Heuristic Search for Non-Linear Cryptanalytic Approximations

Juan M. E. Tapiador, Julio C. Hernandez-Castro, and John A. Clark

***Abstract*— In this work, we show that heuristic techniques (particularly Simulated Annealing) can be successfully applied in the search of good non-linear approximations of cryptographic primitives. We also provide some experimental results, including two excellent non-linear approximations for the output of the Salsa20 stream cipher with 2 and 4 rounds. From these two approximations, very efficient distinguishers for Salsa20 could easily be obtained, leading to a much more practical attack that any other published so far against this cipher.**

I. Introduction

The adoption of the Data Encryption Standard (DES) [24], [25] provided an extraordinary stimulus for the development of public cryptology, and particularly for the advancement of modern cryptanalytic methods. In the context of DES analysis, Reeds and Manferdelli [26] introduced in 1984 the idea of "partial linearity", pointing out that a block cipher with such a characteristic may be vulnerable to known- or chosen plaintext attacks faster than exhaustive key search. Chaum and Evertse subsequently extended the notion of a per round linear factor to that of "sequence of linear factors" [3], proving that DES versions with more than 5 rounds had no partial linearity caused by such a sequence. These concepts were later generalised to that of "linear structure" [7], which embraced properties such as the complementarity of DES or the existence of bit independencies (i.e. some bits of the ciphertext being independent of the values of certain plaintext and key bits), among others.

In a sense, linear cryptanalysis constituted the natural extension to these efforts. Linear cryptanalysis was introduced by Matsui in [16] as a potential technique to attack DES. Its application was corroborated soon after [17], in what is commonly accepted as the first –although barely practical at the time– compromise of the cipher.

In a linear cryptanalytic attack, we are interested in identifying a linear relation between some bits of the plaintext (P), of the ciphertext (C), and of the key (K), such that:

$$P[\alpha] \oplus K[\beta] \oplus C[\gamma] = 0 \tag{1}$$

where the notation $D[\lambda]$ is used to represent a general linear xor of bits (specified by λ) of the data block D.

The most useful situation for the cryptanalyst is that such a relation exists and holds all the time (or never). Even though this is not likely to occur in a cipher with a minimal strength, a slight deviation with respect to the optimal behaviour can be of interest to attack the cipher.

If we *randomly* select $p + k + c$ bits (for plaintext, key and ciphertext, respectively) and place them into equation (1), the probability that it holds would be very close to $1/2$. Therefore, if a cipher displays a tendency for an equation to hold (or not to hold) with high probability, this can be seen as an evidence of the cipher's poor randomization abilities. The further the expression is from holding with probability $1/2$, the better for the cryptanalyst. As usual, we will note the amount by which the probability of a given expression deviates from $1/2$ as its *bias* (often denoted by the Greek letter ϵ).

Most of current cipher designs operate by repeating a set of basic operations (collectively known as a round) a parameterizable number of times. As the number of rounds increases, the cryptographic strength of the cipher usually grows, together with the cost of encryption and decryption. When carrying out a kind of linear cryptanalysis, the first step is to obtain the highest possible bias for the non-linear components within each round (e.g. a S-box), or even for the whole cipher with a very reduced number of rounds. Once this is done, the bias of the approximation over the whole cipher can be approximated by the next lemma:

Piling-Up Lemma (Matsui [16]). Given n independent, random binary variables $X_1, \ldots, X_n$, with biases $\epsilon_1, \ldots, \epsilon_n$, respectively, then:

$$\mathrm{Prob}\left[\bigoplus_{i=1}^{n} X_i = 0\right] = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i \tag{2}$$

In cryptanalytic terms, the Piling-Up Lemma allows us to approximately infer the complexity of the attack over the full cryptographic primitive starting from results over reduced round versions. For a bias $\epsilon \in [0, 1/2]$, the complexity of an attack is approximately $\mathcal{O}(2^{-2log_2(\epsilon)})$.

A. Extensions to Linear Cryptanalysis

Several refinements to the basic idea of linear cryptanalysis have attempted to improve the efficiency of the attacks, either in specific circumstances or in all cases. As soon as 1994, Kaliski and Robshaw proposed an extension based on the use of multiple linear approximations [13]. Harpes, Kramer and Massey [11] presented in 1995 a generalisation in which linear expressions are replaced by I/O sums. An I/O sum is the XOR of a balanced binary-valued function of the input and a balanced binary-valued function of the output.

