Proceedings of International Cryptology Conference 2010 (cryptology2010)

BIAS IN THE NONLINEAR FILTER GENERATOR OUTPUT SEQUENCE

Sui-Guan Teo, Leonie Simpson and Ed Dawson
Information Security Institute,
Queensland University of Technology,
GPO Box 2434, Brisbane, QLD 4001, Australia
{sg.teo,lr.simpson,e.dawson}@qut.edu.au

Abstract :

Nonlinear filter generators are common components used in the keystream generators for stream ciphers and more recently for authentication mechanisms. They consist of a Linear Feedback Shift Register (LFSR) and a nonlinear Boolean function to mask the linearity of the LFSR output. Properties of the output of a nonlinear filter are not well studied. Anderson noted that the m-tuple output of a nonlinear filter with consecutive taps to the filter function is unevenly distributed. Current designs use taps which are not consecutive. We examine m-tuple outputs from nonlinear filter generators constructed using various LFSRs and Boolean functions for both consecutive and uneven (full positive difference sets where possible) tap positions. The investigation reveals that in both cases, the m-tuple output is not uniform. However, consecutive tap positions result in a more biased distribution than uneven tap positions, with some m-tuples not occurring at all. These biased distributions indicate a potential flaw that could be exploited for cryptanalysis.

1 Introduction

Linear Feedback Shift Registers (LFSRs) are commonly used to produce sequences for cryptographic purposes. For example, they may be used as components of the keystream generator in a stream cipher. The theory regarding the properties of LFSR sequences is well known. The research presented in this paper focuses on the sequences produced by binary LFSRs, where each stage of the shift register contains a single bit.

If the feedback polynomial of the LFSR is primitive, the binary sequence produced has several properties which are useful for cryptographic applications. Firstly, the sequence has a known period: provided the initial state is non-zero, a LFSR of length L with primitive feedback polynomial produces a binary sequence of length $2^L - 1$. Thus a large period can be guaranteed by choosing an appropriate value for L. Secondly; the sequence has some good statistical properties. The distribution of all m-tuple patterns, for $m \in \{1,2,\dots L\}$ is almost uniform. For example, when $m = 1$, one period of the LFSR output sequence contains 2^{L-1} ones, and $2^{L-1} - 1$ zeroes. When $m = 2$ if we consider one period of the LFSR output sequence as a series of overlapping two bit patterns, each of the two-bit patterns 01, 10 and 11 occurs 2^{L-2} times, and the pattern 00 occurs $2^{L-2} - 1$ times. Similarly, in one period of the LFSR output sequence, each m-bit pattern occurs 2^{L-m} times, except for the all-zero m-bit pattern which occurs $2^{L-m} - 1$ times. The distribution of m-tuple patterns in random sequences is expected to be uniform.

Although LFSR sequences have many desirable properties, using the LFSR output sequence directly as keystream is not advisable due to the linearity of LFSR sequences. To make use of the desirable properties of the LFSR in a keystream generator for a stream cipher, it is necessary to introduce nonlinearity. A simple method is to use the contents of several stages of the LFSR as inputs to a nonlinear Boolean function, and use the output of the function as the keystream. The nonlinear Boolean function is referred to as a filter function, and keystream generators based on a single LFSR and a nonlinear combining functions are known as nonlinear filter generators (NLFG). A diagram of a NLFG is shown in Figure 1. The NLFG aims to make use of the good properties of the underlying LFSR, so it is worthwhile examining the NLFG output sequence to determine which of the desirable properties of the LFSR sequence are maintained in the NLFG output sequence. The period of the NLFG keystream sequence is known to be $2^L - 1$ (the same as the underlying LFSR sequence) if the LFSR feedback function is primitive and of degree L and the nonlinear filter function is balanced (Simpson, 2000). For most cryptographic purposes, a balanced filter function is used as a balanced output sequence is required.

