The Rainbow Attack on Stream Ciphers Based on Maiorana-McFarland Functions

Khoongming Khoo[1], Guang Gong[2], and Hian-Kiat Lee[1]

[1] DSO National Laboratories
20 Science Park Drive, Singapore 118230
[2] Department of Electrical and Computer Engineering
University of Waterloo, Waterloo, Ont. N2L 3G1, Canada
{kkhoongm, lhiankia}@dso.org.sg, ggong@calliope.uwaterloo.ca

Abstract. In this paper, we present the rainbow attack on stream ciphers filtered by Maiorana-McFarland functions. This can be considered as a generalization of the time-memory-data trade-off attack of Mihaljevic and Imai on Toyocrypt. First, we substitute the filter function in Toyocrypt (which has the same size as the LFSR) with a general Maiorana-McFarland function. This allows us to apply the attack to a wider class of stream ciphers. Moreover, our description replaces the time-memory-data trade-off attack with the rainbow attack of Oeshlin, which offers better performance and implementation advantages. Second, we highlight how the choice of different Maiorana-McFarland functions can affect the effectiveness of our attack. Third, we show that the attack can be modified to apply on filter functions which are smaller than the LFSR or on filter-combiner stream ciphers. This allows us to cryptanalyze other configurations commonly found in practice. Finally, filter functions with vector output are sometimes used in stream ciphers to improve the throughput. Therefore the case when the Maiorana-McFarland functions have vector output is investigated. We found that the extra speed comes at the price of additional weaknesses which make the attacks easier.

Keywords: Time-memory-data trade-off attack, Rainbow attack, Maiorana-McFarland functions.

1 Introduction

The construction of Boolean functions with good cryptographic properties has been a well studied area of research. Some of these properties include balance, high nonlinearity, high order of resiliency, high algebraic degree and high order of propagation criteria. These properties ensure the Boolean functions are resistant against various correlation attacks when used in stream ciphers [3, 17].

A well-known class of Boolean functions with good cryptographic properties are the Maiorana-McFarland class which ensures many of the above mentioned

J. Zhou, M. Yung, and F. Bao (Eds.): ACNS 2006, LNCS 3989, pp. 194–209, 2006.

properties. For example, when n is odd, we can construct t-resilient n-bit functions with nonlinearity achieving the quadratic bound $2^{n-1} - 2^{(n-1)/2}$. By concatenating such a function with its complement, we construct $(t+1)$-resilient m-bit functions with nonlinearity satisfying the quadratic bound $2^{m-1} - 2^{m/2}$ for even $m = n + 1$. These nonlinearities are called the quadratic bounds because they are the maximum nonlinearities attainable for quadratic Boolean functions. When n is even, the Maiorana-McFarland also allows us to construct a large family of bent functions, i.e. Boolean functions with the highest nonlinearity $2^{n-1} - 2^{n/2-1}$. Finally, the saturated functions which achieve optimal order of resiliency $t = n - 1 - d$ and optimal nonlinearity $2^{n-1} - 2^{t+1}$ when the algebraic degree is d can be constructed by this method.

The Maiorana-McFarland class can be viewed as constructions based on concatenating linear functions. This is both an advantage and a weakness. It is an advantage because we can easily manipulate the distance between Maiorana-McFarland functions and linear functions to obtain resiliency and high nonlinearity. This helps to protect against correlation and fast correlation attacks [3, 17]. However, it also means the function becomes linear when we fix certain input bits. Mihaljevic and Imai were able to exploit this property to launch a search space reduction attack on Toyocrypt. Toyocrypt is a 128-bit stream cipher where a 128-bit modular linear feedback shift register (MLFSR) is filtered by a 128-bit Maiorana McFarland function. They were able to reduce the key space from 2^{128} to 2^{96} when 32 consecutive output bits are known. The attack works because for each guess on 96 bits of the 128-bit MLFSR, they were able to form 32 linear equations based on 32 consecutive output bits. This linear system can be solved to determine the remaining 32 bits in the MLFSR. Using the time-memory-data trade-off attack of Biryukov and Shamir, they further reduced the attack complexity to 2^{32} with 2^{80} pre-computation and 2^{64} memory based on 2^{48} keystream bits.

In Section 3, we generalize the time-memory-data trade-off attack on Toyocrypt by Mihaljevic and Imai [9] as follows:

1. We show that the search space reduction attack on Toyocrypt can be applied to a general Maiorana-McFarland function. Because linear feedback shift registers (LFSR's) are more commonly used in stream ciphers, we replace the MLFSR in Toyocrypt by an LFSR.
2. In [9], Mihaljevic and Imai describes how we can improve the search space reduction attack on Toyocrypt by applying the time-memory-data trade-off attack of Biryukov and Shamir [2, 7]. In this paper, we describe how we can improve the search space reduction attack on Maiorana-McFarland functions by applying the rainbow attack of Oeschlin [11]. The rainbow attack is twice as fast as the time-memory-data trade-off attack and offers various implementation advantages. We also incorporate an improvement of the rainbow attack by Mukhopadyay and Sarkar [10] in our attack.
3. We simplify the description of the time-memory-data trade-off attack of [9] by introducing a search function $F^{(c)}(x)$.

Based on our study, we characterize the performance of the attack for different Maiorana-McFarland functions. For n-bit Maiorana McFarland functions formed by concatenating $2^{n/2}$ linear functions of size $n/2$-bit, the search space is reduced from 2^n to $2^{3n/4}$. When we apply the rainbow attack, the search space is further reduced to $2^{n/4-1}$ with $2^{3n/8}$ consecutive keystream bits, $2^{5n/8}$ pre-computation and $2^{n/2}$ memory. This case correspond to the filter function in Toyocrypt with $n = 128$, $k = 64$ [9], the bent functions and the resilient functions whose nonlinearity satisfies the quadratic bound [4, 16].

