Faster Correlation Attack on Bluetooth Keystream Generator E0

Yi Lu* and Serge Vaudenay

EPFL
http://lasecwww.epfl.ch

Abstract. We study both distinguishing and key-recovery attacks against E0, the keystream generator used in Bluetooth by means of correlation. First, a powerful computation method of correlations is formulated by a recursive expression, which makes it easier to calculate correlations of the finite state machine output sequences up to 26 bits for E0 and allows us to verify the two known correlations to be the largest for the first time. Second, we apply the concept of convolution to the analysis of the distinguisher based on all correlations, and propose an efficient distinguisher due to the linear dependency of the largest correlations. Last, we propose a novel maximum likelihood decoding algorithm based on fast Walsh transform to recover the closest codeword for any linear code of dimension L and length n. It requires time $O(n + L \cdot 2^L)$ and memory $\min(n, 2^L)$. This can speed up many attacks such as fast correlation attacks. We apply it to E0, and our best key-recovery attack works in 2^{39} time given 2^{39} consecutive bits after $O(2^{37})$ precomputation. This is the best known attack against E0 so far.

1 Background

Correlation properties play an important role in the security of nonlinear LFSR-based combination generators in stream ciphers. As name implies, the word *correlation* in stream ciphers is frequently referred to as the intrinsic relation between the keystream and a subset of the LFSR subsequences. The earliest studies dated back to [21, 25, 27] in the 80's and the concept of correlation immunity was proposed as a security criterion. In the 90's Meier-Staffelbach [22] analyzed correlation properties of combiners with one memory bit, followed by Golić [12] focusing on correlation properties of a general combiner with m-bit memory. Recently, a series of fast correlation attacks sprang up, to name but a few [5–7, 16, 24]. Thereupon we dedicate this paper to the generalized correlation attacks against E0, a combiner with 4-bit memory used in the short-range wireless technology Bluetooth. Prior to our work, existed various attacks [1, 8, 10, 11, 13–15, 17, 26] against E0. The best key-recovery attacks are algebraic attacks [1, 8], whose basic approach is to use the polynomial canceling all memory bits and involving only key bits, instead of considering the multiple polynomial to cancel the key bits in the distinguishing attack; besides, [9, 13, 14] discussed correlations of E0. In [14], Hermelin-Nyberg for the first time presented a rough computation method to compute the correlation (called bias for our purpose), but neither did they formalize the computation systematically, nor did they attempt to find a larger correlation. In [9, 13], two larger correlations for a short sequence of up to 6 bits were exposed. However, due to the limit of the computation method, no one was certain about the existence of a larger correlation for a longer sequence, which is critical to the security of E0.

* supported in part by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center of the Swiss National Science Foundation under the grant number 5005-67322.

Our first contribution in the paper is that based on Hermelin-Nyberg [14] we formulate a powerful computation method of correlations by a recursive expression, which makes it easier to calculate correlations of the Finite State Machine (FSM) output sequences up to 26 bits for E0 (and allows us to prove the two known correlations to be the only largest for the first time). Second, we apply the concept of convolution to the analysis of the distinguisher based on all correlations, which allows us to build an efficient distinguisher that halves the data complexity of the basic uni-bias-based distinguisher due to the linear dependency of the two largest biases. Our best distinguishing attack takes 2^{43} time given 2^{43}-bit keystream with $O(2^{45})$ precomputation[1]. Finally, by means of Fast Walsh Transform (FWT), we propose a novel Maximum Likelihood Decoding (MLD) algorithm to recover the closest codeword for any linear code. Our proposed algorithm can be easily applied to speed up a class of fast correlation attacks. Furthermore the algorithm is optimal when the length n of the code and the dimension L satisfy the relation $n \geq 2^L$, which is the case when we apply it to recover R_1 for E0. Our best key-recovery attack works in 2^{39} time given 2^{39} consecutive bits after $O(2^{37})$ precomputation. Compared with the minimum time complexity $O(2^{49})$ in algebraic attacks [1, 8], this is the best known attack against E0.

This paper is structured as follows: in Section 2, a description of E0 is given. In Section 3, we analyze the bias inside E0 systematically. Then based on one largest bias, we build a primary distinguisher for E0 in Section 4; an efficient way is shown in Section 5 that makes full use of all the largest biases to advance the distinguisher. In Section 6 we investigate the MLD algorithm for a linear code; the result is then applied to a key-recovery attack against E0 in Section 7. Finally we conclude in Section 8.

2 Description of the Bluetooth Keystream Generator E0

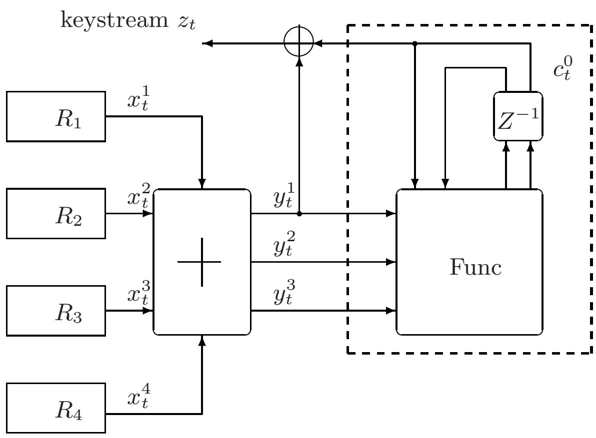

Fig. 1. Outline of E0

[1] Throughout this paper, $O(\cdot)$ is used to provide a rough estimate on complexities, eg. $O(2^{45})$ here means $c \cdot 2^{45}$ operations, where c is a small constant.

As specified in [3], the keystream generator E0 used in Bluetooth belongs to a combination generator with four memory bits[2], denoted by $\sigma_t = (c_{t-1}, c_t)$ at time t, where $c_t = (c_t^1, c_t^0)$. The whole system (Fig.1) uses four Linear Feedback Shift Registers (LFSRs) denoted by $R_1, \ldots, R_4$ with lengths $L_1 = 25$, $L_2 = 31$, $L_3 = 33$, $L_4 = 39$ and primitive feedback polynomials

$$\begin{aligned} p_1(x) &= x^{25} + x^{20} + x^{12} + x^8 + 1, \\ p_2(x) &= x^{31} + x^{24} + x^{16} + x^{12} + 1, \\ p_3(x) &= x^{33} + x^{28} + x^{24} + x^4 + 1, \\ p_4(x) &= x^{39} + x^{36} + x^{28} + x^4 + 1, \end{aligned}$$

respectively. At clock cycle t, the four LFSRs' output bits x_t^i, $i = 1, \ldots, 4$, will be added as integers. The sum $y_t \in \{0, \ldots, 4\}$ is represented in the binary system. Let y_t^i denote its i-th least significant bit ($i = 1, 2, 3$). A 16-state machine (the dashed box in Fig.1) emits one bit c_t^0 out of its state $\sigma_t = (c_{t-1}, c_t)$ and takes the input y_t to update σ_t by σ_{t+1}. Finally, the keystream z_t is obtained by xoring y_t^1 with c_t^0. That is,

$$x_t^1 \oplus x_t^2 \oplus x_t^3 \oplus x_t^4 \oplus c_t^0 = z_t. \tag{1}$$

The detailed mechanism of the FSM is beyond the scope of the paper except the fact that the embedded delay cell (the box labeled Z^{-1} in Fig.1) makes c_t^0 depend only on the initial state σ_0 of the FSM as well as the past vectors $y_{t-1}, y_{t-2}, \ldots, y_0$. For completeness, we briefly outline it: given y_t together with the state σ_t, the FSM moves into the state σ_{t+1}. Table 1 shows the state transition of the FSM, where the four-bit state is represented in the quaternary system (e.g. the FSM changes from $\sigma_t = 13$ into $\sigma_{t+1} = 32$ by the input $y_t = 2$).