A balanced filter function applied to the stages of a LFSR with primitive feedback function and non-zero initial state results in an output keystream where the difference between the number of zeroes and

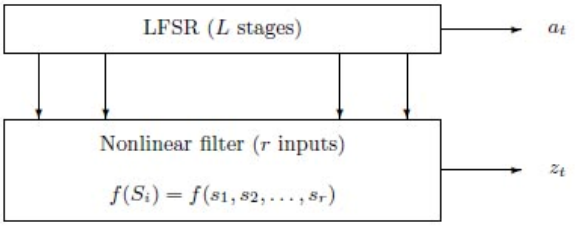

Figure 1: Nonlinear Filter Generator

the number of ones occurring in one period of the keystream sequences is exactly one, that is, close to uniform. Much less is known about the frequency distribution of m-bit patterns in the NLFG output sequence for $m > 1$. An early paper by Anderson (Anderson, 1994) discusses the distribution of m-bit patterns in the NLFG output sequence in the context of a correlation attack on the NLFG. Anderson considers that the common NLFG correlation attack strategy, which regards the keystream as a series of individual bits, discards information about the nonlinear structure of the filter function. Instead, for a given m-input Boolean filter function, he defines an *augmented function* which maps a $(2m - 1)$-bit input to an m-bit output. Essentially this is applying the m-input filter function m times in succession, assuming that the inputs to the filter function are from consecutive positions of the underlying sequence, although this is not stated explicitly in the paper. For the augmented function Anderson examined, the m-bit pattern distributions are clearly biased. For a particular Boolean function, certain m-tuples do not occur as outputs at all. This function was a bent function, which, due to its unbalanced nature, is not suitable as the filter function for a nonlinear filter generator. However, it was not clear whether distributions with non-occurring m-tuples are possible when balanced functions are used. Furthermore, the relationship between the characteristics of the Boolean filter function and the degree of bias in the output is not revealed.

To provide resistance to guess-and-determine style attacks, NLFG-style designs now commonly take the inputs to the filter from positions in the LFSR which are not consecutive, ideally tap positions which form a full positive difference set. The effect of this change in the positions of the input stages of the m-tuple pattern distribution of the NLFG output sequence warrants further investigation.

This paper presents the results of an investigation into the distribution of m-bit patterns in NLFG output sequences. This extends the earlier work of Anderson, where the value of m was used as both the number of inputs to the filter function and the length of the bit patterns examined in the NLFG output sequence. We make a clear distinction between these two parameters. We denote the number of inputs to the filter function by r, and consider the distribution of m-bit patterns in the NLFG output sequence for $m \in \{1,2,\ldots L\}$. We examine the output sequence of NLFGs constructed using various LFSRs and balanced nonlinear Boolean functions. In additional, we investigate both the case where the inputs to the filter function are from consecutive LFSR stages and the case where the inputs are non-consecutive from irregularly selected stages (full positive difference sets where possible).

The remainder of this paper is organised as follows. In Section 2, we describe our experimental design. In Section 3, the results of our experiments are described. In Section 4, we discuss the possible implications of our findings on the use of outputs of nonlinear filters for cryptographic purposes. Section 5 concludes this paper and proposes some future work.

2 Experimental Design

There are two main components of a NLFG, the LFSR and the nonlinear Boolean function. The goal of our experiments was to determine how these components affect the output sequence of the NLFG. We examine how choices in length and feedback polynomials for the LFSR and tap-settings to a nonlinear Boolean function affect the distribution of m-tuple outputs of the keystream sequence. In order to accurately determine the m-tuple distribution, it is necessary to produce an entire period of the keystream sequence. This constrained the length of the LFSRs used in our experiments. LFSR of length L, for L ranging from 13

to 20 bits, were chosen in our experiments. The primitive feedback polynomials chosen are given in Table 1.

LFSR feedback polynomial	LFSR feedback polynomial
L1: $x^{13} + x^4 + x^3 + x^1 + 1$	L2: $x^{13} + x^{12} + x^{10} + x^9 + x^6 + x^3 + 1$
L3: $x^{15} + x^{10} + x^5 + x^1 + 1$	L4: $x^{15} + x^1 + 1$
L5: $x^{16} + x^5 + x^3 + x^2 + 1$	L6: $x^{16} + x^{15} + x^{14} + x^8 + x^4 + x^3 + 1$
L7: $x^{18} + x^5 + x^2 + x^1 + 1$	L8: $x^{18} + x^{16} + x^{15} + x^{12} + x^{11} + x^9 + x^7 + x^6 + x^5 + x^4 + x^2 + x^1 + 1$
L9: $x^{20} + x^{19} + x^4 + x^3 + 1$	L10: $x^{20} + x^{19} + x^{18} + x^{15} + x^{14} + x^{12} + x^{11} + x^{10} + x^4 + x^2 + 1$