For Maiorana-McFarland functions formed by concatenating a few large linear functions, we get a very effective reduction of the search space of an n-bit filter function generator from 2^n to $2^{n/2}$. In this case, the equivalent key length is only half of what is claimed. As shown by Gong and Khoo [6], this case corresponds to the degree-resiliency-nonlinearity optimized saturated functions introduced by Sarkar and Maitra at Crypto 2000 [14]. Thus although this class of functions has the best trade-off among important cryptographic properties like nonlinearity, resiliency and algebraic degree, they are weak against the search space reduction attack. When we apply the rainbow attack, the search space is further reduced to $2^{n/4-1}$ with $2^{n/2}$ consecutive keystream bits, $2^{n/2}$ pre-computation and $2^{3n/8}$ memory.

In Section 5, we extend our attack to the case where the Maiorana-McFarland function is of smaller size than the LFSR. This is a very common construction when the filtering function is implemented as a look-up-table (LUT). The LFSR may be 128-bit long and it is not possible to fit a LUT of size 2^{128} into the memory of the cipher. Thus a smaller filter function has to be used. In that case, the complexity of the search space reduction and rainbow attack depends on the width of the LFSR bits which are tapped to certain input bits of the filter function. These input bits have the property that when they are known, the Maiorana-McFarland function becomes linear.

In Section 6, we extend the attack to the filter combiner model. At each clock cycle, the Boolean function will extract several bits from each of s linear feedback shift registers $LFSR_i$, $i = 0, 1, \ldots, s-1$, as input to produce a keystream bit. As analyzed by Sarkar [13], the filter combiner offers various advantages over the filter function and combinatorial generators. We show that in this case, the search space reduction and rainbow attack can also be applied effectively.

In Section 7, we extend the attack to the case where the filter function is a vectorial Maiorana-McFarland function. Vector output filter function generator has higher throughput for faster communication speed but it has an additional weakness. There is an exponential decrease in the complexity of search space reduction when compared to the single output case. This gives a very efficient attack even by a direct exhaustive search. This complexity can be further reduced by applying the rainbow attack.

In Sections 3 to 7, we simplified the attack scenarios to give a clearer explanation of the attack methods. In Section 8, we describe how these attacks can be easily adapted to apply to stream ciphers that use more general linear finite state machines, tap points and filter functions.

2 Preliminaries on Maiorana McFarland Functions

The *Hadamard Transform* of a Boolean function $f : GF(2)^n \to GF(2)$ is

$$\hat{f}(w) = \sum_{x \in GF(2)^n} (-1)^{w \cdot x + f(x)}.$$

The *nonlinearity* of a function $f : GF(2)^n \to GF(2)$ is defined as

$$N_f = 2^{n-1} - \frac{1}{2} \max_w |\hat{f}(w)|.$$

A high nonlinearity is desirable as it ensures linear approximation of f is ineffective. This offers protection against linear approximation based attacks [8, 17].

A Boolean function $f : GF(2)^n \to GF(2)$ is *t-th order correlation immune*, denoted $CI(t)$, if $\hat{f}(w) = 0$ for all $1 \le wt(w) \le t$ where $wt(w)$ is the number of ones in the binary representation of w. Correlation immunity ensure that f cannot be approximated by linear functions with too few terms, which offers protection against correlation attack [17]. Furthermore, if f is balanced and $CI(t)$, we say f is resilient of order t.

The Maiorana-McFarland functions is defined by the equation:

$$f(x_0, \ldots, x_{n-1}) = g(x_0, \ldots, x_{k-1}) + (x_k, \ldots, x_{n-1}) \cdot \phi(x_0, \ldots, x_{k-1}). \tag{1}$$

where $f : GF(2)^n \to GF(2)$, $g : GF(2)^k \to GF(2)$ and $\phi : GF(2)^k \to GF(2)^{n-k}$. ϕ is usually an injection or 2-to-1 map which would require $k \le n/2$ or $k \le n/2+1$ respectively.

The Maiorana-McFarland functions had been used extensively to construct Boolean functions with good cryptographic properties in the past decade (see [4] for a summary). Some notable examples are listed in the following two Propositions.

Proposition 1. *(extracted from [4, 16])*

1. *Let $f : GF(2)^n \to GF(2)$ be defined by equation (1). Let n be odd, $k = (n-1)/2$ and $\phi : GF(2)^{(n-1)/2} \to GF(2)^{(n+1)/2}$ be an injection such that*

$$wt(\phi(x_0, \ldots, x_{k-1})) \ge t+1 \text{ and } |\{z \in GF(2)^{(n+1)/2} | wt(z) \ge t+1\}| \ge 2^{(n-1)/2}.$$

   Then f is a t-resilient function with nonlinearity $2^{n-1} - 2^{(n-1)/2}$.
2. *Let $f(x_0, \ldots, x_{n-1})$, n odd, be a t-resilient function constructed as in part 1. Then*

$$g(x_0, \ldots, x_{n-1}, x_n) = f(x_0, \ldots, x_{n-1}) + x_n,$$

   is $t+1$-resilient and has nonlinearity $2^{m-1} - 2^{m/2}$ where $m = n+1$ is even.
3. *Let n be even, $k = n/2$ and $\phi(x_0, \ldots, x_{k-1})$ be a permutation in equation (1), then $f(x)$ is a bent function, i.e. it has the highest possible nonlinearity $2^{n-1} - 2^{n/2-1}$.*

The first construction is quite useful because a nonlinearity of $2^{n-1} - 2^{(n-1)/2}$ is considered high for functions with odd number of input bits. Furthermore, when $n \equiv 1 \pmod 4$, we can also obtain resiliency of order $(n-1)/4$ [4, page 555]. The second construction derives highly nonlinear resilient function with even number of input bits from the first construction. The third construction on bent functions are widely used in cryptography because of their high nonlinearity. Some examples include the ciphers CAST and Toyocrypt [1, 9]. The functions presented in Proposition 1 has the common property that $k \approx n/2$.

The saturated functions are functions which attain optimal trade-off between algebraic degree d, order of resiliency $t = n - d - 1$ and nonlinearity $2^{n-1} - 2^{t+1}$. Such functions were constructed by Sarkar and Maitra in [14]. It was shown by Gong and Khoo in [6] that the saturated functions correspond to n-bit Maiorana-McFarland functions as follows.