More formally, by introducing two temporary bits $s_{t+1} = (s_{t+1}^1, s_{t+1}^0)$ each clock, the following indirect iterative expressions between s_{t+1} and c_{t+1} suffice to update c_t:

$$s_{t+1} = \left\lfloor \frac{y_t + 2 \cdot c_t^1 + c_t^0}{2} \right\rfloor, \tag{2}$$

$$c_{t+1}^1 = s_{t+1}^1 \oplus c_t^1 \oplus c_{t-1}^0, \tag{3}$$

$$c_{t+1}^0 = s_{t+1}^0 \oplus c_t^0 \oplus c_{t-1}^1 \oplus c_{t-1}^0. \tag{4}$$

One can check Table 1 by those equations. We denote λ_t hereafter the content of LFSRs at time t. Then the state of E0 at time t is fully represented by the pair (λ_t, σ_t).

Table 1. State transition of σ_{t+1} given y_t and σ_t

	σ_t															
	00	01	02	03	10	11	12	13	20	21	22	23	30	31	32	33
y_t = 0	00	11	23	32	03	12	20	31	01	10	22	33	02	13	21	30
1	00	10	23	31	03	13	20	32	01	11	22	30	02	12	21	33
2	01	10	20	31	02	13	23	32	00	11	21	30	03	12	22	33
3	01	13	20	30	02	10	23	33	00	12	21	31	03	11	22	32
4	02	13	21	30	01	10	22	33	03	12	20	31	00	11	23	32

[2] The description of E0 (sometimes called one-level E0) here only involves the keystream generation after the initialization.

3 Biases Inside E0

Property 1. Assuming $y_t = 2$ holds for $t = t_0, t_0+1, \ldots, t_0+3$, then

$$c^0_{t_0} \oplus c^0_{t_0+1} \oplus c^0_{t_0+2} \oplus c^0_{t_0+3} \oplus c^0_{t_0+4} = 1.$$

Proof. It's easy to verify that the state transition given $y_t = 2$ (the third bottom row in Table 1) is indeed a linear transformation over $GF(2)^4$, that actually satisfies the recurrence relation: $\sigma_{t+1} = A \times \sigma_t \oplus 01$, where states σ_{t+1} are represented by column vectors of $(c^1_t, c^0_t, c^1_{t+1}, c^0_{t+1})$, and A is the following 4×4 square matrix over $GF(2)$:

$$A = \begin{pmatrix} 0\,0\,1\,0 \\ 0\,0\,0\,1 \\ 0\,1\,0\,0 \\ 1\,1\,1\,1 \end{pmatrix}.$$

Note that $x^4 + x^3 + x^2 + x + 1$ is the minimal polynomial of A, from which we deduce $\sigma_{t_0} \oplus \sigma_{t_0+1} \oplus \sigma_{t_0+2} \oplus \sigma_{t_0+3} \oplus \sigma_{t_0+4} = 33$. □

Remark 2. Since $\Pr(y_t = 2) = \frac{6}{16}$, this seemingly suggests that

$$\Pr(c^0_t \oplus c^0_{t+1} \oplus c^0_{t+2} \oplus c^0_{t+3} \oplus c^0_{t+4} = 1) \approx \frac{1}{2} + (\frac{6}{16})^4 = \frac{1}{2} + \frac{81}{4096}.$$

As mentioned in [9, 13] (without relating to the above special case), this bit exhibits a much higher bias as shown later in Corollary 7. We will now introduce essential material in order to find a systematic algorithm to compute biases.

Proposition 3. *If (λ_0, σ_0) is random and uniformly distributed, then for any t*

- (λ_t, σ_t) *is random and uniformly distributed,*
- $(c_t, c_{t-1}, \ldots, c_{t-24})$ *is independent of y_t.*

Proof (sketch). The former half of the theorem is justified by the fact that (λ_t, σ_t) is a permutation of (λ_0, σ_0) for any t. About the latter half of theorem, first, we know that $(\lambda_{t-24}, \sigma_{t-24})$ is random and uniformly distributed by previous conclusion. Thus, $y_{t-24}, \ldots, y_{t-1}$ are i.i.d. random variables all independent of both σ_{t-24} and y_t. By Eq.(2,3,4), we complete the proof. □

Interestingly, we deduce that if (λ_0, σ_0) is uniformly distributed, then any sequence of 39-bit consecutive E0 keystream is uniformly distributed; in particular, no better key-recovery attack against E0 exists other than tradeoffs given a sequence of 39-bit consecutive keystream.

The following definition is derived from *normalized correlation* [22, p.71].

Definition 4. *The bias of a random Boolean variable X is defined as*

$$\Delta(X) = \Pr(X = 0) - \Pr(X = 1) = E[(-1)^X].$$

The normalized correlation between two random Boolean variables X and Y is just the bias of $X \oplus Y$. Assuming that y_t is the sum of four balanced independent random bits and that c_t is uniformly distributed, then we know that $\Delta(a \cdot s_{t+1} \oplus w \cdot c_t)$ is a constant for any $a, w \in GF(2)^2$, denoted by $\Omega(a, w)$. Table 2 shows $\Omega(a, w)$ computed by Eq.(2), where dashed entries are zeros. The following important lemma (see Appendix A for proof) inspired by [14], gives an easy way of computing the bias for iterative structures.

Table 2. Bias of all linear combination of s_{t+1} and c_t: $\Omega(a, w)$

$\Omega(a,w)$		w = 0	1	2	3
a	0	-	-	-	-
	1	-	-	$-1/4$	-
	2	-	$1/4$	$5/8$	-
	3	$-5/8$	-	-	$1/4$

Lemma 5. *Given $f : \mathcal{E} \times GF(2)^k \to GF(2)$ and $g : GF(2)^m \to GF(2)^k$, let X and Y be two independent random variables in $\mathcal{E}$ and $GF(2)^m$ respectively. Assuming that $g(Y)$ is uniformly distributed in $GF(2)^k$, for any $v \in GF(2)^m$, we have*

$$\Delta(f(X, g(Y)) \oplus v \cdot Y) = \sum_{w \in GF(2)^k} \Delta(f(X, g(Y)) \oplus w \cdot g(Y)) \cdot \Delta(w \cdot g(Y) \oplus v \cdot Y).$$

Corollary 6. *We set $h : (x^1, x^0) \mapsto (x^0, x^1 \oplus x^0)$ to be a permutation defined over $GF(2)^2$, and $\delta(a_1, \ldots, a_d) = \Delta(a_1 \cdot c_1 \oplus \cdots \oplus a_d \cdot c_d)$, where $a_1, \ldots, a_d \in GF(2)^2$. Assuming (λ_0, σ_0) is uniformly distributed, for any $d \leq 26$, we have*

$$\delta(a_1, \ldots, a_d) = \sum_{w \in GF(2)^2} \Omega(a_d, w) \cdot \delta(a_1, \ldots, a_{d-3}, a_{d-2} \oplus h(a_d), a_{d-1} \oplus a_d \oplus w).$$