Beyond these improvements, the most natural idea is to consider whether the linear approximations on which

Juan M. Estevez-Tapiador is Visiting Research Fellow at the Department of Computer Science, University of York, York YO10 5DD, England, UK. (email: jet@cs.york.ac.uk). Julio C. Hernandez-Castro is Associate Professor at the Department of Computer Science, Carlos III University of Madrid, 28091 Leganes, Madrid, Spain. (email: jcesar@inf.uc3m.es). John A. Clark is Professor of Critical Systems at the Department of Computer Science, University of York, York YO10 5DD, England, UK. (email: jac@cs.york.ac.uk).

1-4244-1340-0/07$25.00 ©2007 IEEE

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.

TABLE I
LINEAR AND NON-LINEAR APPROXIMATIONS TO DES S-BOXES S5 AND S1 (REPRODUCED FROM [15]).

S-box	Input	Output	Bias
S5	x_4	$y_0 \oplus y_1 \oplus y_2 \oplus y_3$	20/64
S5	x_4	$y_1 \oplus y_2 \oplus y_3$	10/64
S1	x_2	y_2	2/64
S5	$x_1 \oplus x_0x_1 \oplus x_0x_4 \oplus x_1x_5 \oplus x_4x_5 \oplus x_0x_1x_5 \oplus x_0x_4x_5$	$y_0 \oplus y_1 \oplus y_2 \oplus y_3$	24/64
S5	$x_1 \oplus x_3 \oplus x_0x_3 \oplus x_0x_5 \oplus x_1x_3 \oplus x_1x_5 \oplus x_0x_1x_3 \oplus x_0x_2x_5$	$y_1 \oplus y_2 \oplus y_3$	18/64

the basic rationale relies can be replaced with non-linear approximations. In 1996, Knudsen and Robshaw introduced the idea of extending Matsui's linear cryptanalytic techniques to the more general case in which non-linear relations are also considered (see [15]). To motivate this approach, they provide a practical example showing that it is feasible to obtain much more accurate approximations to DES S-boxes by considering non-linear relations instead of linear ones (see Table I). In that same work, they identified non-linear approximations to the S-boxes in LOKI91 [2], a DES-like block cipher that operates on 64-bit blocks and uses a 64-bit key. One of the most remarkable features of LOKI91 is that it uses four identical S-boxes which map 12 to 8 bits. While the three best linear approximations known to these S-boxes exhibited biases of 88/4096, 108/4096 and 116/4096, the authors found non-linear relations with biases of 136/4096, 130/4096 and 110/4096.

B. Motivation and Related Work

Linear cryptanalysis was proposed to attack block ciphers and mainly applied to Feistel-like constructions [8]. In this type of design, the effect of a round is entirely linear (and often independent of the key) except for a single component, which is typically implemented using one or more S-boxes. It is precisely in this context wherein the search for (linear) approximations of these components makes sense.

As S-boxes (specially in the past) were often not too big, the search space in which to look for linear approximations is small enough, in many cases, to allow an exhaustive search in a reasonable amount of time. As a result, the method for determining the best linear approximation has not been itself a matter of extensive study (an early exception to this was [18]). However, the situation becomes dramatically different when considering non-linear approximations: there are 2^{2^n} different Boolean functions of n variables (recall that only 2^n are linear). Even for a low number of inputs (e.g. $n = 8$) the search space is astronomically huge, so a brute-force approach will not work.

There is, however, a different but very related field that has been tackled quite successfully by applying heuristic search techniques: the design and analysis of cryptographic Boolean functions.

The design of Boolean functions with desirable cryptographic properties (e.g. high non-linearity, low autocorrelation, high algebraic degree, reasonable order of correlation immunity, etc.) has traditionally been a central area of cryptological research. In the latter half of the 1990s, a few works suggested that heuristic search techniques could be applied to efficiently derive good boolean functions.

Millan et al. [19] were the first to show that small changes to the truth table of Boolean functions do not radically alter their non-linearity nor their autocorrelation properties. This provides for an efficient delta-fitness function for local searches. Later works by the same authors (e.g. [20]) demonstrated that Boolean functions that are correlation-immune or satisfying the strict avalanche criterion can be found too with smart hill-climbing.

This approach was subsequently generalised in a number of ways. Millan et al. applied it to the design of bijective [21] and regular [22] S-boxes. Clark and Jacob used Simulated Annealing in [4] to achieve some results hitherto unattained by other means. Some of the results presented in that work constituted also a counter-example to a then existing conjecture on autocorrelation.

C. Contribution and Overview

The main purpose of this paper is to show that heuristic methods very similar to that used for designing cryptographic Boolean functions can be applied to find non-linear approximations to cryptographic primitives. Section II introduces the specific application on which we will concentrate: the search for non-linear approximations to stream ciphers, and particularly to Salsa20. In Section III, we describe the experimental setup used. The most relevant results of our experiments are shown in Section IV.

