 790–795 (2016)
12. Ayoub, A.: A flexible ultralight hardware security module for EPC RFID tags. PhD thesis. uwspace (2021). http://hdl.handle.net/10012/17613
13. Baumert, L.D.: Cyclic difference sets, volume 182. Springer (2006)
14. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Paper 2018/046 (2018). https://eprint.iacr.org/2018/046
15. Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for R1CS. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2019, pp. 103–128. Springer International Publishing, Cham (2019)
16. Chai, Q.: Design and analysis of security schemes for low-cost RFID systems. PhD thesis. uwspace (2012). http://hdl.handle.net/10012/6512
17. Chang, A.C., Golomb, S.W., Gong, G., Kumar, P.V.: On the linear span of ideal autocorrelation sequences arising from the Segre hyperoval. Sequences and their Applications-Proceedings of SETA'98, Discrete Mathematics and Theoretical Computer Science (1999)
18. Chang, X., Dai, Z.D., Gong, G.: Some cryptographic properties of exponential functions. In Pieprzyk J, Safavi-Naini R (eds) Advances in Cryptology — ASIACRYPT'94, p 413–418. Berlin, Heidelberg. Springer Berlin Heidelberg (1995)
19. Courtois, N.T., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In Biham, E. (ed) Advances in Cryptology — EUROCRYPT 2003, p 345–359. Berlin, Heidelberg. Springer Berlin Heidelberg (2003)
20. Dillon, J.F.: Multiplicative difference sets via additive characters. Designs, Codes and Cryptography **17**(1), 225–235 (1999)
21. Dillon, J.F., Dobbertin, H.: New cyclic difference sets with singer parameters. Finite Fields and Their Applications **10**(3), 342–389 (2004)
22. Ding, L., Gu, D., Wang, L., Jin, C., Guan, J.: A real-time related key attack on the WG-16 stream cipher for securing 4G-LTE networks. Journal of Information Security and Applications **63**, 103015 (2021)
23. Ding, L., Jin, C., Guan, J., Wang, Q.: Cryptanalysis of lightweight WG-8 stream cipher. IEEE Transactions on Information Forensics and Security **9**(4), 645–652 (2014)
24. Ding, L., Jin, C., Guan, J., Zhang, S., Cui, T., Han, D., Zhao, W.: Cryptanalysis of WG family of stream ciphers. The Computer Journal **58**(10), 2677–2685 (2015)
25. Dobbertin, H.: Kasami Power Functions, pp. 133–158. Permutation Polynomials and Cyclic Difference Sets. Springer, Netherlands, Dordrecht (1999)
26. El-Razouk, H., Reyhani-Masoleh, A., Gong, G.: New hardware implementations of the WG stream cipher, cacr report (2012). https://cacr.uwaterloo.ca/techreports/2012/cacr2012-31.pdf
27. El-Razouk, H., Reyhani-Masoleh, A., Gong, G.: New implementations of the WG stream cipher. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, **22**(9):1865–1878 (2013)
28. El-Razouk, H., Reyhani-Masoleh, A., Gong, G.: New hardware implementationsof $WG(29, 11)$ and WG-16 stream ciphers using polynomial basis. IEEE Transactions on Computers **64**(7), 2020–2035 (2014)
29. Evans, R., Hollmann, H.D., Krattenthaler, C., Xiang, Q.: Gauss sums, jacobi sums, and p-ranks of cyclic difference sets. Journal of Combinatorial Theory, Series A **87**(1), 74–119 (1999)
30. Fan, X., Gong, G.: Specification of the stream cipher WG-16 based confidentiality and integrity algorithms, cacr report (2013). https://cacr.uwaterloo.ca/techreports/2013/cacr2013-06.pdf
31. Fan, X., Mandal, K., Gong, G.: WG-8: A lightweight stream cipher for resource-constrained smart devices. In International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness,p 617–632. Springer (2013)
32. Fan, X., Zidaric, N., Aagaard, M., Gong, G.: Efficient hardware implementation of the stream cipher WG-16 with composite field arithmetic. In Proceedings of the 3rd international workshop on Trustworthy embedded devices, p 21–34. (2013)
33. Fei, Y., Gong, G., Gongye, C., Mandal, K., Rohit, R., Xu, T., Yi, Y., Zidaric, N.: Correlation power analysis and higher-order masking implementation of WAGE. In International Conference on Selected Areas in Cryptography, p 593–614. Springer (2020)