Proof. By Eq.(2,3,4) we have

$$\delta(a_1, \ldots, a_d) = \Delta(a_d \cdot s_d \oplus a_1 \cdot c_1 \oplus \cdots \oplus (a_{d-2} \oplus h(a_d)) \cdot c_{d-2} \oplus (a_{d-1} \oplus a_d) \cdot c_{d-1}).$$

Then we apply Lemma 5 with $X = y_{d-1}$, $Y = (c_1, \ldots, c_{d-1})$, $g(Y) = c_{d-1}$, $f(X, g(Y)) = a_d \cdot s_d$ and $v = (a_1, \ldots, a_{d-3}, a_{d-2} \oplus h(a_d), a_{d-1} \oplus a_d)$, and we obtain

$$\delta(a_1, \ldots, a_d) = \sum_w \Omega(a_d, w) \delta(a_1, \ldots, a_{d-3}, a_{d-2} \oplus h(a_d), a_{d-1} \oplus a_d \oplus w).$$

Note that the assumption of Lemma 5 holds by Proposition 3. □

Now we use Corollary 6 iteratively to deduce some important biases of $\{c_t^0\}$ with Table 2 and the initial values $\delta(0, 0) = 1$, and $\delta(a, b) = 0$ for $(a, b) \neq (0, 0)$. A full list of nonzero triplets is given below for illustration:

$\delta(0,0,0) = 1$, $\delta(1,3,2) = \frac{1}{4}$, $\delta(2,3,3) = -\frac{5}{8}$,
$\delta(1,0,2) = \frac{5}{8}$, $\delta(2,0,3) = \frac{1}{4}$, $\delta(3,3,1) = -\frac{1}{4}$.

Corollary 7. *Assuming (λ_0, σ_0) is random and uniformly distributed, we have*

$$\Pr(c_t^0 \oplus c_{t+1}^0 \oplus c_{t+2}^0 \oplus c_{t+3}^0 \oplus c_{t+4}^0 = 1) = \frac{1}{2} + \frac{25}{512},$$
$$\Pr(c_t^0 = c_{t+5}^0) = \frac{1}{2} + \frac{25}{512}.$$

Note that both biases were mentioned in [9, 13] (without formal proof). Now by Corollary 6, we can easily prove it as shown next.

Proof. We show the equivalent $\delta(1,1,1,1,1) = -\frac{25}{256}$ of the first bias as follows:

$$\begin{aligned}\delta(1,1,1,1,1) &= \sum_w \Omega(1,w)\cdot\delta(1,1,2,w)\\ &= -\frac{1}{4}\delta(1,1,2,2)\\ &= -\frac{1}{4}\sum_w \Omega(2,w)\cdot\delta(1,0,w)\\ &= -\frac{1}{4}\times\left(\frac{1}{4}\delta(1,0,1)+\frac{5}{8}\delta(1,0,2)\right)\\ &= -\frac{25}{256}.\end{aligned}$$

The second bias is similarly proved from $\delta(1,0,0,0,0,1) = \frac{25}{256}$. □

Also, we computed all ℓ-tuple biases for $\ell \leq 26$ and found that $\delta(1,1,1,1,1)$, $\delta(1,0,0,0,0,1)$ are the only largest ones. All biases for $\ell = 6$ are listed in Table 14, Appendix C. Throughout the paper we let

$$\gamma = \delta(1,0,0,0,0,1) = -\delta(1,1,1,1,1) = \frac{25}{256}.$$

4 A Primary Distinguisher for E0

4.1 The Connection Polynomial of the Equivalent Single LFSR

Let θ_i be the order of the connection polynomial $p_i(x)$ of R_i, for $i = 1,2,3,4$. Since all $p_i(x)$ are primitive polynomials, $\theta_i = 2^{L_i} - 1$; furthermore, by Lemma 6.57 of [18, p.218], the equivalent LFSR to generate the same sequence of the sum of the four original LFSRs outputs over GF(2) has the connection polynomial $\prod_{i=1}^{4} p_i(x)$ with order $\theta = \text{lcm}(\theta_1,\theta_2,\theta_3,\theta_4) \approx 2^{125}$ (by Lemma 6.50, [18, p.214]) and degree $L = \sum_{i=1}^{4} L_i = 128$.

4.2 Finding the Multiple Polynomial with Low Weight

Let d_0 be the degree of a general polynomial $p(x)$. We use the standard approximation to estimate the minimal weight w_d of multiples of $p(x)$ with degree at most d by the following constraint: w_d is the smallest w such that

$$\frac{1}{2^{d_0}} \times \binom{d}{w-1} \geq 1. \tag{5}$$

Listed in Table 3 is the estimated[3] w_d corresponding to d with $p(x) = \prod_{i=1}^{4} p_i(x)$ ($d_0 = 128$) by solving Inequality (5).

To find multiples with low weight, efficient algorithms like [4] exist provided the degree is low, say, less than 2000, which does not apply to E0. So we can use the conventional birthday paradox to find $Q(x)$ with the minimal d (i.e. $w = w_d$), which takes precomputation time $PT \approx O(d^{\lceil \frac{w-1}{2} \rceil})$; or we apply the generalized birthday problem [29] to find $Q(x)$ of same weight but higher degree with much less precomputation as tradeoff. Table 4 compares the two algorithms. In Appendix B, we also provide some non-optimal multiples as examples, including $Q_4(x)$ with $w = 4$ and $d \approx 2^{65}$.

[3] Two special cases occur for $d = d_0$ and $d = \theta$ because we know the exact value of w_d.

Table 3. The estimated minimal weight w_d of multiples of $p_1(x)p_2(x)p_3(x)p_4(x)$ with degree d by (5)

d	128	247	458	855	1749	2387	2^{18}	2^{23}	2^{27}	2^{33}	2^{44}	2^{65}	θ
w_d	$= 49$	≈ 31	≈ 24	≈ 20	≈ 17	≈ 16	≈ 9	≈ 7	≈ 6	≈ 5	≈ 4	≈ 3	$= 2$

Table 4. Complexity of finding multiple of $p_1(x)p_2(x)p_3(x)p_4(x)$ with degree d, weight w

	birthday problem							
	with minimal d						tradeoff	
d	2^{18}	2^{23}	2^{27}	2^{33}	2^{44}	2^{65}	2^{32}	2^{43}
w	9	7	6	5	4	3	9	5
$\log_2 PT$	72	69	68	66	66	65	35	45

4.3 Building a Uni-bias-based Distinguisher for E0

Let $Q(x) = \sum_{i=1}^{w} x^{q_i}$ be the normalized multiple of $\prod_{i=1}^{4} p_i(x)$ with degree d and weight w, where $0 = q_1 < q_2 < \ldots < q_w = d$. As $\oplus_{i=1}^{w} y^1_{t_0+q_i} = 0$ holds for all t_0, by Eq.(1), we deduce

$$\oplus_{i=1}^{w}(z_{t_0+q_i+5} \oplus z_{t_0+q_i}) = \oplus_{i=1}^{w}(c^0_{t_0+q_i+5} \oplus c^0_{t_0+q_i}). \tag{6}$$

By the Piling-up Lemma [20] and Corollary 7, we know the right-hand side of Eq.(6) is equal to zero with probability $\frac{1}{2} + \frac{1}{2} \cdot \gamma^w$. With standard linear cryptanalysis techniques, we can distinguish the keystream $\{z_t\}$ of E0 from a truly random sequence with $\gamma^{-2 \cdot w}$ samples, simply by checking the left-hand side of Eq.(6) equals zero most of the time. Based on $Q(x)$ with d and w, we minimize the data complexity n by choosing $n = \gamma^{-2 \cdot w} + d$. Table 5 shows the minimum $n = 2^{34}$ is achieved with $d = 2^{33}$, $w = 5$. Table 6 summarizes the best performance of our primary distinguisher for E0 based on either the use of $Q_4(x)$ with weight 4 in Appendix B, or a search of $Q(x)$.