Many extensions and improvements to the method described here can be made. In Section V we make a few concluding remarks and outline some possible extensions to this work.

II. SEARCHING FOR NON-LINEAR APPROXIMATIONS TO STREAM CIPHERS

Even though the main target of linear cryptanalysis (and its extensions) are block ciphers, the underlying ideas can be applied to a number of other cryptographic primitives. In this paper, we illustrate an approach which is focused on stream ciphers.

For simplicity, we assume that both the key and the IV are fixed for each sought approximation. We shall return to this point later in the conclusions. Consider now that the keystream generated by the cipher is analyzed with a sliding window of size $N = n + m$ bits, denoted by

$$[x_0, x_1, \ldots, x_{n-1}, y_0, y_1, \ldots, y_{m-1}]$$

We implicitly assume that this is the order in which the bits are produced, i.e. x_0 is generated before x_1, x_1 before x_2, and so on.

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.

In a (simplified) linear cryptanalytic attack, we are interested in identifying a linear relation between bits generated at different points in time, i.e.:

$$f(x_0, \ldots, x_{n-1}) \oplus g(y_0, \ldots, y_{m-1}) = 0 \qquad (3)$$

with both $f(\cdot)$ and $g(\cdot)$ being linear Boolean functions.

If such a relation holds with a significant bias, the cipher can be attacked in a number of ways, the most immediate being the construction of a keystream predictor. Having observed the first $x_0, \ldots, x_{n-1}$ and $y_0, \ldots, y_{m-2}$, it is straightforward to infer the value of $g(y_0, \ldots, y_{m-1})$ from equation (3).

Of particular interest to us in this work is the case when $f(\cdot)$ is non-linear and $g(\cdot)$ is a linear function. However, the general case in which both functions are non-linear is quite interesting and has many other implications too.

The search for such approximations is the key objective of this paper. We illustrate our approach by using Salsa20 with a reduced number of rounds. This stream cipher is briefly described in what follows.

A. The Salsa20 Stream Cipher

Salsa20 is a hash function which, used in counter mode, has been proposed as a stream cipher for eSTREAM, the ECRYPT Stream Cipher Project. It ciphers a 64-byte plaintext block by hashing the key, a nonce, and a block number, then performing an xor with the plaintext.

Salsa20 is quite an interesting design, in which its author, Daniel J. Bernstein, deliberately chooses not to use any complex operations, mixing instead many very efficient operations that consistently perform better across different hardware platforms. For example, no multiplications or S-box lookups are performed not only to avoid their irregular performance in different architectures but also to stop any timing attacks.

Salsa20 uses many rounds, 20 are recommended by the author, although two variants with 12 and 8 rounds were also considered secure and proposed for different scenarios. Each round is composed of simple, efficient operations such as addition mod 2^{32}, constant distance rotations, and xors.

The author claims that Salsa20 simultaneously aims for a high security level, high speed in software, and high speed in hardware. On the Pentium III, Salsa20 with the proposed 20 rounds is able of encrypting at a speed of 14 cycles per byte.

1) Salsa20 Security: The Salsa20 design is quite conservative, as 20 rounds seems to be far enough from the best published attack (7 rounds) as to provide an adequate security margin. It uses keys of 32 or 16 bytes, and 8 byte nonces, together with an 8 byte block counter.

Regarding side-channel attacks, Salsa20 has been designed to be resistant to timing attacks, and to be easy to protect against differential power attacks.

A preliminary diffusion study performed by the author can be found in [1], showing that after four rounds diffusion seems to be optimal.

In 2005 Paul Crowley published a truncated differential cryptanalysis attack (taking 2^{165} operations) on Salsa20/5 [6]. This attack was followed in 2006 by another by Fischer, Meier, Berbain, Biasse, and Robshaw [9] that significantly improved on Crowley's and extended it to 6 rounds, though around 2^{177} operations were necessary. Later, Tsunoo, Saito, Kubo, Suzaki, and Nakashima mounted an attack on 7 rounds with a cost of around 2^{184} operations [30].

This clearly implies that the security margin in Salsa20/8 is uncomfortably low, as acknowledged by Bernstein. It is important to recall that although better than exhaustive key search, all these attacks are quite far from being practical.

Some additional measures have been taken in Salsa20 design to difficult attacks such as those that have recently broken most of the hash functions in use: the control that the cryptanalyst has over the contents of the internal cipher state has been reduced, so Wang's recent construction on MD5 collisions are much more difficult to apply because attackers never have total control of all the input bits.