Proposition 2. *(Sarkar, Maitra [14, 6]) Fix $d \geq 2$ and let $n = 2^{d-1} + d - 2$. Define $f : GF(2)^n \to GF(2)$ by equation (1) where $k = d - 1$. Let $\phi : GF(2)^{d-1} \to GF(2)^{2^{d-1}-1}$ be an injection such that $wt(\phi(x_0, \ldots, x_{k-1})) \geq 2^{d-1} - 2$. Then $deg(f) = d$, f is t-resilient having nonlinearity $2^{n-1} - 2^{t+1}$ where $t = n - 1 - d$. In that case, the order of resiliency is optimal by Siegenthaler's inequality [17] and nonlinearity is optimal by Sarkar-Maitra inequality [14].*

The function in Proposition 2 has the property that $k \approx \log_2(n) << n$. Propositions 1 and 2 construct Boolean functions with optimal cryptographic properties by concatenating linear functions. But we shall show that their linear structures can be exploited to give efficient attacks on stream ciphers in Section 3.

3 The Rainbow Attack on Maiorana-McFarland Functions

In [9], Mihaljevic and Imai presented a time-memory-data trade-off attack on the stream cipher Toyocrypt. Toyocrypt is a filter function generator where we have an MLFSR of length 128 bit filtered by a 128-bit Boolean function of the form:

$$f(x_0, \ldots, x_{127}) = g(x_0, \ldots, x_{63}) + (x_{64}, \ldots, x_{127}) \cdot \pi(x_0, \ldots, x_{63}),$$

where g has 3 terms in its algebraic normal form (ANF) of degree $4, 17, 63$ and π permutes the bit positions of $(x_0, \ldots, x_{63})$. Mihaljevic and Imai showed that the effective key diversity of such a generator can be reduced from 128 bits to 96 bits when 32 consecutive output bits are known. Based on this observation, they modified the Biryukov-Shamir [2] time-memory-data tradeoff attack for improved cryptanalysis.

It is easy see that the filter function in Toyocrypt is a Maiorana-McFarland function with parameters $n = 128$, $k = 64$. Due to the wide usage of the Maiorana-McFarland construction, it will be useful to generalize the Mihaljevic-Imai attack to a general Maiorana-McFarland filter function generator. In this attack, we look at a stream cipher where an n-stage LFSR is filtered by a n-bit Maiorana-McFarland function defined by equation (1). We assume bit i of the LFSR is the i-th input of $f(x)$.

Suppose we know l consecutive output bits $y_0,\ldots,y_{l-1}$ and let $(x_i,\ldots,x_{i+n-1})$ be the LFSR state corresponding to y_i. Based on equation (1), we have:

$$y_i = g(x_i,\ldots,x_{i+k-1}) + (x_{i+k},\ldots,x_{i+n-1})\cdot\phi(x_i,\ldots,x_{i+k-1}),\ i=0,\ldots,l-1. \quad (2)$$

Suppose we guess $k+l$ consecutive input bits $x_0,\ldots,x_{k+l-1}$. Then $g(x_i,\ldots,x_{i+k-1})$ and $\phi(x_i,\ldots,x_{i+k-1})$ will be known for $i=0,\ldots,l-1$. In that case, the l equations in (2) will be linear and they will contain $n-(k+l)$ unknown variables $x_{k+l},\ldots,x_{n-1}$. The variables $x_n,\ldots,x_{n+l-2}$ in equation (2) can be linearly expressed in terms of $x_0,\ldots,x_{n-1}$ using the LFSR feedback relation.

Thus we have l linear equations in $n-(k+l)$ variables. For this linear system to be solvable, we need:

$$l \geq n-(k+l) \Longrightarrow l \geq \frac{n-k}{2}.$$

Therefore, suppose we know $l=\lceil (n-k)/2\rceil$ consecutive output bits $y_0,\ldots,y_{l-1}$ and we guess the $k+l=n-l=\lfloor (n+k)/2\rfloor$ consecutive input bits $x_0,\ldots,x_{k+l-1}$. Then we can solve for the remaining l input bits $x_{k+l},\ldots,x_{n-1}$ in equation (2) and counter check whether our guess is correct, by back-substitution and comparing with a sufficiently long keystream, e.g. of length $2n$ bits. Thus we have proven that:

Theorem 1. *Consider an n-bit LFSR filtered by equation (1) where bit i of the LFSR is the ith input of $f(x)$. The key space is reduced from 2^n to $2^{\lfloor (n+k)/2\rfloor}$ bits when $\lceil (n-k)/2\rceil$ consecutive output bits are known.*

Remark 1. For ease of notation, we assume that $(n+k)/2$, $(n-k)/2$ are integers from now on. The case when they are not integers can be handled by adding the appropriate ceiling $\lceil\ \rceil$ and floor $\lfloor\ \rfloor$ operations as in Theorem 1.

Next we improve the attack complexity of Theorem 1 by applying the rainbow attack [11]. The rainbow attack can be seen as an improvement of the time-memory-data trade-off attack [2, 7, 9] which is twice as fast and has various implementation advantages.

Let $f : GF(2)^n \to GF(2)$ be the filter function of an n-bit LFSR. Define $\tilde{f} : GF(2)^n \to GF(2)^n$ as:

$$\tilde{f}(\tilde{x}) = n\text{-bit output of filter function generator},$$

when the LFSR is initialized by $\tilde{x}\in GF(2)^n$.

Let $c\in GF(2)^{(n-k)/2}$ be a fixed string. Given $x\in GF(2)^{(n+k)/2}$, we can use the proof of Theorem 1 to find $s\in GF(2)^{(n-k)/2}$ such that:

$$\tilde{f}(x||s) \text{ restricted to first } (n-k)/2 \text{ bits } = c,$$
$$\text{i.e. } \tilde{f}(x||s) = (c||y).$$

where '$||$' is concatenation of bit strings. Based on this computation, we introduce a *search function* $F^{(c)} : GF(2)^{(n+k)/2} \to GF(2)^{(n+k)/2}$ for the description of our attack. We define $F^{(c)}(x)$ to be the right most $(n+k)/2$ bits of $\tilde{f}(x||s)$, i.e.,

$$F^{(c)}(x) = y$$

The function $F^{(c)}$ will be the search function used in our description of the rainbow attack on Maiorana McFarland functions. We note that this function can also simplify the description of the attack in [9].