Table 1: LFSR feedback polynomial

Three balanced Boolean functions, F1, F2 and F3 were chosen for use as nonlinear filters. All three appear in the cryptographic literature. F1 is a 5-bit Boolean function used in the Grain stream cipher (Hell, Johansson, & Meier, 2005). F2 is a 6-bit Boolean function obtained from a report by Faugère & Ars (Faugère & Ars, 2003). F3 is a 7-bit Boolean function used in Pomraranch (Jansen, Helleseth, & Kolosha, 2005). The algebraic normal forms of these nonlinear Boolean functions are:

F1: $x2 + x5 + x1x4 + x3x4 + x4x5 + x1x2x3 + x1x3x4 + x1x3x5 + x2x3x5 + x3x4x5$
F2: $x1x2x3 + x2x3x6 + x1x2 + x3x4 + x5x6 + x4 + x5$
F3: ANF omitted due to size. The truth table of the Boolean function can be obtained from Jansen, Helleseth, & Kolosha (2005).

Relevant characteristics of these three Boolean functions, namely the algebraic degree, nonlinearity and correlation immunity are shown in Table 2.

Function	Algebraic degree	Nonlinearity	Correlation immunity
F1	3	12	1
F2	3	24	0
F3	4	56	2

Table 2: Characteristics of the Boolean functions

For each of the feedback polynomials, three different sets of tap settings for the Boolean functions were chosen. One set of tap settings used consecutive taps from the LFSR and two sets used uneven (or FPDS where possible) taps from the LFSR. In the uneven tap settings scenario, two sets of tap settings were used. These are denoted T1 and T2 in Table 3.

LFSR	5-bit Boolean function		6-bit Boolean function		7-bit Boolean function	
	T1	T2	T1	T2	T1	T2
L1	0,1,4,8,12	0,1,3,7,12	0,1,2,5,9,12	0,2,5,7,10,12	0,1,3,5,9,11,12	0,2,3,5,8,10,12
L2						
L3	0,1,4,9,14	0,4,6,13,14	0,1,4,8,10,14	0,2,3,10,13,14	0,1,4,5,10,13,14	0,2,3,7,9,11,14
L4						
L5	0,1,3,11,15	0,1,5,11,15	0,1,4,8,13,15	0,3,7,8,11,15	0,1,3,6,10,12,15	0,2,3,7,10,13,15
L6						
L7	0,3,8,13,17	0,2,5,9,17	0,2,3,8,15,17	0,1,4,10,12,17	0,1,3,9,11,14,17	0,2,8,9,12,15,17
L8						
L9	0,1,3,7,19	0,2,7,9,19	0,1,3,7,12,19	0,1,4,11,13,19	0,1,3,8,11,17,19	0,3,5,9,10,14,19
L10						

Table 3: Tap settings used in our experiments.

For each LFSR, nonlinear filter function and tap setting combination, the NLFG was run to generate a sequence $2^L + m - 1$ in length. The frequency distribution of m-tuples was calculated for m = 2 to 13. From this, the m-tuple which occurs least and most frequently for m-tuples of sizes 2 to 13 were noted. The standard deviation is a useful summary measure for the m-tuple distribution of a sequence. The smaller the standard deviation, the closer the m-tuple distribution of the sequence is to a uniform distribution. To enable comparisons where different size LFSR are used, the proportions of all m-tuples which have a specific value is calculated and the standard deviation of the proportion is also calculated. Recall the m-tuple distribution for the maximal length sequence produced by a LFSR is almost uniform. For example, the 3-tuple distribution for a 15-bit LFSR L3 with the nonlinear filter function F1 is given in Table 4. Clearly, from this table, the distribution is far from uniform. This is shown by the large standard deviation and chi-square value.

m-tuple	Expected no. of occurrences	Observed number of occurrences	Standard Deviation	Proportion of all 3-tuples	Standard Deviation of proportion of all 3-tuples	Goodness-of-fit test value
000	4095	2815	768.208	0.023438	0.023445	1152.098
001	4096	4352		0.132818		
010		4864		0.148442		
011		4352		0.132818		
100		4352		0.132818		
101		4864		0.148442		
110		4352		0.132818		
111		2816		0.085940		

Table 4: 3-tuple distribution of a NLFG sequence.