In a recent article by Turan, Doganaksoy and Calik [28], six new statistical tests are presented and used to analyze the randomness properties of various eSTREAM stream ciphers proposals. These tests, instead of only examining the randomness properties of the keystream, concentrate on the correlations between keys, IVs, internal state and keystream. These tests have been successful in detecting deviations in other eSTREAM stream ciphers proposals such as Decim, F-FCRS-8, Frogbit, Mag and Zk-Crypt, but no statistically significant deviation from randomness was found in Salsa20, which has also survived another study [29] by the same authors that additionally found weaknesses in Polar Bear.

Salsa20 easily passes the tests included in another work by Juhani and Saarinen [12], which showed that the usual suspects MAG, Frogbit, F-FCSR Decim, ZK-Crypt, together with POMARANCH, are all susceptible to Chosen-IV Statistical Attacks.

III. Experimental Setup

We wish to explore the space of possible approximations by searching for a solution that maximizes the relation established in expression (3). The search space has a size of $\mathcal{O}(2^{2^n+2^m})$, which makes exhaustive search impossible (by classical computational means, at least).

In the experiments reported in this paper, we have used the well-established technique of Simulated Annealing [14]. This is a search heuristic inspired by the cooling processes of molten metals. Basically, it can be seen as a basic hill-climbing coupled with the probabilistic acceptance of non-improving solutions. This mechanism allows a local search that eventually can escape from local optima.

The search starts at some initial state (solution) $S_0 \in \mathbb{S}$. The algorithm employs a control parameter $T \in \mathbb{R}^+$ known as the temperature. This starts at some positive value T_0 and is gradually lowered at each iteration, typically by geometric cooling: $T_{i+1} = \alpha T_i,\ \alpha \in (0, 1)$.

At each temperature, a number MIL (Moves in Inner Loop) of neighbor states are attempted. A candidate state

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.

C in the neighborhood $N(S_i)$ of S_i is obtained by applying some move function to S_i. The new state is accepted if it is better than S_i (as measured by a fitness function $F : \mathbb{S} \rightarrow \mathbb{R}$). To escape from local optima, the technique also accepts candidates which are slightly worse than S_i, meaning that its fitness is no more than $|T \ln U|$ lower, with U a uniform random variable in $(0, 1)$. As T becomes smaller, this term gets closer to 0, so as the temperature is gradually lowered it becomes harder to accept worse moves.

The algorithm terminates when some stopping criterion is met, usually after a fixed number MaxIL of inner loops have been executed, or when some maximum number MUL of consecutive inner loops without improvements have been reached.

The basic algorithm is shown in Figure 1. In our experiments, we used the parameters shown in Table II.

A. Solutions and Move Function

The representation and operations for evolving Boolean functions used in this work are based on those provided by Fuller, Millan and Dawson in [10]. Each candidate approximation S is a structure comprising functions $f(\cdot)$ and $g(\cdot)$. Function $f(\cdot)$ is represented by its truth table (TT), while $g(\cdot)$ is simply a linear mask. This representation of $g(\cdot)$ (rather than using another truth table for it) makes the search faster.

To obtain a candidate C in the neighborhood of S, we have defined a move function governed by three (actually two) parameters:

- The probabilities P_f and P_g, with $P_f + P_g = 1$, of defining a neighbour of S either by changing $f(\cdot)$ and keeping $g(\cdot)$ unaltered, or vice-versa.
- Once we have selected which function will be altered, the probability P_c controls how many bits of the function will be flipped.

The basic operation of the move function is as follows:

```
1   Pick U1 ∈ (0,1) with uniform probability
2   if U1 < Pf then
3     for each bit i in the TT of f(·) do
4       Pick U2 ∈ (0,1) with uniform probability
5       if U2 < Pc then
6         Flip bit i
7   else
8     for each bit i in linear mask g(·) do
9       Pick U2 ∈ (0,1) with uniform probability
10      if U2 < Pc then
11        Flip bit i
```

B. Evaluation and Fitness

The fitness function used to guide the search is simply the bias achieved by the approximation:

$$F(S) = \epsilon$$

This is computed by generating a sufficiently large amount of keystream and analysing it with a sliding window of $N = n + m$ bits, as described above. For each evaluation, a certain number of keystream bits are generated by fixing

```
S ← S0
T ← T0
repeat until stopping criterion is met
    repeat MIL times
        Pick C ∈ N(S) with uniform probability
        Pick U ∈ (0,1) with uniform probability
        if F(C) > F(S) + T ln U then
            S ← C
    T ← αT
```

Fig. 1. Basic Simulated Annealing for maximization problems.

TABLE II
SIMULATED ANNEALING PARAMETERS.