Setup:

1. Randomly choose a binary string $c \in GF(2)^{(n-k)/2}$ and define the function $F^{(c)}(x)$ as described above.
2. Define a chain of $t-1$ distinct functions $F_1(x), F_2(x), \ldots, F_{t-1}(x)$ which are slight variations of the search function $F^{(c)}(x)$ as follows. Randomly seed an $(n+k)/2$-bit LFSR (with maximum period) and generate a sequence of $(n+k)/2$-bit vectors $X_1, X_2, \ldots, X_{t-1}$. Let the variant functions be defined as $F_j(x) = F^{(c)}(x) \oplus X_j$ (Please see Remark 4 for further explanation on this step).
3. Let p and t be integers defined by $pt^2 = 2^{(n+k)/2}$. Form a $p \times t$ by 2 array as follows:
   For $i = 1 \ldots p \times t$, randomly choose a start point $y_{i,0}$ and compute the chain of values $y_{i,1} = F_1(y_{i,0}), y_{i,2} = F_2(y_{i,1}), \ldots, y_{i,t} = F_{t-1}(y_{i,t-1})$. Store the start and end points $(SP_i, EP_i) = (y_{i,0}, y_{i,t})$.

Attack:

1. We look among the keystream to find a n-bit string whose first $(n-k)/2$ bits matches the pattern c. Let the last $(n+k)/2$ bits of this string be y.
2. For $j = 2 \ldots t$, search among the endpoints EP_i in our table to check if

$$EP_i = F_{t-1}(\ldots(F_{t-j+1}(y)\ldots).$$

   If there is a match, then $(x||s)$ is the secret initial state of the LFSR where

$$x = F_{t-j}(\ldots F_1(SP_i)\ldots),$$

   and s is the $(n-k)/2$-bit string computed such that the leftmost $(n-k)/2$ bits of $\tilde{f}(x||s)$ is c. The string s can be found by solving linear equations as in the proof of Theorem 1.

Based on [2] and [11], the parameters in the attack satisfy the following constraint:

To look up the rainbow table, we need to compute $F_{t-1}(y)$, $F_{t-1}(F_{t-2}(y))$, $F_{t-1}(F_{t-2}(F_{t-3}(y)))$, ... The time taken is $T = \sum_{i=1}^{t-1} i = t(t-1)/2$ function computations. Let the amount of data collected be D. By [2], our table only need to cover $1/D$ of the whole search space $N = 2^{(n+k)/2} = pt^2$ because we just need one string out of D possible strings in the available keystream. Since we are only storing the end points, the memory M needed is pt/D. Thus we derive the relation:

$$TM^2D^2 = t(t-1)/2 \times (pt/D)^2 \times D^2$$
$$\approx p^2t^4/2 = N^2/2 \Longrightarrow T = N^2/(2M^2D^2).$$

Let the memory be $M = 2^{mem}$ and the number of strings of the form $(c||x)$ (where $c \in GF(2)^{(n-k)/2}$ is fixed) in the collected data be $D = 2^d$. This means we need to sample $2^{(n-k)/2+d}$ consecutive keystream bits to collect this data. Thus the processing time is $N^2/(2M^2D^2) = 2^{n+k-2(d+mem)-1}$. The pre-processing time is $N/D = 2^{(n+k)/2-d}$. We state the result formally as:

Theorem 2. *Consider an n-bit LFSR filtered by equation (1) where bit i of the LFSR is the ith input of $f(x)$. The LFSR initial state can be found with complexity $2^{n+k-2(d+mem)-1}$ by using $2^{(n+k)/2-d}$ pre-processing and 2^{mem} memory when $2^{(n-k)/2+d}$ consecutive output bits are known.*

Remark 2. We note that in the time-memory-data trade-off attack of [9], the function chains in a table are derived from a constant function $F^{(c)}(x)$, thus they have a high chance of collision because the table (which have to cover the search space) is large. An alternative is suggested in [9] which uses multiple tables where the function for each table is a variant of $F^{(c)}(x)$. In that case, we let $N = 2^{(n+k)/2} = pt^2$ and construct t table of size $p \times t$. This set-up gives optimal performance by the "matrix stopping rule" [7]. Let the amount of data be D. Then the amount of memory needed is $M = pt/D$ because we are storing p end points in each of t tables and we only need to cover $1/D$ of the search space. The time to look up a table is t and the processing complexity which is the time to look up t table is $T = t^2$. We deduce that:

$$TM^2D^2 = t^2 \times (pt/D)^2 \times D^2$$
$$= p^2t^4 = N^2.$$

Thus the processing complexity is $T = N^2/(M^2D^2)$ which is twice as slow as in the rainbow attack.

Remark 3. The rainbow attack, besides being twice as fast as the time-memory-data trade-off attack, also has the following implementation advantages [11]. The number of table look up in the rainbow attack is reduced by a factor of t when compared to the time-memory-data trade-off (which uses t tables). Rainbow tables have no loops because each reduction function F_j is only used once. So we do not need to spend time to detect and reject loops when constructing the table. Merging chains in rainbow tables have identical endpoints. So it can be used to determine merging chains, just like distinguished points. In time-memory-data trade-off, distinguished points are used to detect merging chains and loops. However, the chains have variable lengths. In comparison, rainbow chains avoid merging chains and loops by using distinct reduction functions. Thus the rainbow chains have constant lengths. As explained in [11], this is more efficient and effective.

Remark 4. The method we use to generate the functions $F_j(x)$ from an LFSR is by Mukhopadyay and Sarkar [10]. The method suggested in [2, 9] (and originally by Hellman in [7]) is to generate $F_j(x)$ by permuting the output bits of $F^{(c)}(x)$. But it was shown by Fiat and Noar that there exist search functions which are polynomial time indistinguishable from a random function but for which the time-memory trade-off attack fails [5], when permutation of output bits are used. The advantage of the approach of [10] is that it is not possible to construct a Fiat-Noar type example for the LFSR-based rainbow method. Moreover, LFSR sequences are very efficient to compute.