Table 5. Data complexity of the primary distinguisher for E0

d	L	247	458	855	1749	2387	2^{18}	2^{23}	2^{27}	2^{33}	2^{44}	2^{65}	2^{32}	2^{43}
w	49	31	24	20	17	16	9	7	6	5	4	3	9	5
$\log_2 n$	329	209	162	135	115	108	61	47	41	34	44	65	61	43

Table 6. Summary of the best primary distinguisher for E0

Type		d	w	Precomputation	Data	Time
use $Q(x) = Q_4(x)$		2^{65}	4	-	2^{65}	
find $Q(x)$ with	minimal d	2^{33}	5	2^{66}	2^{34}	
	tradeoff	2^{43}	5	2^{45}	2^{43}	

5 The Advanced Multi-bias-based Distinguisher for E0

5.1 Preliminaries

Definition 8. *Given* $f, g : GF(2)^\ell \to \mathbf{R}$, *for* $a \in GF(2)^\ell$, *we define*

1. $(f \otimes g)(a) = \sum_{b \in GF(2)^\ell} f(b) \cdot g(a \oplus b)$; $f^{\otimes w}(a) = (\underbrace{f \otimes \cdots \otimes f}_{w \text{ times}})(a)$
2. $\hat{f}(a) = \sum_{b \in GF(2)^\ell} (-1)^{a \cdot b} f(b)$
3. $\|f\| = \sqrt{\sum_{a \in GF(2)^\ell} f^2(a)}$
4. $\Delta(f) = 2^{\frac{\ell}{2}} \|f - \frac{1}{2^\ell} \cdot \mathbf{1}\|$, *where* $\mathbf{1}$ *denotes a constant function equal to* 1

Note that the first two definitions correspond to convolution and Walsh transform respectively. We recall these basic facts: for any $f, g : GF(2)^\ell \to \mathbf{R}$, we have

- $\widehat{f \otimes g}(a) = \hat{f}(a) \cdot \hat{g}(a)$, for $a \in GF(2)^\ell$;
- $2^\ell \|f\|^2 = \|\hat{f}\|^2$;
- if f is a distribution, i.e. $\sum_a f(a) = 1$ and $f(a) \geq 0$ for all $a \in GF(2)^\ell$, then the distribution of the XOR of w i.i.d. random vectors with distribution f is $f^{\otimes w}$, moreover, $\Delta^2(f) = \sum_{a \neq \mathbf{0}} \hat{f}^2(a)$;
- If the random Boolean variable A follows the distribution f, then $\Delta(f) = \Delta(A)$, where $\Delta(A)$ is defined in Definition 4.

5.2 An Efficient Way to Deploy Multi-Biases in E0 Simultaneously

Given a linear mapping $h : GF(2)^\ell \to GF(2)^r$ of rank r, we define r-bit vectors $A_t = h(c^0_{rt}, \ldots, c^0_{rt+\ell-1})$ and $B_t = \oplus_{i=1}^w A_{t+q_i}$. Note that B_t can be derived from the keystream $\{z_t\}$ directly. Except for accidentally bad choices of h, we make a heuristic assumption that all A_t's are independent. Let $\mathcal{D}$ be the probability distribution of the ℓ-bit vector $(c^0_{rt}, \ldots, c^0_{rt+\ell-1})$, and let $\mathcal{D}_A$ be the probability distribution of the r-bit vector A_t. The Walsh transforms of $\mathcal{D}_A$ and $\mathcal{D}$ are linked by

$$\hat{\mathcal{D}}_A(b) = \hat{\mathcal{D}}\left(h^t(b)\right), \text{ for all b} \in GF(2)^r.$$

Now we discuss how to design h in order to reduce data complexity. From Baignères [2, Theorem 3, p.10], we know that we can distinguish a distribution f of r-bit random vectors from a uniform distribution with $1/\Delta^2(f)$ samples. Here, the distribution of B_t is $f = \mathcal{D}_A^{\otimes w}$. So the modified distinguisher needs data complexity

$$n = \frac{r}{\Delta^2(\mathcal{D}_A^{\otimes w})} + d \text{ (bits)}.$$

Let k be the number of the largest Walsh coefficients $\hat{\mathcal{D}}_A(b)$ over all nonzero b with absolute value[4] η. Since $\Delta^2(\mathcal{D}_A^{\otimes w}) \approx k\eta^{2w}$, we obtain

$$n \approx \frac{r}{k}\eta^{-2w} + d.$$

In order to lower n, it's necessary to have $r < k$. This implies the k largest coefficients are linearly dependent, which happens to be true in E0: recall that the 6-bit vectors of the three largest biases satisfy the linear relation,

$$(1, 1, 1, 1, 1, 0) \oplus (0, 1, 1, 1, 1, 1) = (1, 0, 0, 0, 0, 1).$$

As a simple solution we may just pick $\ell = 6$, $r = 2$, $\alpha_1 = (1, 1, 1, 1, 1, 0)$ and $\alpha_2 = (0, 1, 1, 1, 1, 1)$ (where α_i denotes the i-th row of h), then we obtain $k = 3$. And

[4] Note that from Subsection 4.3 we have $\eta \leq \gamma$ for $\ell \leq 26$ regardless of r and h.

n is reduced to a factor of $\frac{2}{3}$ for negligible d. Indeed, recall that we proved by computation that the largest Walsh coefficient for $\ell \leq 26$ are either (0,. . .,0,1,1,1,1,1,0,. . .,0) or (0,. . .,0,1,0,0,0,0,1,0,. . .,0). Thus $k \leq (\ell - 4) + (\ell - 5) = 2\ell - 9$. This leads to a more general solution, if we pick $\ell = r + 4$, and the i-th row of h as

$$\alpha_i = (\underbrace{0, \ldots, 0}_{i-1 \text{ zeros}}, 1, 1, 1, 1, 1, \underbrace{0, \ldots, 0}_{\ell-i-4 \text{ zeros}}) \text{ for } i = 1, \ldots, r,$$

then we obtain $k = 2r - 1$. And so the improved factor $\frac{r}{2r-1}$ of data complexity tends to $\frac{1}{2}$ for negligible d when r goes to infinity; however, because of the underlying assumption for E0, ℓ is restricted to no larger than 26, i.e. $r \leq 22$. To conclude, we show that the modified distinguisher (Algorithm 1) needs data complexity

$$n \approx \frac{r}{2r-1} \cdot \gamma^{-2w} + d, \text{ for } 1 \leq r \leq 22. \tag{7}$$

Observe that Section 4 actually deals with the special case of $r = 1$. Table 7 shows the best improvement achieved with $r = 22$. We see that the minimum n drops from previous 2^{34} to 2^{33}.