GENERAL	
Maximum number of inner loops	1000
Maximum number of moves in inner loop	1000
Maximum number of failed inner loops	250
Initial temperature	200
Cooling rate	0.95
PRNG	Mersenne Twister
MOVE FUNCTION	
Probability of change	0.05
Probability of moving $f(\cdot)$	0.5
Probability of moving $g(\cdot)$	0.5
EVALUATION	
n	8 bits
m	8 bits
Number of evaluations per individual	8192

IV=0 and key=0 (we used keys of 128 bits.) The first N bits are then placed into $f(\cdot)$ and $g(\cdot)$, and a test is carried out to determine whether they produce the same value or not. Then the window moves to the right and the process is repeated.

Upon reaching the end of the keystream, an estimation for the bias can be computed as:

$$\epsilon = \left| \frac{E}{N_{att}} - \frac{1}{2} \right|$$

E being the number of times both outputs coincided and N_{att} the number of attempts tried. For subsequent analysis, it is also useful to define this magnitude in absolute terms:

$$Hits = \left| E - \frac{N_{att}}{2} \right|$$

Here it is important to note that, under the assumption of random input, the expected hit rate is $\frac{N_{att}}{2}$ due to the linearity of $g(\cdot)$. In a general case where $f(\cdot)$ and $g(\cdot)$ are both non-linear, the expected hit rate is not necessarily $1/2$, but depends on the Hamming weight of both functions.

IV. RESULTS AND ANALYSIS

We ran 35 experiments with the configuration described above for Salsa20 with 2 rounds, and another 35 for 4 rounds. The five best approximations found for each version of the

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.

TABLE III

5 BEST APPROXIMATIONS FOUND IN 35 EXPERIMENTS FOR 2 AND 4 ROUNDS.

App.	Cipher	$f(x_0, \ldots, x_7)$	$g(y_0, \ldots, y_7)$	Bias
A1	Salsa20/2	0180687A2FDDB0890EC9D87F94800BDB81BFDD477310DDC54EFF10814801B610	01	0.293579
A2	Salsa20/2	2070C328C0788FD84848530680010C40A09094C98C4A20018DFFF7BE1699F265	03	0.287109
A3	Salsa20/2	1070D22200788FD82C485306C880CC40A0909009AC0A20280FFF77B61ED9FE29	03	0.286621
A4	Salsa20/2	FEAFE6ED571EC03EEF2F7C1DF7BFFA04DF10F37E70ED08A3BEFFB0F9F734BAA2	02	0.284302
A5	Salsa20/2	FEAFF6EC511E92BEFD2EF41DFB9D7B24DE90F17770F508A39EFFB0E9F736BA8A	02	0.283203
B1	Salsa20/4	2C5716C8F0F6A905E9D1D71FA7D9FC38CBD2FE5BBC1DB7FB0A2D2024B58A13D0	63	0.078247
B2	Salsa20/4	41B7B1AFEEC114D7C7B6720BF0C1FE1387B7567FCF37486FEA8CB91ECC914676	58	0.075195
B3	Salsa20/4	4D11E847E69B41DB16214EA197400DB9B9A49DB9E5B5FFC768E44889D3158DAF	83	0.074951
B4	Salsa20/4	7D94F9601212A1CA607E06FD1A0D7D6743BCDBA1250BAA5A27615DD98743C1B6	1D	0.074341
B5	Salsa20/4	47986C5A44CC8347EC663120686481DD8DA03A53AD380A4C62C4325A316B5F5B	95	0.073242

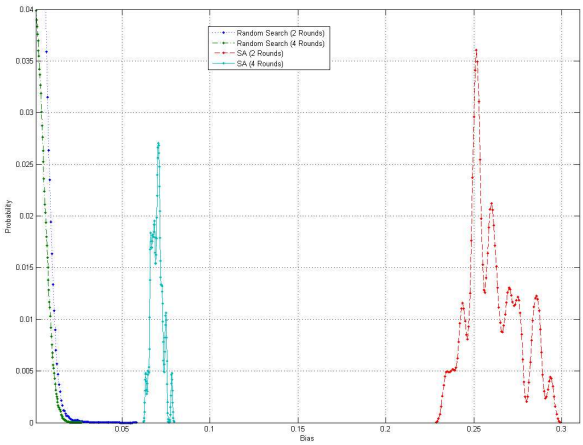

Fig. 2. Bias distribution of the approximations.

TABLE IV

STATISTICAL SIGNIFICANCE.