Example 1. We apply Theorem 2 to Toyocrypt with the parameters $n = 128$, $k = 64$, $d = 16$ and $mem = 64$. The complexity of the time-memory-data trade-off attack is 2^{80} for pre-processing and 2^{31} for processing when we know 2^{48} consecutive output bits. This attack is twice as fast as the attack in [9].

Remark 5. To obtain 2^{16} 128-bit ciphertext blocks where the first 32 bits is a fixed pattern c in Example 1, we need to scan through 2^{48} keystream bits. This scanning complexity is not taken into account in the processing complexity 2^{31}, which only covers the search of the rainbow table. Part of the reason for not mixing the two complexities is that searching the rainbow table involves computing the function $F^{(c)}$ (by solving a linear system) which is more complex than scanning for a fixed pattern from the keystream. The same remark applies to Example 2 and 3 later in the paper.

4 On the Security of Different Maiorana-McFarland Functions against the Rainbow Attack

In general, the parameter k in the Maiorana-McFarland construction (equation 1) is in the range $1 \leq k \leq n/2$.

4.1 The Case When k Is Approximately $n/2$

Consider the extreme case $k \approx n/2$. There are many optimal functions belonging to this class as summarized in Proposition 1. In this case, $(n-k)/2 \approx n/4$ and the key diversity is reduced to $(n+k)/2 \approx 3n/4$ bits when $\approx n/4$ consecutive keystream bits are known. Suppose we collect $2^d = 2^{n/8}$ ciphertexts corresponding to a pre-computed $n/4$-bit pattern, i.e. $2^{3n/8}$ consecutive keystream bits. Then in a rainbow attack with 2^{mem} memory, the complexity is $2^{5n/8}$ for pre-processing and $2^{5n/4-2mem-1}$ for the actual attack by Theorem 2. If n is not too big, it is reasonable to use $2^{mem} = 2^{n/2}$ memory which means the attack complexity is $2^{n/4-1}$. If we can obtain more keystream bits, then the pre-computation and attack complexity can be reduced further.

4.2 The Case When k Is Much Smaller Than n

The other extreme is when $k << n$. This scenario may occur when we use a saturated function from Proposition 2. In this case, $(n+k)/2 \approx n/2$ and the key diversity is reduced to $\approx n/2$ bits when $\approx n/2$ consecutive output bits are known. Suppose we collect $2^d = 1$ (where $d = 0$) ciphertext corresponding to a pre-computed $n/2$-bit pattern, i.e. we need $2^{n/2}$ consecutive keystream bits. Then in a time-memory-data trade-off attack using 2^{mem} memory, the complexity is $2^{n/2}$ for pre-processing and $2^{n-2mem-1}$ for processing. Unlike the case $k \approx n/2$, we can use less memory here because the search space is smaller. If we use $2^{mem} = 2^{3n/8}$ memory, then the attack complexity is $2^{n/4-1}$.

From the above discussion, we see that as k decreases, the memory, pre-computation and attack complexity decreases but the number of consecutive keystream bits needed increases. Sometimes it is not possible to obtain so many keystream bits for time-memory-data trade-off attack on equation 2. It may be more feasible to use Theorem 1 directly and perform an exhaustive search with complexity $2^{(n+k)/2}$ based on $(n-k)/2$ consecutive output bits.

5 When the LFSR and Boolean Functions Have Different Sizes

As a generalization, we consider the above attack when an n-bit LFSR is filtered by a m-bit Maiorana McFarland function $f(x)$ where $m < n$. Let the function be of the form

$$f(x_0, \ldots, x_{m-1}) = g(x_0, \ldots, x_{r-1}) + (x_r, \ldots, x_{m-1}) \cdot \phi(x_0, \ldots, x_{r-1}). \tag{3}$$

where $f : GF(2)^m \rightarrow GF(2)$, $g : GF(2)^r \rightarrow GF(2)$ and $\phi : GF(2)^r \rightarrow GF(2)^{m-r}$.

Therefore the function $f(x)$ becomes linear when the first r input bits are fixed. Let these r input bits be tapped from the leftmost k bits of the LFSR, and the remaining $m-r$ input bits of $f(x)$ be tapped from the rightmost $n-k$ LFSR bits.

As before, assume l consecutive output bits of $f(x)$ are known and we guess $k+l$ leftmost LFSR bits. Then we can form l linear equations with $n-(k+l)$ unknown variables of the LFSR initial state. This system of equations can be solved when $l = \lceil (n-k)/2 \rceil$. So knowing $\lceil (n-k)/2 \rceil$ consecutive output bits will reduce the initial state space from n bits to $k+l = \lfloor (n+k)/2 \rfloor$ bits. It is easy to see that we can apply the rainbow attack as in Section 3 by using the same search function $F^{(c)}(x)$. The attack complexity for direct exhaustive search and rainbow attack is the same as before but now, the parameter k depends not just on $f(x)$ but also on the tap points from the LFSR. We summarize our discussion as a theorem:

Theorem 3. *Consider an n-bit LFSR which is filtered by a m-bit Maiorana-McFarland function defined by equation (3). Suppose the first r bits of $f(x)$ is tapped from the leftmost k bits of the LFSR, and the remaining $m-r$ input bits of $f(x)$ is tapped from the rightmost $n-k$ LFSR bits. Then the key space is reduced from 2^n to $2^{(n+k)/2}$ when $(n-k)/2$ consecutive output bits are known.*

Furthermore, the LFSR initial state can be found with $2^{n+k-2(d+mem)-1}$ *processing,* $2^{(n+k)/2-d}$ *pre-processing and* 2^{mem} *memory when* $2^{(n-k)/2+d}$ *consecutive output bits are known.*

6 Extending the Attack to Filter Combiner Model

In this section, we extend the search space reduction and rainbow attack on the filter combiner model. For ease of explanation, we consider the case of two linear feedback shift registers $LFSR_1$ and $LFSR_2$. The attacks on more LFSR's are similar. At each clock cycle, a Boolean function will take as input several state bits from each of $LFSR_1$ and $LFSR_2$ to output a keystream bit.