Algorithm 1 The advanced distinguisher for E0

Parameters:
  $r \in [1, 22], \ell = r + 4$
  $h : GF(2)^\ell \rightarrow GF(2)^r$
  $\mathcal{D}_A$: the probability distribution of the r-bit vector A_t
  $Q(x) = \sum_{i=1}^{w} x^{q_i}$: the multiple polynomial of $p_1(x)p_2(x)p_3(x)p_4(x)$ with degree d
  n: the sample size by Eq.(7)
Input:
  keystream $z_0 z_1 \cdots z_{n-1}$ of either a truly random source $\mathcal{S}_0$ or the output $\mathcal{S}_1$ generated by E0
  initialize counters $u_0, u_1, \ldots, u_{2^r-1}$
  for $t = 0, 1, \ldots, \lfloor \frac{n-d-4}{r} \rfloor - 1$ **do**
    compute $b = \oplus_{i=1}^{w} h(z_{rt+q_i}, \cdots, z_{rt+q_i+\ell-1})$
    increment u_b
  end for
  if $\sum_b u_b \cdot \log\left(2^r \cdot \mathcal{D}_A^{\otimes w}(b)\right) > 0$ **then**
    accept $\mathcal{S}_1$ as the source
  else
    accept $\mathcal{S}_0$ as the source
  end if

Table 7. Data complexity of the advanced distinguisher for E0

d	L	247	458	855	1749	2387	2^{18}	2^{23}	2^{27}	2^{33}	2^{44}	2^{65}	2^{32}	2^{43}
w	49	31	24	20	17	16	9	7	6	5	4	3	9	5
$\log_2 n$	328	208	161	134	114	107	60	46	40	33	44	65	60	43

6 A Maximum Likelihood Decoding Algorithm

We restate the MLD problem for a general linear code (see [19] for details) of length n and dimension L with generator matrix G (let G_t denote the t-th column vector of G): find the closest codeword $(x_1, \ldots, x_n)$ to the received vector $(s_1, \ldots, s_n)$, and decode the message $\mathbf{r} = (r_1, \ldots, r_L)$ such that $x_t = \mathbf{r}G_t$, i.e. find such $\mathbf{r}$ that minimizes $N(\mathbf{r}) = \sum_{t=1}^{n}(s_t \oplus x_t)$.

6.1 The Time-domain Analysis

Obviously, the trivial approach (yet common in most correlation attacks) to find $\mathbf{r}$ is an exhaustive search in time-domain: for every message $\tilde{\mathbf{r}}$, we compute the distance $N(\tilde{\mathbf{r}})$ and keep the smallest. The final record leads to $\mathbf{r}$. The time complexity is $O(n \cdot 2^L)$ with memory n-bits.

6.2 The Frequency-domain Analysis

We introduce an integer-valued function,

$$\mathcal{W}(x) = \sum_{1 \leq t \leq n : G_t = x^\top} (-1)^{s_t}, \tag{8}$$

for all $x \in GF(2)^L$, where $\top$ denotes the matrix transpose. We compute the Walsh transform $\hat{\mathcal{W}}$ of $\mathcal{W}$ as follows:

$$\hat{\mathcal{W}}(\mathbf{r}) = \sum_{x \in GF(2)^L} (-1)^{\mathbf{r} \cdot x} \mathcal{W}(x) = \sum_{t=1}^{n} (-1)^{s_t \oplus \mathbf{r}G_t} = \sum_{t=1}^{n} (-1)^{s_t \oplus x_t} = n - 2N(\mathbf{r}).$$

We thereby reach the theorem below.

Theorem 9.

$$N(\mathbf{r}) = \frac{1}{2}\left(n - \hat{\mathcal{W}}(\mathbf{r})\right),$$

for all $\mathbf{r} \in GF(2)^L$*, where* $\mathcal{W}$ *is defined by Eq.(8).*

This generalizes the result [19, p. 414] of a special case when $n = 2^L$ and $G_t^\top$ corresponds to the binary representation of t. So we just compute the table of $\mathcal{W}$, perform FWT [30], and find the maximal $\hat{\mathcal{W}}(\mathbf{r})$. The time and memory complexities of FWT are $O(L \cdot 2^L)$, $O(2^L)$ respectively. Since the precomputation of $\mathcal{W}$ takes time $O(n)$ with memory $O(n)$, we conclude that our improved MLD algorithm runs in $O(n + L \cdot 2^L)$ with memory $O(2^L)$ (additionally, using linear transformation allows to compute FWT over $GF(2)^k$ with memory $O(2^k)$ where $k = \lceil \log_2 n \rceil$). Note that when $n \geq 2^L$, the time complexity corresponds to $O(n)$, which is optimal in the sense that it stands on the same order of magnitude as the data complexity does. Table 8 compares the original exhaustive search algorithm with the improved frequency transformation algorithm. Note that the technique of FWT was used in another context [7] to speed up other kinds of fast correlation attacks. In the next section we will see how it helps to speed up the attack [10] by a factor of 2^{24}. We estimate similar correlation attacks like [6] can be speeded up by a factor of 10; undoubtedly, some other attacks can be significantly improved by our algorithm as well.

Table 8. Maximum likelihood decoding algorithms

	time	memory
Exhaustive Search	$n \cdot 2^L$	n
Frequency Transformation	$n + L \cdot 2^L$	$\min(n, 2^L)$

6.3 A More Generalized MLD Algorithm

We further generalize the preceding problem by finding the L-bit vector $\mathbf{r}$ such that given a sequence of ℓ-bit vectors $S_1, \ldots, S_k$ and $f : GF(2)^\ell \to \mathbf{R}$ together with matrices $G_1, \ldots, G_k$ of size L by ℓ, the sequence of ℓ-bit vectors $X_1, \ldots, X_k$ defined by $X_t = \mathbf{r}G_t$ minimizes $N(\mathbf{r}) = \sum_{t=1}^{k} f(S_t \oplus X_t)$. Note that previous subsections are merely a special case of $\ell = 1$, $k = n$ and $f(a) = a$ for $a \in GF(2)$.

Define a real function $\mathcal{W}$ by:

$$\mathcal{W}(x) = \frac{1}{2^\ell} \sum_{1 \le t \le k, a \in GF(2)^\ell : aG_t^\top = x} (-1)^{a \cdot S_t} \hat{f}(a),$$

for all $x \in GF(2)^L$. We compute the Walsh transform $\hat{\mathcal{W}}$ of $\mathcal{W}$ as follows:

$$\begin{aligned}
\hat{\mathcal{W}}(\mathbf{r}) &= \sum_{x \in GF(2)^L} (-1)^{\mathbf{r} \cdot x} \mathcal{W}(x) \\
&= \frac{1}{2^\ell} \sum_{t=1}^{k} \sum_{a \in GF(2)^\ell} (-1)^{a \cdot (\mathbf{r}G_t \oplus S_t)} \hat{f}(a) \\
&= \sum_{t=1}^{k} f(\mathbf{r}G_t \oplus S_t) \\
&= N(\mathbf{r}).
\end{aligned}$$

Algorithm 2 directly follows above computation. The total running time of our algorithm is $O(k\ell L 2^\ell + L2^L)$ with memory $O(2^L)$. To speed up the computation of $\mathcal{W}$, we could precompute the inner products of all pairs of ℓ-bit vectors in time $O(2^{2\ell})$ with memory $O(2^{2\ell})$. Thus, the total running time of the algorithm is $O(2^{2\ell} + kL2^\ell + L2^L)$ with memory $O(2^{2\ell} + 2^L)$.