Approx.	Bias	Equivalent value observed in a $N(0,1)$	Probability
A1	0.293579	26.57	$< 10^{-21}$
A2	0.287109	25.99	$< 10^{-21}$
A3	0.286621	25.94	$< 10^{-21}$
A4	0.284302	25.73	$< 10^{-21}$
A5	0.283203	25.63	$< 10^{-21}$
B1	0.078247	7.08	$7.2076 \cdot 10^{-13}$
B2	0.075195	6.81	$6.0087 \cdot 10^{-12}$
B3	0.074951	6.78	$4.8799 \cdot 10^{-12}$
B4	0.074341	6.73	$4.8799 \cdot 10^{-12}$
B5	0.073242	6.63	$1.6784 \cdot 10^{-11}$

cipher are shown in Table III. The cryptanalytic significance of these biases will be discussed later.

Figure 2 shows the bias distribution corresponding to the 70 approximations. The Simulated Annealing clearly outperforms the results expected by a simple random search. For Salsa20/2, the approximations present biases between 0.234009 and 0.293579, while biases for Salsa20/4 are between 0.063477 and 0.078247. In all the cases, the average time required to run a search was of around 2 hours in a typical PC. The best approximation for each version of the cipher are provided in Algebraic Normal Form in Table V at the end of the paper.

A. *Effectiveness of the Search*

Now we provide a brief analysis concerning the significance of our results. We first start analyzing whether they are statistically significative or not.

If we perform 8192 experiments with a $B(1/2, 1)$, the result should behave as a $B(1/2, 8192)$, whose standard deviation is $\sigma = \sqrt{0.5 \cdot 0.5 \cdot 8192} \simeq 45.2548$. As 8192 is large enough, a Binomial $B(1/2, 8192)$ can be well approximated by a $N(4096, 2048)$; or, equivalently

$$\frac{Hits - 4096}{\sigma} \sim N(0, 1)$$

Even when the input provided to the cipher is completely random, if the number of experiments is high enough, a random search would find candidates increasingly "better." Therefore, the number of total evaluations performed during the search process (say I) should be somehow incorporated into the analysis. For our purposes, if

$$1 - \frac{1}{I} = \mathrm{erf}(\frac{n}{\sqrt{2}})$$

where

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

is the Gauss error function, then we should expect a number of hits at around n standard deviations from the mean.

In the case of the best results obtained for 2 rounds, a bias of 0.293579 implies 1203 hits, which translates to $1023/\sigma \simeq 26.57$ standard deviations away from the mean. This is statistically equivalent to observe a value of 26.57 coming from a $N(0, 1)$, an extremely unusual event with an associated probability of occurrence of less than 10^{-21}. This

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.

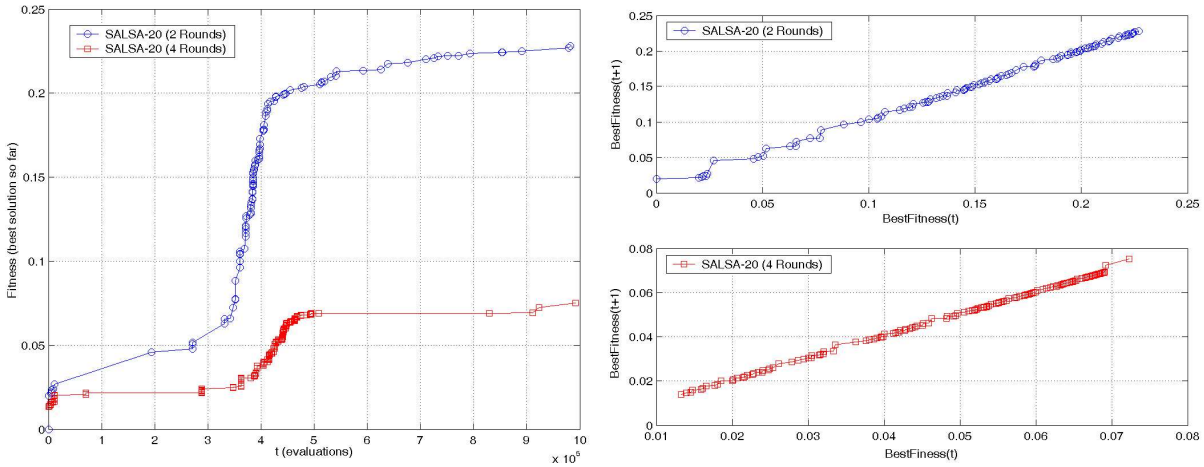

Fig. 3. Search dynamics during a typical running. The left graph shows the evolution of the bias corresponding to the best approximation found so far. In the right graph, it is shown how the improvement yielded by a new best solution with respect to the previous one is mostly a sequence of jumps followed by a linear refinement.

means that by following a pure blind search, the average number of evaluations required to yield this number is more than 10^{21} (recall that our search is bounded by 10^6 evaluations.) Table IV shows the equivalent value observed in a $N(0,1)$ and its associated probability for the the 10 best approximations found.