3 Experimental Results

Our experiments involved 90 NLFGs, comprising of different LFSRs, Boolean functions and tap settings. A sequence of length $2^L + m - 1$ bits for each NLFG was generated and the output sequence was examined for m-tuples for values of m ranging from 2 to 13 bits. We make a number of observations based on the results of our experiment. The factors which could impact on the m-tuple distribution include the positions of the inputs to the filter functions, the number of inputs to the filter function, and the length and feedback function of the LFSR.

Observation 3.1: The m-tuple distribution of NLFG output sequences is generally non-uniform.

Note that our observation supports the earlier findings by Anderson. We also note that the degree of non-uniformity varies depending on the combination of LFSR feedback function, the nonlinear filter functions and positions of input taps to the filter function. There are a few cases when the m-tuple output of the NLFG was uniform for smaller m-tuple values. For example, the 3-tuple distribution for a NLFG using the L5, F1 and T1 combination had the m-tuple distribution expected of a maximal length sequence. However, as m increased, the distribution became less uniform. Close to uniform distribution were more frequent when $m < 4$ and when uneven tap settings were used. With the exception of one case when $m = 5$, all m-tuple distributions when $m > 4$ were not uniform for NLFGs which used uneven tap settings. The almost uniform m-tuple distribution never occurred for consecutive tap settings for all m-tuples tested.

Observation 3.2: The m-tuple distribution is less uniform when tap settings are consecutive.

When comparing the m-tuple distribution for the output sequences obtained from NLFGs with the same LFSR and filter function but with different positions in the LFSR selected for inputs to the filter function, the distributions when the tap settings are consecutive are more varied than when the tap settings are uneven. For example, in the case for a 5-tuple distribution of a NLFG using the feedback function L3 with

consecutive tap settings and the F1 as the nonlinear filter, the least frequent 5-tuple occurred 2815 times and the most frequent 5-tuple occurred 4864 times. The standard deviation obtained was 320.057. When the same feedback function and filter function was used in a NLFG with uneven tap setting T1, the least frequent 5-tuple occurred 3520 times and the most frequent 5-tuple occurred 4672 times. The standard deviation obtained 133.982. For the same LFSR and nonlinear filter with the uneven tap setting T2, the minimum obtained was 4095 and the maximum obtained was 4096 and the standard deviation obtained 0.331. This trend was apparent for every NLFG sequence examined.

Observation 3.3: For some NLFGs with balanced Boolean functions, some m-tuples do not occur.

It is possible that particular m-tuples may not appear in a NLFG output sequence. In our experiments, we noted that this can occur when the nonlinear Boolean functions are balanced. The number of different m-tuples which do not appear in the output sequence is higher for NLFGs using consecutive tap settings than when uneven tap settings are used. For example, a NLFG using the feedback function L1, F2 Boolean function and the consecutive tap settings has 20 non-occurring 10-bit tuples. In contrast, a NLFG using the same feedback function and Boolean function with the T1 tap setting has only three non-occurring 10-bit tuples. As the m-tuple size increases, the number of non-occurring m-tuples also increases for uneven tap settings.

We also noted from our experiments that, for a given choice of filter function and tap setting, as the size of the LFSR increased, the number of m-tuples which do not appear in the output sequence remained constant once a certain size was reached for the LFSRs we tested. For example, there were 17 non-occurring 10-bit m-tuples for a 6-input Boolean function when $L \geq 15$ to 20.

Observation 3.4: Distribution of m-tuples for NLFGs using consecutive tap settings are similar regardless of the size of the LFSR.

The distributions of the proportions of m-tuples for NLFGs using consecutive tap settings were similar regardless of the size of the LFSR. However, this was not the case for uneven tap settings. For NLFGs using uneven tap settings, the standard deviation of the m-tuples in terms of proportions are different for different LFSR lengths, tap settings and Boolean functions.

4 Discussion

In this section, we consider the potential impact of biased m-tuple distributions in the output sequences from NLFGs. These sequences are used keystream for stream ciphers, in initialisation functions and as building blocks for message authentication codes (MAC).