Algorithm 2 The generalized MLD algorithm

Parameter:
  f, ℓ
Input:
  $G = (G_1, \ldots, G_k)$: the generator matrix
  vector stream $S_1, S_2, \cdots, S_k$
Processing:
  apply FWT to compute the table of $\hat{f}$
  initialize the table of $\mathcal{W}$ to 0
  for all ℓ-bit a **do**
    for $t = 1, \ldots, k$ **do**
      increment $\mathcal{W}(aG_t^\top)$ by $\frac{1}{2^\ell}(-1)^{a \cdot S_t} \hat{f}(a)$
    end for
  end for
  apply FWT to find $\mathbf{r}$ that achieves the minimal $\hat{\mathcal{W}}(\mathbf{r})$
  output $\mathbf{r}$

In the special case that is applicable to E0 (as is done in the next section): $G_{t+1} = AG_t$ for $t = 1, \ldots, k$, we precompute another table to map any L-bit vector x to $xA^\top$. It takes time 2^L with memory 2^L. The total time of the algorithm is thus $O\left(2^{2\ell} + (L+k)\,2^\ell + L2^L\right)$, with memory $O(2^{2\ell} + 2^L)$.

7 The Key-recovery Attack against E0

We approach similarly as in [10] to transform our distinguisher of Subsection 4.3 into a key-recovery attack. Our main contribution, however, is to decrease the time complexity by applying the preceding algorithm.

Let $Q(x) = \sum_{i=1}^{w} x^{q_i}$ be the multiple polynomial of $p_2(x)p_3(x)p_4(x)$ with degree d and weight w. Using techniques in Subsection 4.2 to find $Q(x)$ with (precomputation) complexity PC, we list the corresponding triplets (w, d, PC) for small w in Table 9.

Table 9. Complexity of finding multiple of $p_2(x)p_3(x)p_4(x)$ with degree d, weight w

	birthday problem with min. d				birthday problem tradeoff
weight w	5	4	3	2	5
degree d	2^{27}	2^{36}	2^{52}	2^{100}	$2^{34.3}$
Precomputation PC	2^{54}	2^{54}	2^{52}	-	$2^{36.3}$

Let $\tilde{\mathbf{x}}^{\mathbf{1}}$ be a guess for $\mathbf{x}^{\mathbf{1}}$, the initial state of R_1 which generates the keystream $\{z_t\}$ together with the other three fixed LFSRs. Denote $\tilde{x}_t^1$ the output bit of R_1 with the initial state $\tilde{\mathbf{x}}^{\mathbf{1}}$ at time t. We define

$$b_t(\tilde{\mathbf{x}}^{\mathbf{1}}) = \oplus_{i=1}^{w}(z_{t+q_i} \oplus z_{t+q_i+5}) \oplus \oplus_{i=1}^{w}(\tilde{x}^1_{t+q_i} \oplus \tilde{x}^1_{t+q_i+5}). \tag{9}$$

It can be shown that the second addend in Eq.(9) is also an m-sequence generated by the same LFSR. For brevity, we set

$$r_t = \oplus_{i=1}^{w}(\tilde{x}^1_{t+q_i} \oplus \tilde{x}^1_{t+q_i+5}),$$
$$s_t = \oplus_{i=1}^{w}(z_{t+q_i} \oplus z_{t+q_i+5}),$$

for $t = 1, \ldots, n$ (it corresponds to the data complexity $n + d$). We rewrite Eq.(9) as

$$b_t(\tilde{\mathbf{x}}^{\mathbf{1}}) = s_t \oplus r_t.$$

Given n-bit sequence of $b_t(\tilde{\mathbf{x}}^{\mathbf{1}})$'s, we count the occurrences[5] $N(\tilde{\mathbf{x}}^{\mathbf{1}})$ of ones, i.e. $N(\tilde{\mathbf{x}}^{\mathbf{1}}) = \sum_{t=0}^{n-1} b_t(\tilde{\mathbf{x}}^{\mathbf{1}})$. Using the analysis of [28], we estimate $N(\mathbf{x}^{\mathbf{1}})$ is the smallest of all $N(\tilde{\mathbf{x}}^{\mathbf{1}})$ with

$$n \approx \frac{4L_1 \log 2}{\gamma^{2w}}. \tag{10}$$

Note that this estimated figure is actually comparable to the conventional estimation [6, 16] on critical data complexity n_0 in correlation attacks, where

$$n_0 = \frac{L_1}{1 - h(\frac{1}{2} + \frac{1}{2}\gamma^w)} \approx \frac{2L_1 \log 2}{\gamma^{2w}}, \tag{11}$$

and h is the binary entropy function[6]. According to [6] simulations showed the probability of success is close to 1 (resp. $\frac{1}{2}$) for $n = 2n_0$ (resp. $n = n_0$) which is consistent with our analysis. Table 10 shows our estimated minimal n for $N(\mathbf{x}^{\mathbf{1}})$ to achieve a top rank corresponding to w. Now define $G_t = (a_0, \ldots, a_{L_1-1})^\top$, where $a_0 + a_1 x + \cdots + a_{L_1-1}x^{L_1-1} = x^t \mod p_1(x)$. Clearly our current problem to recover

[5] w is fixed in the attack, so we omit it in the notation $N(\tilde{\mathbf{x}}^{\mathbf{1}})$.

[6] $h(p) = -p \log_2 p - (1-p)\log_2(1-p)$ for $0 < p < 1$.

R_1 right fits into the MLD problem in Subsection 6.2. So we use the preceding MLD algorithm to recover $\mathbf{r}$ first, then apply linear transform to solve $\mathbf{x^1}$. Finally we conduct the same analysis as in Section 5 to decrease data complexity down to $O(\frac{r}{2r-1} \times \frac{4L_1 \log 2}{\gamma^{2w}})$; and we apply the technique introduced in Subsection 6.3 to obtain the reduced time complexity $O(n + \theta_1 \cdot 2^r + L_1 \cdot 2^{L_1})$. So, choosing $r = 12$, we can halve the time and data complexities. The attack complexities to recover R_1 for E0 are listed in Table 11. Once we recover R_1, we target R_2 next based on multiple of $p_3(x)p_4(x)$. Last, we use the technique of guess and determine in [11] to solve R_3 and R_4 with knowledge of the shortest two LFSRs. The detailed complexities of each step are shown in Table 12. A comparison of our attacks with the similar attack[7] [10] and the best two algebraic attacks [1, 8] is shown in Table 13.