Let the length of $LFSR_1$ be n_1 and that of $LFSR_2$ be n_2. Let $f : GF(2)^m \rightarrow GF(2)$ be defined by:

$$f(x_0, \ldots, x_{m-1}) = (x_r, \ldots, x_{m-1}) \cdot \phi(x_0, \ldots, x_{r-1}) + g(x_0, \ldots, x_{r-1}). \tag{4}$$

Therefore when we fix the first r input bits, $f(x)$ becomes linear.

Let the first r input bits of $f(x)$, i.e. $(x_0, x_1, \ldots, x_{r-1})$ be tapped from among the leftmost k_1 and k_2 bits of $LFSR_1$ and $LFSR_2$. Let the rest of the $n - r$ input bits be tapped from the rightmost $n_1 - k_1$ and $n_2 - k_2$ bits of $LFSR_1$ and $LFSR_2$.

Suppose we know l consecutive output bits $y_0, y_1, \ldots, y_{l-1}$. Let us guess the leftmost $k_1 + l$ and $k_2 + l$ bits of $LFSR_1$ and $LFSR_2$. Then at time i, $\phi(x_i, \ldots, x_{i+r-1})$, $g(x_i, \ldots, x_{i+r-1})$ are known for all $i = 0, 1, \ldots, l - 1$. This means:

$$y_i = f(x_i, \ldots, x_{i+m-1}) = (x_{i+r}, \ldots, x_{i+m-1}) \cdot \phi(x_i, \ldots, x_{i+r-1}) + g(x_i, \ldots, x_{i+r-1}).$$

is a linear equation for $i = 0, 1, \ldots, l - 1$.

We have l equations in $n_1 - (k_1 + l) + n_2 - (k_2 + l)$ variables. For this linear system to be solvable, we need

$$n_1 - (k_1 + l) + n_2 - (k_2 + l) \le l$$
$$\Longrightarrow l \ge \lceil ((n_1 - k_1) + (n_2 - k_2))/3 \rceil.$$

We take $l = \lceil ((n_1 - k_1) + (n_2 - k_2))/3 \rceil$. Thus the search space is reduced from $2^{n_1+n_2}$ to:

$$2^{(k_1+l)+(k_2+l)} \approx 2^{(2(n_1+n_2)+(k_1+k_2))/3}.$$

The rainbow attack can may be applied for our scenario as follows. Let $l = \lceil ((n_1 - k_1) + (n_2 - k_2))/3 \rceil$, we define a function $\tilde{f} : GF(2)^{n_1} \times GF(2)^{n_2} \rightarrow GF(2)^{k_1+k_2+3l}$ to be:

$$\tilde{f}(\tilde{x_1}, \tilde{x_2}) = \text{the } (k_1 + k_2 + 3l)\text{-bit output keystream},$$

when $(LFSR_1, LFSR_2)$ are initialized by $(\tilde{x_1}, \tilde{x_2})$. For a fixed string $c \in GF(2)^l$ and $x_i \in GF(2)^{k_i+l}$, we can find $s_i \in GF(2)^{n_i-(k_i+l)}$ such that $\tilde{f}(x_1||s_1, x_2||s_2)$=

$(c||y)$ by the method described above. Based on this computation, we define a search function $F^{(c)} : GF(2)^{k_1+l} \times GF(2)^{k_2+l} \rightarrow GF(2)^{k_1+k_2+2l}$ to be the rightmost k_1+k_2+2l bits of $\tilde{f}(\tilde{x_1}, \tilde{x_2})$, i.e.

$$F^{(c)}(x_1, x_2) = y$$

By using this search function, we can perform a rainbow attack as in Section 3. The search space is $N = 2^{k_1+k_2+2l}$. Assuming we have $M = 2^{mem}$ memory and we have 2^{l+d} consecutive keystream bits from which we can sample $D = 2^d$ ciphertext whose first l bits correspond to c. Then the preprocessing complexity is $N/D = 2^{k_1+k_2+2l-d}$ and processing complexity is $N^2/(2M^2D^2) = 2^{2(k_1+k_2+2l-(d+mem))-1}$.

Example 2. Let us consider a filter combiner generator where $LFSR_1$ and $LFSR_2$ have lengths $n_1 = 64 = n_2$. Let $f(x)$ be defined by equation (4) where $m = 64$ and $r = 32$. Let the first r bits of $f(x)$ be tapped from the leftmost k_1, k_2 bits of $LFSR_1$ and $LFSR_2$ where $k_1 = 16 = k_2$.

The complexity of direct search without applying rainbow attack is

$$2^{k_1+k_2+2l} = 2^{16+16+2\times 32} = 2^{96}.$$

where $l = \lceil ((64-16) + (64-16))/3 \rceil = 32$. The complexity is less than the intended security of $2^{n_1+n_2} = 2^{128}$.

Assuming we have $2^{mem} = 2^{64}$ memory and $2^{l+d} = 2^{48}$ consecutive keystream bits where $d = 16$. The initial LFSR state can be recovered with 2^{80} preprocessing and 2^{31} processing. Thus the attack complexities are similar to Toyocrypt.

In a similar way, the search space reduction and rainbow attack of a filter combiner with s LFSR can be computed. We state this formally as:

Theorem 4. *Consider a filter combiner where equation (4) filters the content of $LFSR_1, LFSR_2, \ldots, LFSR_s$ of size $n_1, n_2, \ldots, n_s$ respectively. Let the first r bits of equation (4) be tapped from the leftmost k_i bits of $LFSR_i$. And let the remaining $m-r$ bits be tapped from the rightmost $n_i - k_i$ bits of $LFSR_i$, $i = 0, 1, \ldots, s-1$.*

Let $l = ((n_1 - k_1) + \ldots + (n_s - k_s))/(s+1)$. Then the key space of the filter combiner is reduced from $2^{n_1+\ldots+n_s}$ to $2^{k_1+\ldots+k_s+s\times l}$ when l consecutive output bits are known. Furthermore, the LFSR initial states can be found with $2^{2(k_1+\ldots+k_s+s\times l-(d+mem))-1}$ processing, $2^{k_1+\ldots+k_s+s\times l-d}$ pre-processing and 2^{mem} memory when 2^{l+d} consecutive keystream bits are known.