34. Fu, S.: Gong G (2022) Polaris: Transparent succinct zero-knowledge arguments for R1CS with efficient verifier. Proc. Priv. Enhancing Technol **1**, 544–564 (2022)
35. The GAP Group: GAP - Groups. Algorithms, and Programming, Version **4**(12), 1 (2022)
36. Golomb, S.W.: Shift Register Sequences. Aegean Park Press, USA (1981)
37. Golomb, S.W., Gong, G.: Signal design for good correlation: for wireless communication, cryptography, and radar. Cambridge University Press (2005)
38. Gong, G., Aagaard, M., Fan, X.: Resilience to distinguishing attacks on WG-7 cipher and their generalizations. Cryptography and Communications **5**(4), 277–289 (2013)
39. Gong, G., Gaal, P., Golomb, S.: A suspected infinite class of cyclic hadamard difference sets. In Proceedings of 1997 IEEE Information Theory Workshop, pp 614–625. July 6-12, 1997, Longyearbyen, Svalbard, Norway (1997)
40. Gong, G., Golomb, S.W.: Transform domain analysis of DES. IEEE transactions on Information Theory **45**(6), 2065–2073 (1999)
41. Gong, G., Khoo, K.: Additive autocorrelation of resilient boolean functions. In International Workshop on Selected Areas in Cryptography, pp 275–290. Springer (2003)
42. Gong, G., Rønjom, S., Helleseth, T., Hu, H.: Fast discrete fourier spectra attacks on stream ciphers. IEEE Transactions on Information Theory **57**(8), 5555–5565 (2011)
43. Gong, G., Youssef, A.M.: On Welch-Gong transformation sequence generators. In International Workshop on Selected Areas in Cryptography, pp 217–232. Springer (2000)
44. Gong, G., Youssef, A.M.: Cryptographic properties of the Welch-Gong transformation sequence generators. IEEE Transactions on Information Theory **48**(11), 2837–2846 (2002)
45. Gordon, B., Mill, W., Welch, L.: Some new difference sets. Canadian J Math **14**(4), 614–625 (1962)
46. Grassi, L., Khovratovich, D., Rechberger, C., Roy, A., Schofnegger, M.: Poseidon: A new hash function for Zero-Knowledge proof systems. In 30th USENIX Security Symposium (USENIX Security 21). USENIX Association, pp 519–535. (2021)
47. Grassi, L., Lüftenegger, R., Rechberger, C., Rotaru, D., Schofnegger, M.: On a generalization of substitution-permutation networks: The HADES design strategy. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology - EUROCRYPT 2020, pp. 674–704. Springer International Publishing, Cham (2020)
48. Hall, M.: A survey of difference sets. Proceedings of the American Mathematical Society **7**(6), 975–986 (1956)
49. Herlestam, T.: On functions of linear shift register sequences. In: Pichler, F. (ed.) Advances in Cryptology – EUROCRYPT'85. Berlin, Heidelberg, Springer, Berlin Heidelberg (1986)
50. Joseph, M., Sekar, G., Balasubramanian, R.: Distinguishing attacks on (ultra-) lightweight WG ciphers. In International Workshop on Lightweight Cryptography for Security and Privacy, pp 45–59. Springer (2016)
51. Kaleem, M.K.: Physical layer approach for securing RFID systems. MASc thesis. uwspace (2013). http://hdl.handle.net/10012/7702
52. Kasami, T.: The weight enumerators for several classes of subcodes of the 2nd order binary reed-muller codes. Information and Control **18**(4), 369–394 (1971)
53. Kaur, J., Sarker, A., Kermani, M.M., Azarderakhsh, R.: Hardware constructions for error detection in lightweight Welch-Gong (WG)-oriented stream cipher WAGE benchmarked on FPGA. IEEE Transactions on Emerging Topics in Computing **10**(2), 1208–1215 (2021)
54. Krengel, E.: Fast WG stream cipher. In 2008 IEEE Region 8 International Conference on Computational Technologies in Electrical and Electronics Engineering, pp 31–35. (2008). IEEE
55. Lam, C.H.: Verification of pipelined ciphers. masc thesis. uwspace (2009). http://hdl.handle.net/10012/4267
56. Lam, C.H., Aagaard, M., Gong, G.: Hardware implementations of multi-output welch-gong ciphers, cacr report (2009). https://cacr.uwaterloo.ca/techreports/2011/cacr2011-01.pdf
57. Luo, Y., Chai, Q., Gong, G., Lai, X.: A lightweight stream cipher WG-7 for RFID encryption and authentication. In 2010 IEEE Global Telecommunications Conference GLOBECOM 2010, pp. 1–6 (2010). IEEE