B. Cryptanalytic Significance

The cryptanalytic implications of these results are clear: for the huge bias on Salsa20/2 is far from surprising due to the low number of rounds, the results over Salsa20/4 are much more relevant, as they provide for a extraordinarily efficient distinguisher being able to set apart the output of Salsa20/4 with null key and IV from that of a true random number generator, with a complexity of only around 2^{13} encryption operations. This result can, obviously, be generalised to be applicable for any keys and IVs. This is relevant as has direct implications over the Salsa20/8 variant proposed by the author for speed oriented applications, as this result could suggest that the security margin for Salsa20/8 is not enough because very efficient attacks against Salsa20/4 exist. Many authors start to get nervous about security margins when attacks over versions with half the proposed rounds are easy and efficient, as in this case. It is important to recall that other attacks exist for up to Salsa20/7, but these are largely impractical and have no definite short-term implications. Some additional work should be done to try to extend the best found approximations to higher numbers of rounds.

C. Search Dynamics

Figure 3 shows the evolution of the fitness value associated with the best candidate found so far in a typical search. Around the first 30% of the movements are completely random, during which the algorithm tries to locate a good candidate in the search space. In almost all the executions tried, the next behaviour has been identical: once an "appropriate" candidate is found, its fitness is considerably incremented in the next few movements. This corresponds to the rapid growth observed in the curve. The resulting candidate after this stage is subsequently "refined" by using the remainder movements, though the improvement is often not significant.

We can extract two main conclusions from this behaviour of a typical search. First, the heuristic is obviously working at improving the quality of the candidates, reaching approximations that might not have been found by a blind search. On the other hand, around a 85% of the movements do not considerably improve the solution, while the remainder 15% –which are mostly consecutive moves– are responsible for all the improvement.

This last analysis is an evidence that, though effective, we are probably not using the best approach in the search. It is not clear whether the fitness function is appropriate for providing guidance, so alternative measures should be considered. Moreover, a restriction of the search space could be useful. In a similar problem, Millan et al. [23] showed how searching over the space of algebraic normal forms of Boolean functions of degree less than or equal to $n/2$ can be used to provide Bent functions of highest possible degree. Clark et al. [5] searched over the space of Walsh spectra for a permutation of an initial spectrum that corresponds to a Boolean function. One of the most interesting features of both papers is that the form of representation used greatly facilitates the restriction of the possible solutions.

In Millan et al.'s work bent function theory shows that the maximal algebraic degree is $n/2$ and so restricting the space to functions with terms of or lower than this degree is clearly well motivated (and ANF is an obvious form to enforce this restriction). In work of Clark et al. working over permuta-

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.

tions of a supplied Walsh spectrum allows various Walsh values to be fixed at convenient values. This allows criteria such as balance, non-linearity and correlation immunity to be "assigned". Walsh spectra are transformed to polar vector form and it is the Boolean-ness that provides the guidance. Essentially, the function space is the set of all vectors induced under inversion by Walsh spectral permutations. Some are Boolean functions (with elements of $+1$ and -1) whilst others are not. In later work, Stanica et al. [27] applied further restrictions to the Walsh spectra by considering only rotation symmetric functions (again with hitherto unattained results).

Both the above works demonstrate in different ways the usefulness of appropriate representations and restrictions. Here we have chosen a family of higher order approximations over the output stream sequence. Restricting $g(\cdot)$ to be a linear function serves good practical purposes. The overall approximation can be interpreted as a predictor for the rightmost bit. Also, under the assumption of bit independence the expected bias for the overall approximation should be 0 whatever the characteristics of $f(\cdot)$. (This is not essential but is certainly convenient.) Allowing $f(\cdot)$ to roam free, as it were, over the space of higher order approximations is, we believe, almost forced by modern cryptographic design! The dangers of linearity are well documented and designers seek to design out linear attacks. If polynomial attacks are designed out then we need to search for higher order attacks to have any chance of success!

V. Conclusions and Future Work

The design of Boolean functions with desirable cryptographic properties has been a field in which many nature-inspired techniques have proven to be very useful. In this paper, we have shown how similar approaches may be applied to search for (non-linear) cryptanalytic approximations. Even though we have experimented with a reduced number of rounds and with a fixed key/IV, the results above show clearly that even simple heuristics can discover effective discriminators.

As a result, we have found some excellent non-linear approximations for the output of Salsa20/2 and Salsa20/4. These are useful for mounting a realistic and very efficient distinguishing attack over Salsa20/4 in a very straightforward way, giving the human cryptanalyst an excellent starting point for guiding his work.

In the results reported in this paper, we have experimented with non-linear approximations over 8 variables. The analysis of non-linear relations over longer portions of data will be studied in future work.