7 Extending the Attack to Vectorial Maiorana-McFarland Functions

In this section, we consider the case where an n-bit LFSR is filtered by a vectorial Maiorana-McFarland functions $F : GF(2)^n \rightarrow GF(2)^m$ defined by:

$$F(x_0, \ldots, x_{n-1}) = (f_0(x_0, \ldots, x_{n-1}), \ldots, f_{m-1}(x_0, \ldots, x_{n-1})) \quad (5)$$

where each function $f_j : GF(2)^n \rightarrow GF(2)$ is defined by:

$$f_j(x_0,\ldots,x_{n-1}) = (x_k,\ldots,x_{n-1}) \cdot \phi_j(x_0,\ldots,x_{k-1}) + g_j(x_0,\ldots,x_{k-1}).$$

for $j = 0, 1, \ldots, m-1$. This case may occur in practice because the encryption speed of a vector output generator is m times faster than a single bit filter function generator.

For good security, we want any linear combination of $f_j(x)$ to correspond to a t-resilient Maiorana-McFarland function with high nonlinearity. The usual method to construct $F(x)$ is to ensure that linear combinations of $f_j(x)$ correspond to concatenation of linear functions which are distinct and each linear function in the concatenation is an expression in $t+1$ or more variables. This can be achieved by using linear codes as shown in [12].

We assume bit i of the LFSR is the i-th input of $F(x)$. Suppose we know l consecutive output words, i.e. $l \times m$ consecutive output bits.

word 1: $y_{0,0}, y_{0,1}, \ldots, y_{0,m-1}$

word 2: $y_{1,0}, y_{1,1}, \ldots, y_{1,m-1}$

...

word l: $y_{l-1,0}, y_{l-1,1}, \ldots, y_{l-1,m-1}$

Let us guess the $k+l$ leftmost bits of the LFSR, i.e. $(x_0, x_1, \ldots, x_{k+l-1})$. Then $(x_i, x_{i+1}, \ldots, x_{i+k-1})$ is known at time $i = 0, 1, , \ldots, l-1$ and the following equations are linear.

$$\begin{aligned}
y_{i,0} &= g_0(x_i,\ldots,x_{i+k-1}) + (x_{i+k},\ldots,x_{i+n-1}) \cdot \phi_0(x_i,\ldots,x_{i+k-1}) \\
y_{i,1} &= g_1(x_i,\ldots,x_{i+k-1}) + (x_{i+k},\ldots,x_{i+n-1}) \cdot \phi_1(x_i,\ldots,x_{i+k-1}) \\
&\cdots \\
y_{i,m-1} &= g_{m-1}(x_i,\ldots,x_{i+k-1}) + (x_{i+k},\ldots,x_{i+n-1}) \cdot \phi_{m-1}(x_i,\ldots,x_{i+k-1})
\end{aligned}$$

We have $l \times m$ equations in $n-(k+l)$ unknowns. For this linear system to be solvable, we need:

$$\begin{aligned}
n-(k+l) &\leq l \times m \\
\Longrightarrow l &\geq \lceil (n-k)/(m+1) \rceil.
\end{aligned}$$

We take $l = \lceil (n-k)/(m+1) \rceil$. Thus the search space is reduced from 2^n to $2^{k+l} = 2^{k+\lceil (n-k)/(m+1) \rceil}$.

The rainbow attack can may be applied for our scenario as follows. Let $l = \lceil (n-k)/(m+1) \rceil$, we define a search function $\tilde{F} : GF(2)^n \rightarrow GF(2)^{k+(m+1)l}$ to be

$$\tilde{F}(\tilde{x}) = \text{the } (k+(m+1)l)\text{-bit output keystream},$$

when the LFSR is initialized by $\tilde{x} \in GF(2)^n$. For a fixed string $c \in GF(2)^{ml}$ and $x \in GF(2)^{k+l}$, we can find $s \in GF(2)^{n-(k+l)}$ such that $\tilde{F}(x||s) = (c||y)$ by the method described above. Based on this computation, we define a search function $F^{(c)} : GF(2)^{k+l} \rightarrow GF(2)^{k+l}$ to be the rightmost $k+l$ bits of $\tilde{F}(x||s)$, i.e.,

$$F^{(c)}(x) = y$$

By using this search function, we can perform a rainbow attack as in Section 3. The search space is $N = 2^{k+l}$. Assume we have $M = 2^{mem}$ memory and 2^{ml+d} consecutive keystream bits (from which we can sample $D = 2^d$ ciphertext whose first l m-bit words correspond to c). Then the preprocessing complexity is $N/D = 2^{k+l-d}$ and processing complexity is $N^2/(2M^2D^2) = 2^{2(k+l-(d+mem))-1}$.

Theorem 5. *Let $l = \lceil (n-k)/(m+1) \rceil$. Consider an n-bit LFSR filtered by equation (5) where bit i of the LFSR is the i-th input of $F(x)$. The key space is reduced from 2^n to 2^{k+l} when ml consecutive output bits are known.*

Furthermore, the LFSR initial states can be found with $2^{2(k+l-(d+mem))-1}$ processing, 2^{k+l-d} pre-processing and 2^{mem} memory when 2^{ml+d} consecutive keystream bits are known.

Remark 6. We may also consider the case where the vector Maiorana-McFarland function has different size as the LFSR as in Section 5.

Another extension is when the filter function in the filter combiner model is a vectorial Maiorana McFarland function. In that case, the result is a combination of Theorem 4 and 5.

Example 3. Consider the parameters in Toyocrypt where we have a 128-bit stream cipher filtered by a 128-bit vector Maiorana-McFarland function $F(x)$ with parameter $k = 64$ and m output bits. $F(x)$ may correspond to the vector function in Corollary 1 of [12] which is 1-resilient and has nonlinearity $2^{127} - 2^{64}$.

Then by Theorem 5, $l = \lceil 64/(m+1) \rceil$ and the search space reduction is 2^{64+l}. Suppose we apply rainbow attack with 2^d ciphertext whose first $m \times l$ bits correspond to a fixed string c. Then we need a keystream of length 2^{ml+d}. If we have 2^{mem} memory, the pre-processing complexity is 2^{64+l-d} and the processing complexity is $2^{2(64+l-(d+mem))-1}$. These values are tabulated for different output size m in Table 1, 2.

In Table 1, we list the size of the reduced search space for different output size.