Table 10. The estimated minimal n for $N(\mathbf{x^1})$ to top the rank from Eq.(10)

weight w	5	4	3	2	1
n	2^{40}	2^{33}	2^{27}	2^{20}	2^{14}

Table 11. Summary of primary partial key-recovery attacks against R_1 for E0

	w	d	n	data	precomputation	time	memory
Attack A	5	$2^{34.3}$	2^{39}	2^{39}	$2^{36.3}$	2^{39}	2^{25}
Attack B	4	2^{36}	2^{33}	2^{36}	2^{54}	2^{36}	2^{25}

Table 12. Detailed complexities of our key-recovery attack against E0

	w	d	n	data	precomputation	time	memory
R_1	5	$2^{34.3}$	2^{39}	2^{39}	$2^{36.3}$	2^{39}	2^{25}
R_2	3	2^{36}	2^{27}	2^{36}	2^{37}	2^{36}	2^{27}
R_3 and R_4	-	-	-	76	-	2^{33}	-
total	-	-	-	2^{39}	2^{37}	2^{39}	2^{27}

8 Conclusions

This paper formulates a systematic computation method of correlations by a recursive expression, which makes it easier to calculate correlations of the FSM output sequences up to 26 bits for E0 (and allows us to prove for the first time that the two known biases are the only largest). Then we successfully apply the concept of convolution to the analysis of the distinguisher based on all correlations, which allows us to build an efficient distinguisher that halves the data complexity of the basic uni-bias-based distinguisher due to the linear dependency of the two largest biases. Finally, by means of FWT, we propose a novel MLD algorithm to recover the

[7] The estimate of data complexity in [10] uses a different heuristic formula than ours. However we believe that their estimate and ours in Attack B are essentially the same.

Table 13. Complexities comparison of our attacks with the similar attack and algebraic attacks

		Precomputation	Time	Data	Memory
Algebraic Attacks	[1]	-	$2^{67.58}$	$2^{23.07}$	$2^{46.14}$
	[8]	2^{28}	2^{49}	$2^{23.4}$	2^{37}
Attack [10]		2^{54}	2^{63}	2^{34}	2^{34}
Our Attacks	A	2^{37}	2^{39}	2^{39}	2^{27}
	B	2^{54}	2^{37}	2^{36}	2^{27}

closest codeword for any linear code. Our proposed algorithm can be easily adapted to speed up a class of fast correlation attacks. Furthermore the algorithm is optimal when the length n of the code and the dimension L satisfy the relation $n \geq 2^L$, which is the case when we apply it to recover R_1 for E0. This results in the best known key-recovery attack against E0. Considering a maximal keystream length of 2745 bits for practical E0 in Bluetooth, our results still remain the academic interest. Meanwhile, our attack successfully illustrates the attack methodology of Baignères et al.[8]

Acknowledgment

Words fail the first author when she would like to express her heartfelt thanks to her forever faithful bosom friend Aoife Hegarty, who, in the face of a rough time in her own life, has been generously offering continuous one-way help through all those years ...

References

1. F. Armknecht, M. Krause, *Algebraic Attacks on Combiners with Memory*, Advances on Cryptography - CRYPTO 2003, Lecture Notes in Computer Science, vol.2729, D. Boneh ed., Springer-Verlag, pp. 162-175, 2003
2. T. Baignères, *A Generalization of Linear Cryptanalysis*, Diploma Thesis, EPFL, 2003
3. Bluetooth$^{\mathrm{TM}}$, *Bluetooth Specification*, version 1.2, pp. 903-948, November, 2003, available at `http://www.bluetooth.org`
4. A. Canteaut, F. Chabaud, *A New Algorithm for Finding Minimum-weight Words in a Linear Code: Application to Primitive Narrow-sense BCH Codes of Length 511*, INRIA, technical report, No. 2685, 1995
5. A. Canteaut, M. Trabbia, *Improved Fast Correlation Attacks Using Parity-check Equations of Weight 4 and 5*, Advances in Cryptology - EUROCRYPT 2000, Lecture Notes in Computer Science, vol.1807, B. Preneel ed., Springer-Verlag, pp. 573-588, 2000
6. V. Chepyzhov, T. Johansson, B. Smeets, *A Simple Algorithm for Fast Correlation Attacks on Stream Ciphers*, Fast Software Encryption 2000, Lecture Notes in Computer Science, vol.1978, B. Schneier ed., Springer-Verlag, pp. 181-195, 2000
7. P. Chose, A. Joux, M. Mitton, *Fast Correlation Attacks: An Algorithmic Point of View*, Advances in Cryptology - EUROCRYPT 2002, Lecture Notes in Computer Science, vol.2332, L. R. Knudsen ed., Springer-Verlag, pp. 209-221, 2002
8. N. T. Courtois, *Fast Algebraic Attacks on Stream Ciphers with Linear Feedback*, Advances on Cryptography - CRYPTO 2003, Lecture Notes in Computer Science, vol.2729, D. Boneh ed., Springer-Verlag, pp. 176-194, 2003
9. P. Ekdahl, T. Johansson, *Some Results on Correlations in the Bluetooth Stream Cipher*, Proceedings of the 10th Joint Conference on Communications and Coding, Austria, 2000

[8] Personal communication. Prior work available in [2].

10. P. Ekdahl, *On LFSR Based Stream Ciphers (Analysis and Design)*, Ph.D. Thesis, Lund Univ., Nov. 2003
11. S. Fluhrer, S. Lucks, *Analysis of the E0 Encryption System*, Selected Areas in Cryptography- SAC 2001, Lecture Notes in Computer Science, vol. 2259, S. Vaudenay and A. Youssef eds., Springer-Verlag, pp. 38-38, 2001
12. J. D. Golić, *Correlation Properties of a General Binary Combiner with Memory*, Journal of Cryptology, vol. 9, pp. 111-126, Nov. 1996
13. J. D. Golić, V. Bagini, G. Morgari, *Linear Cryptanalysis of Bluetooth Stream Cipher*, Advances in Cryptology - EUROCRYPT 2002, Lecture Notes in Computer Science, vol. 2332, L. R. Knudsen ed., Springer-Verlag, pp. 238-255, 2002
14. M. Hermelin, K. Nyberg, *Correlation Properties of the Bluetooth Combiner*, Information Security and Cryptology- ICISC'99, Lecture Notes in Computer Science, vol. 1787, JooSeok. Song ed., Springer-Verlag, pp. 17-29, 2000
15. M. Jakobsson, S. Wetzel, *Security Weakness in Bluetooth*, Topics in Cryptology - CT-RSA 2001, Lecture Notes in Computer Science, vol. 2020, D. Naccache ed., Springer-Verlag, pp. 176-191, 2001
16. T. Johansson, F. Jonsson, *Improved Fast Correlation Attacks on Stream Ciphers via Convolutional Codes*, Advances on Cryptography - CRYPTO'99, Lecture Notes in Computer Science, vol.1666, M. Wiener ed., Springer-Verlag, pp. 181-197, 1999
17. M. Krause, *BDD-Based Cryptanalysis of Keystream Generators*, Advances in Cryptology - EUROCRYPT 2002, Lecture Notes in Computer Science, vol. 2332, L. R. Knudsen ed., Springer-Verlag, pp. 222-237, 2002
18. R. Lidl, H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge, 1986
19. F. J. MacWilliams, N. J. A. Sloane, *The Theory of Error-correcting Codes*, North-Holland, 1996
20. M. Matsui, *Linear Cryptanalysis Method for DES Cipher*, Advances in Cryptology - EUROCRYPT'93, Lecture Notes in Computer Science, vol.765, Springer-Verlag, pp. 386-397, 1993
21. W. Meier, O. Staffelbach, *Fast Correlation Attacks on Certain Stream Ciphers*, Journal of Cryptology, vol. 1, pp. 159-176, Nov. 1989
22. W. Meier, O. Staffelbach, *Correlation Properties of Combiners with Memory in Stream Ciphers*, Journal of Cryptology, vol. 5, pp. 67-86, Nov. 1992
23. A. J. Menezes, P. C. van. Oorschot, S. A. Vanstone, *Handbook of Applied Cryptography*, CRC, 1996
24. W. Penzhorn, *Correlation Attacks on Stream Ciphers: Computing Low Weight Parity Checks based on Error Correcting Codes*, Fast Software Encryption 96, Lecture Notes in Computer Science, vol.1039, D. Gollmann ed., Springer-Verlag, pp. 159-172, 1996
25. R. A. Rueppel, *Correlation Immunity and the Summation Generator*, Advances in Cryptology - CRYPTO'85, Lecture Notes in Computer Science, Springer-Verlag, pp. 260-272, 1986
26. M. Saarinen, *Re: Bluetooth and E0*, Posted at `sci.crypt.research`, 02/09/00
27. T. Siegenthaler, *Correlation-Immunity of Nonlinear Combining Functions for Cryptographic Applications*, IEEE Transactions on Information Theory, vol. 30, pp. 776-780, 1984
28. S. Vaudenay, *An Experiment on DES - Statistical Cryptanalysis*, Proceedings of the 3rd ACM Conferences on Computer Security, pp. 139-147, 1996
29. D. Wagner, *A Generalized Birthday Problem*, Advances in Cryptology - CRYPTO 2002, Lecture Notes in Computer Science, vol.2442, Springer-Verlag, pp. 288-304, 2002
30. R. K. Yarlagadda, J. E. Hershey, *Hadamard Matrix Analysis and Synthesis with Applications to Communications and Signal/Image Processing*, Kluwer Academic, pp. 17-22, 1997