58. Mandal, K.: Design and analysis of cryptographic pseudorandom number/sequence generators with applications in RFID. PhD thesis. uwspace (2013). http://hdl.handle.net/10012/7730
59. Mandal, K., Fan, X., Gong, G.: Design and implementation of Warbler family of lightweight pseudorandom number generators for smart devices. ACM Trans Embed Comput Syst 15(1) (2016)
60. Mandal, K., Gong, G.: Feedback reconstruction and implementations of pseudorandom number generators from composited de Bruijn sequences. IEEE Transactions on Computers **65**(9), 2725–2738 (2016)
61. Mandal, K., Gong, G.: Filtering nonlinear feedback shift registers using Welch-Gong transformations for securing RFID applications. EAI Endorsed Transactions on Security and Safety **3**(7), 12 (2016)

62. Mandal, K., Gong, G.: On ideal t-tuple distribution of orthogonal functions in filtering de Bruijn generators. Advances in Mathematics of Communications **16**(3), 597–619 (2022)
63. Mandal, K., Gong, G., Fan, X., Aagaard, M.: On selection of optimal parameters for the WG stream cipher family. In 2013 13th Canadian Workshop on Information Theory, pp 17–21 (2013). IEEE
64. Mandal, K., Gong, G., Fan, X., Aagaard, M.: Optimal parameters for the WG stream cipher family. Cryptography and Communications **6**(2), 117–135 (2014)
65. Mandal, K., Yang, B., Gong, G., Aagaard, M.: Analysis and efficient implementations of a class of composited de Bruijn sequences. IEEE Transactions on Computers **69**(12), 1835–1848 (2020)
66. Mascia, C., Piccione, E., Sala, M.: An algebraic attack on stream ciphers with application to nonlinear filter generators and WG-PRNG (2021). arXiv preprint arXiv:2112.12268
67. Mohajerani, K., Haeussler, R., Nagpal, R., Farahmand, F., Abdulgadir, A., Kaps, J.-P., Gaj, K.: FPGA benchmarking of round 2 candidates in the NIST lightweight cryptography standardization process: Methodology, metrics, tools, and results. Cryptology ePrint Archive, Paper 2020/1207 (2020). https://eprint.iacr.org/2020/1207
68. Mohajerani, K., Haeussler, R., Nagpal, R., Farahmand, F., Abdulgadir, A., Kaps, J.-P., Gaj, K.: Hardware benchmarking of round 2 candidates in the NIST lightweight cryptography standardization process. In 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp 164–169. (2021)
69. Mota, R.K.: Role of cryptographic Welch-Gong (WG-5) stream cipher in RFID security. masc thesis. uwspace (2012). http://hdl.handle.net/10012/6769
70. Nawaz, Y.: Design of stream ciphers and cryptographic properties of nonlinear functions. PhD thesis. uwspace (2007). http://hdl.handle.net/10012/3447
71. Nawaz, Y., Gong, G.: The WG stream cipher, estream submission (2005). http://www.ecrypt.eu.org/stream/p2ciphers/wg/wg_p2.pdf
72. Nawaz, Y., Gong, G.: WG: A family of stream ciphers with designed randomness properties. Information Sciences **178**(7), 1903–1916 (2008)
73. No, J.-S., Chung, H., Yun, M.-S.: Binary pseudorandom sequences of period $2^m - 1$ with ideal autocorrelation generated by the polynomial $z^d + (z + 1)^d$. IEEE Trans. Inf. Theory **44**(3), 1278–1282 (1998)
74. No, J.-S., Golomb, S.W., Gong, G., Lee, H.-K., Gaal, P.: Binary pseudorandom sequences of period $2^m - 1$ with ideal autocorrelation. IEEE Trans. Inf. Theory **44**(3), 1278–1282 (1998)
75. No, J.-S., Lee, H.-K., Chung, H., Song, H.-Y., Yang, K.: Trace representation of Legendre sequences of Mersenne prime period. IEEE Transactions on Information Theory **42**(6), 2254–2255 (1996)
76. Nyberg, K., Knudsen, L.R.: Provable security against a differential attack. J Cryptol **8**(1), 27–37 (1995)
77. Orumiehchiha, M.A., Pieprzyk, J., Steinfeld, R.: Cryptanalysis of WG-7: a lightweight stream cipher. Cryptography and Communications **4**(3), 277–285 (2012)
78. Orumiehchiha, M.A., Rostami, S., Shakour, E., Pieprzyk, J.: A differential fault attack on the WG family of stream ciphers. Journal of Cryptographic Engineering **10**(2), 189–195 (2020)
79. Philip, M.A., Vaithiyanathan: A survey on lightweight ciphers for IoT devices. In 2017 International Conference on Technological Advancements in Power and Energy (TAP Energy), pp. 1–4 (2017). IEEE