In Table 2, we list the pre-processing and processing complexities of rainbow attack for different amount of memory and keystream. Here we fix the attack complexity as 2^{31}, which is considered sufficiently fast in practice.

In this example, we see that as the number of output bits increases, the search space, memory, pre-processing and processing complexities decrease while the amount of consecutive keystream bits needed increases.

Table 1. Reduced Search Space for $n = 128$, $k = 64$ and Different Output Size m

Output Size m	1	2	3	4	5	6	8	16
Reduced Search Space	2^{96}	2^{86}	2^{80}	2^{77}	2^{75}	2^{74}	2^{72}	2^{68}

Table 2. Complexities of Rainbow Attack for $n = 128$, $k = 64$ and Different Output Size m where Attack Complexity is fixed as 2^{31}, this gives lower Memory and Keystream Requirements

Output Size m	1	2	3	4	5	6	8	16
Consecutive Keystream	2^{48}	2^{52}	2^{56}	2^{58}	2^{61}	2^{68}	2^{72}	2^{68}
Memory	2^{64}	2^{62}	2^{56}	2^{55}	2^{53}	2^{52}	2^{51}	2^{48}
Pre-processing Complexity	2^{80}	2^{78}	2^{72}	2^{71}	2^{69}	2^{68}	2^{67}	2^{64}
Processing Complexity	2^{31}	2^{31}	2^{31}	2^{31}	2^{31}	2^{31}	2^{31}	2^{31}
Number of Ciphertext	2^{16}	2^{8}	2^{8}	2^{6}	2^{6}	2^{6}	2^{5}	2^{4}

8 Further Generalizations

Some ways in which our attacks can be further generalized are as follows:

1. In Theorem 2 and 5, we have adopted the convention that the first k input bits of $f(x)$ are always tapped from the leftmost k bits of the LFSR. It is easy to see that the attacks have the same complexities if we tap any k consecutive bits of the LFSR.
2. Similarly, in Theorem 3, we can tap the first r input bits of $f(x)$ from any consecutive k bits of the LFSR. In Theorem 4, we can tap the r bits from any consecutive $k_1, \ldots, k_s$ bits of $LFSR_1, \ldots, LFSR_s$ respectively.
3. In our attacks, we have presented the rainbow attack on Maiorana McFarland functions because it is a well-known and common construction in the Boolean function literature. In that case, the function becomes linear when the leftmost k bits are known. To make the attack more general, we can look at any n-bit Boolean function which becomes linear when k (not necessarily consecutive) input bits are known.
4. In our attacks, we can replace the LFSR by any linear finite state machine like a modular linear feedback shift register (MLFSR), Galois linear feedback shift register (GLFSR) or linear cellular automata. This is because the attacks only make use of the property that any LFSR state bits at time i is a linear function of the initial state.

References

1. C. Adams, "The CAST-128 Encryption Algorithm", RFC 2144.
2. A. Biryukov and A. Shamir, "Cryptanalytic Time/Memory/Data Trade-offs for Stream Ciphers", LNCS 1976, *Asiacrypt 2000*, pp. 1-13, Springer-Verlag, 2000.
3. A. Canteaut and M. Trabbia, "Improved Fast Correlation Attack using Parity Check Equations of Weight 4 and 5", LNCS 1807, *Eurocrypt 2000*, pp. 573-588, Springer-Verlag, 2000.
4. C. Carlet, "A Larger Class of Cryptographic Boolean Functions via a Study of the Moriana-McFarland Construction", LNCS 2442, *Crypto'2002*, pp. 549-564, Springer-Verlag, 2002.

5. A. Fiat and M. Naor, "Rigorous Time/Space Tradeoffs for Inverting Functions", *STOC 1991*, pp. 534-541, 1991.
6. G. Gong and K. Khoo, "Additive Autocorrelation of Resilient Boolean Functions", LNCS 3006, *Selected Areas of Cryptography 2003*, pp. 275-290, Springer-Verlag, 2003.
7. M. Hellman, "A Cryptanalytic Time-Memory Trade-Off", *IEEE Trans. on Information Theory*, vol. 26, pp. 401-406, 1980.
8. M. Matsui, "Linear cryptanalysis method for DES cipher", LNCS 765, *Eurocrypt'93*, pp. 386-397, 1994.
9. M.J. Mihaljevic and H. Imai, "Cryptanalysis of Toyocrypt-HS1 Stream Cipher", IEICE Trans. Fundamentals, vol. E85-A no. 1, pp. 66-73, 2002.
10. S. Mukhopadyay and P. Sarkar, "Application of LFSRs in Time/Memory Trade-Off Cryptanalysis", Indian Statistical Institute Technical Report No. ASD/04/9 (Revised Version), 2005.
11. P. Oeschlin, "Making a Faster Cryptanalytic Time-Memory Trade-Off", LNCS 2729, *Crypto 2003*, Springer-Verlag, 2003.
12. E. Pasalic and S. Maitra, "Linear Codes in Constructing Resilient Functions with High Nonlinearity", LNCS 2259, *Selected Areas in Cryptography 2001*, pp. 60-74, Springer-Verlag, 2001.
13. P. Sarkar, "The Filter-Combiner Model for Memoryless Synchronous Stream Ciphers", LNCS 2442, *Crypto 2002*, pp. 533-548, Springer-Verlag, 2002.
14. P. Sarkar and S. Maitra, "Nonlinearity Bounds and Constructions of Resilient Boolean Functions", LNCS 1880, *Crypto 2000*, pp. 515-532, Springer-Verlag, 2000.
15. P. Sarkar and S. Maitra, "Construction of Boolean Functions with Important Cryptographic Properties", LNCS 1807, *Eurocrypt 2000*, pp. 485-506, Springer-Verlag, 2000.
16. J. Seberry, X.M. Zhang and Y. Zheng, "On Constructions and Nonlinearity of Correlation Immune Functions", LNCS 765, *Eurocrypt'93*, pp. 181-199, 1994.
17. T. Siegenthaler, "Decrypting a Class of Stream Ciphers using Ciphertexts only", *IEEE Transactions on Computers*, vol. C34, no. 1, pp. 81-85, 1985.