Appendix

A. Proof of Lemma 5

Let $Z \in GF(2)^k$ be a random variable independent of X with uniform distribution. We have

$$\begin{aligned}
&\sum_{w} \Delta(f(X,Z) \oplus w \cdot Z) \cdot \Delta(w \cdot g(Y) \oplus v \cdot Y) \\
&= \sum_{w} \mathrm{E}[(-1)^{f(X,Z)\oplus w\cdot Z}] \cdot \mathrm{E}[(-1)^{w\cdot g(Y)\oplus v\cdot Y}] \\
&= \sum_{w,x,y,z} \Pr(x,z) \cdot \Pr(y) \cdot (-1)^{f(x,z)\oplus v\cdot y \oplus w\cdot(z\oplus g(y))} \\
&= 2^k \cdot \sum_{x,y} \Pr(X=x, Z=g(y)) \cdot \Pr(Y=y) \cdot (-1)^{f(x,g(y))\oplus v\cdot y} \\
&= \sum_{x,y} \Pr(x,y) \cdot (-1)^{f(x,g(y))\oplus v\cdot y} \\
&= \mathrm{E}[(-1)^{f(X,g(Y))\oplus v\cdot Y}],
\end{aligned}$$

which is $\Delta(f(X,g(Y)) \oplus v \cdot Y)$.

B. Examples of Multiple Polynomials $Q(x)$

Example of $Q(x)$ with Low Degree. Here is a multiple polynomial of degree less than 855 with weight 31:

$$\begin{aligned}
Q_1(x) = {} & x^{668} + x^{579} + x^{553} + x^{313} + x^{262} + x^{121} + x^{117} + x^{109} + x^{106} + x^{101} + \\
& x^{100} + x^{97} + x^{94} + x^{87} + x^{82} + x^{76} + x^{72} + x^{71} + x^{57} + x^{47} + \\
& x^{40} + x^{37} + x^{34} + x^{32} + x^{23} + x^{21} + x^{17} + x^{16} + x^{3} + x^{2} + 1.
\end{aligned}$$

Observe that $Q_1(x)$ is not optimal as $w_{855} = 20$ from Table 3.

Examples of $Q(x)$ with Weight Four. Recall that $\theta_i = 2^{L_i} - 1$ is the order of $p_i(x)$ for $i = 1, 2, 3, 4$. By definition, $p_i(x) | x^{\theta_i} + 1$. On the other hand, $p_i(x)p_j(x) | \mathrm{lcm}(x^{\theta_i} + 1, x^{\theta_j} + 1) = x^{\mathrm{lcm}(\theta_i,\theta_j)} + 1$ for $i \neq j$, hence we deduce the following three multiple polynomials of $p(x)$ with weight 4 with ease:

$$\begin{aligned}
Q_2(x) &= (x^{\mathrm{lcm}(\theta_1,\theta_2)} + 1)(x^{\mathrm{lcm}(\theta_3,\theta_4)} + 1), \\
Q_3(x) &= (x^{\mathrm{lcm}(\theta_1,\theta_3)} + 1)(x^{\mathrm{lcm}(\theta_2,\theta_4)} + 1), \\
Q_4(x) &= (x^{\mathrm{lcm}(\theta_1,\theta_4)} + 1)(x^{\mathrm{lcm}(\theta_2,\theta_3)} + 1),
\end{aligned}$$

where

$$\begin{aligned}
&\mathrm{lcm}(\theta_1,\theta_2) = 2^{56} - 2^{31} - 2^{25} + 1, \mathrm{lcm}(\theta_1,\theta_3) = 2^{58} - 2^{33} - 2^{25} + 1, \\
&\mathrm{lcm}(\theta_1,\theta_4) = 2^{64} - 2^{39} - 2^{25} + 1, \mathrm{lcm}(\theta_2,\theta_3) = 2^{64} - 2^{33} - 2^{31} + 1, \\
&\mathrm{lcm}(\theta_2,\theta_4) = 2^{70} - 2^{39} - 2^{31} + 1, \mathrm{lcm}(\theta_3,\theta_4) = (2^{39} - 1)\textstyle\sum_{k=0}^{10} 2^{3k}.
\end{aligned}$$

The degrees of $Q_2(x), Q_3(x), Q_4(x)$ are approximately $2^{69}, 2^{70}, 2^{65}$ respectively. Note that we may also expect optimal multiples with degree in the same order of magnitude and weight 3.

C. Table of $\log_2 |\hat{\mathcal{D}}(a)|$ for $\ell = 6$

Table 14. $\log_2 |\hat{\mathcal{D}}(c^0_t, c^0_{t+1}, \ldots, c^0_{t+5})|$ where dashed entries denote $-\infty$

		$c^0_{t+3}c^0_{t+4}c^0_{t+5}$							
		000	001	010	011	100	101	110	111
$c^0_t c^0_{t+1} c^0_{t+2}$	000	0	-	-	-	-	-	-	-
	001	-	-	-	-4	-	-	-	-
	010	-	-	-	-	-	-	-4	-
	011	-	-	-	-	-	-6	-	**-3.356**
	100	-	**-3.356**	-	-	-	-	-	-8
	101	-	-	-	-	-4	-	-	-
	110	-	-8	-	-5.356	-	-5.356	-	-5.356
	111	-	-	-6	-	-	-	**-3.356**	-