Some stream ciphers use the output of NLFGs as keystream to encrypt messages. There is a potential major flaw in this design choice if the NLFG has a highly biased m-tuple distribution. Firstly, there is a possibility of mounting a distinguishing attack on the keystream. If an attacker were to perform a statistical analysis on the m-tuple outputs, they might be able to mount a distinguishing attack based on the frequency of the various m-tuples in the keystream. Another possible attack is a ciphertext-only attack on the stream cipher. Biased m-tuple distribution combined with the redundancy of the plaintext may provide leakage of information to allow an attack to partially decrypt ciphertext messages without initial knowledge of the secret key. An example of an attack which exploits biased eight-tuple distributions in RC4 is the ciphertext-only attack by Mantin and Shamir (2001).

Modern stream ciphers use a secret key and a publicly known initialisation vector (IV) as input to an initialisation function to generate the initial state of the keystream generator. This initialisation function should be nonlinear. A potential problem with using the output of a nonlinear filter for initialisation is that if m-tuples occur more often than others, then it is possible that some initial states will occur more often than others, resulting in biased keystream distribution. In the case where some m-tuples do not occur at all, this means some initial states might not occur at all for any key-iv pair, reducing the effective key space of the stream cipher.

In recent years, stream cipher designers have proposed ciphers which aim to provide simultaneous confidentiality and integrity protection. These are commonly called authenticated encryption (AE) stream ciphers. Some AE stream ciphers use nonlinear filter generators in components used to compute the Message Authentication Code (MAC) tag. One example of such a cipher is Sfinks (Braeken *et al*., 2005). For MAC algorithms which make use of nonlinear filters, the distribution of MAC tags for messages may not be uniform. An attacker may be able to exploit this in a MAC collision attack.

5 Conclusion and Future Work

In this paper, we examined the output of various NLFGs and analysed the distribution of m-tuples in the output sequence for $m \in \{2,3,\ldots,13\}$. We show that the m-tuple distributions of NLFGs are biased, regardless of tap settings used, although the bias is generally greater when the tap settings to the filter function are consecutive. In some cases, there are some m-tuples which do not occur at all in the outputs. This happens for small m-values if the NLFGs use consecutive tap settings rather than uneven tap settings. The experiments also show that the frequency distributions of m-tuples for NLFGs using consecutive tap settings are similar regardless of the size of the LFSR.

The findings in this experiment may have cryptanalytic applications. The significant m-tuple bias in the output sequence may be exploited in attacks ranging from distinguishing attacks to ciphertext-alone attacks. If a NLFG is to be used in a cryptographic application, we recommend against consecutive tap settings.

A limitation of the work is the use of only three balanced Boolean functions of input sizes 5, 6 and 7 bits. This makes it difficult to draw conclusions about the effect of the Boolean function itself on the m-tuple distributions of NLFG output sequences. Further experiments investigating the m-tuple distribution of NLFG sequences formed using Boolean functions with the same number of inputs but with different nonlinearity or algebraic degree remains future work.

References

Anderson, R. (1994). Searching for the optimum correlation attack. In Preneel, B. (editor). Fast Software Encryption 1994, Volume 1008 of LNCS, pp 137–143. Springer.

Braeken, A., Lano, J., Mentens, N., Preneel, B., & Verbauwhede, I. (2005). SFINKS : A Synchronous Stream Cipher for Restricted Hardware Environments. Available from http://www.ecrypt.eu.org/stream/ciphers/sfinks/sfinks.ps

Faugère, J.-C., & Ars, G. (2003). An Algebraic Cryptanalysis of Nonlinear Filter Generators using Gröbner bases. Available from http://www.inria.fr/rrrt/rr-4739.html

Hell, M., Johansson, T., & Meier, W. (2005). Grain - A Stream Cipher for Constrained Environments. Available from http://www.ecrypt.eu.org/stream/p3ciphers/grain/Grain_p3.pdf

Jansen, C., Helleseth, T., & Kolosha, A. (2005). Cascade Jump Controlled Sequence Generator and Pomaranch Stream Cipher (Version 3). Available from http://www.ecrypt.eu.org/stream/p3ciphers/pomaranch/pomaranch_p3.pdf

Mantin, I., & Shamir, A., (2002). A Practical Attack on Broadcast RC4. In Matsui, M. (editor). Fast Software Encryption 2001. Volume 2355 of LNCS, pp 152–164. Springer.

Simpson, L. R. (2000). Divide and Conquer Attacks on Shift Register Based Stream Ciphers. PhD thesis, Queensland University of Technology.