80. Rohit, R.: Design and cryptanalysis of lightweight symmetric key primitives. PhD thesis. uwspace (2020). http://hdl.handle.net/10012/15556
81. Rohit, R., AlTawy, R., Gong, G.: MILP-based cube attack on the reduced-round WG-5 lightweight stream cipher. In IMA International Conference on Cryptography and Coding, pp. 333–351. Springer (2017)
82. Rønjom, S.: Powers of subfield polynomials, cyclic codes and algebraic attacks with applications to the WG stream ciphers. In WCC2015-9th International Workshop on Coding and Cryptography 2015 (2015)
83. Rønjom, S.: Improving algebraic attacks on stream ciphers based on linear feedback shift register over $\mathbb{F}_{2^k}$. Designs, Codes and Cryptography **82**(1), 27–41 (2017)
84. Rostami, S., Shakour, E., Orumiehchiha, M.A., Pieprzyk, J.: Cryptanalysis of WG-8 and WG-16 stream ciphers. Cryptography and Communications **11**(2), 351–362 (2019)
85. Sattarov, M.: Hardware implementations of the lightweight Welch-Gong stream cipher family using polynomial bases. MASc thesis. uwspace (2019). http://hdl.handle.net/10012/14437
86. Scholtz, R., Welch, L.: GMW sequences (corresp.). IEEE Transactions on Information Theory **30**(3),548–553 (1984)
87. Tan, Y., Gong, G., Zhu, B.: Enhanced criteria on differential uniformity and nonlinearity of cryptographically significant functions. Cryptography and communications **8**(2), 291–311 (2016)
88. Wu, H., Preneel, B.: Resynchronization attacks on WG and LEX. In International Workshop on Fast Software Encryption, pp. 422–432. Springer (2006)

89. Wu, T.: On message authentication in 4G LTE system. PhD thesis. uwspace (2015). http://hdl.handle.net/10012/9601
90. Yang, B., Mandal, K., Aagaard, M.D., Gong, G.: Efficient composited de Bruijn sequence generators. IEEE Transactions on Computers **66**(8), 1354–1368 (2017)
91. Yang, G.: Optimized hardware implementations of lightweight cryptography. PhD thesis. uwspace (2017). http://hdl.handle.net/10012/11237
92. Yang, G., Fan, X., Aagaard, M., Gong, G.: Design space exploration of the lightweight stream cipher WG-8 for FPGAs and ASICs. In Proceedings of the Workshop on Embedded Systems Security, pp. 1–10. (2013)
93. Yi, Y., Gong, G., Mandal, K.: Implementation of three LWC schemes in the WiFi 4-way handshake with software defined radio. NIST LWC workshop 2019, arXiv preprint (2019). arXiv:1909.11707
94. Yi, Y., Mandal, K., Gong, G.: Implementation of lightweight ciphers and their integration into entity authentication with IEEE 802.11 physical layer transmission. In International Symposium on Foundations and Practice of Security, pp. 113–129. Springer (2022)
95. Youssef, A.M., Gong, G.: On the interpolation attacks on block ciphers. In International Workshop on Fast Software Encryption, pp. 109–120. Springer (2000)
96. Yu, N.Y.: On periodic correlation of binary sequences. PhD thesis. uwspace (2007). http://hdl.handle.net/10012/2634
97. Zhang, B.Y., Gong, G.: Randomness properties of stream ciphers for wireless communications. In The Sixth International Workshop on Signal Design and Its Applications in Communications, pp. 107–109. (2013). IEEE
98. Zidaric, N.: Hardware implementations of the WG-16 stream cipher with composite field arithmetic. MASc thesis. uwspace (2014). http://hdl.handle.net/10012/8844
99. Zidaric, N.: Automated design space exploration and datapath synthesis for finite field arithmetic with applications to lightweight cryptography. PhD thesis. uwspace (2020). http://hdl.handle.net/10012/15928
100. Zidaric, N., Aagaard, M., Gong, G.: Hardware optimizations and analysis for the WG-16 cipher with tower field arithmetic. IEEE Transactions on Computers **68**(1), 67–82 (2018)
101. Zidaric, N., Aagaard, M., Gong, G.: Rapid hardware design for cryptographic modules with filtering structures over small finite fields. In International Workshop on the Arithmetic of Finite Fields, pp. 128–145. Springer (2018)
102. Zidaric, N., Aagaard, M., Gong, G.: FSR, feedback shift register package, Version 1.2.2. GAP package (2019)