As discussed above, the particular shape of the search dynamics is an evidence that some refinements should be made to the approach followed here. First, the search space seems to be too big, so imposing constraints on the form of the solution (i.e. the algebraic degree) seems a good starting point. Guiding functions more sophisticated than the bias alone should also be considered.

Resilience to the sets of approximations we have used is a barely researched field, and we suspect that many ciphers may be susceptible to these attacks. From this point of view, our work is only in its initial stage.

References

[1] D.J. Bernstein, "Snuffle 2005: the Salsa20 encryption function – Salsa20 Diffusion." Available at http://cr.yp.to/snuffle/diffusion.html, March 2007.

[2] L. Brown, M. Kwan, J. Pieprzyk, and J. Seberry. "Improving Resistance to Differential Cryptanalysis and the Redesign of LOKI." ASIACRYPT'91. *Lecture Notes in Computer Science*, Vol. 453, pp. 36–50. Springer-Verlag, 1993.

[3] D. Chaum and J.-H. Evertse, "Cryptanalysis of DES with a Reduced Number of Rounds; Sequences of Linear Factors in Block Ciphers." CRYPTO'85. *Lecture Notes in Computer Science*, Vol. 218, pp. 192–211. Springer-Verlag, 1986.

[4] J.A. Clark and J.L. Jacob, "Two Stage Optimisation in the Design of Boolean Functions." ACISP'00. *Lecture Notes in Computer Science*, Vol. 1841, pp. 242-254. Springer-Verlag, 2000.

[5] J.A. Clark et al., "Almost Boolean Functions: The Design of Boolean Functions by Spectral Inversion." CEC'03. IEEE Press, 2004.

[6] P. Crowley, "Truncated Differential Cryptanalysis of Five Rounds of Salsa20." SASC'06. Cryptology ePrint archive, report 2005/375, Oct. 2005. Available at: *http://eprint.iacr.org/2005/375*

[7] J.-H. Evertse, "Linear Structures in Blockciphers," EUROCRYPT'87. *Lecture Notes in Computer Science*, Vol. 304, pp. 249–266. Springer-Verlag, 1988.

[8] H. Feistel, "Cryptography and Computer Privacy." *Scientific American*, Vol. 228, No. 5, pp. 15–23, 1973.

[9] S. Fischer, W. Meier, C. Berbain, J.-F. Biasse, and M. Robshaw, "Non-Randomness in eSTREAM Candidates Salsa20 and TSC-4." INDOCRYPT'06. *Lecture Notes in Computer Science*, Vol. 4329, pp. 2–16. Springer-Verlag, 2006.

[10] J. Fuller, W. Millan, and E. Dawson, "Efficient Algorithms for Analysis of Cryptographic Boolean Functions." AWOCA-02, Frasier Island, Australia, 2002.

[11] C. Harpes, G.G. Kramer, and J.L. Massey, "A Generalization of Linear Cryptanalysis and the Applicability of Matsui's Piling-Up Lemma." EUROCRYPT'95. *Lecture Notes in Computer Science*, Vol. 921, pp. 24–38. Springer-Verlag, 1995.

[12] M. Juhani, O. Saarinen, "Chosen-IV Statistical Attacks on eSTREAM Stream Ciphers." ECRYPT Stream Cipher Project. Available at *http://www.ecrypt.eu.org/stream/papersdir/2006/013.pdf*, March 2007.

[13] B.S. Kaliski and M.J.B. Robshaw, "Linear Cryptnalaysis using Multiple Approximations." CRYPTO'94. *Lecture Notes in Computer Science*, Vol. 839, pp. 26–39. Springer-Verlag, 1994.

[14] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi, "Optimization by Simulated Annealing." *Science* 220(4598):671–680, May 1983.

[15] L.R. Knudsen and M.J.B. Robshaw, "Non-Linear Approximations in Linear Cryptanalysis." EUROCRYPT'96. *Lecture Notes in Computer Science*, Vol. 1070, pp. 224–236. Springer-Verlag, 1996.

[16] M. Matsui, "Linear Cryptanalysis Method for DES cipher." EUROCRYPT'93. *Lecture Notes in Computer Science*, Vol. 765, pp. 386–397. Springer-Verlag, 1994.

[17] M. Matsui, "The First Experimental Cryptanalysis of the Data Encryption Standard." CRYPTO'94. *Lecture Notes in Computer Science*, Vol. 839, pp. 1–11. Springer-Verlag, 1994.

[18] M. Matsui, "On Correlation between the Order of S-boxes and the Strength of DES." EUROCRYPT'94. *Lecture Notes in Computer Science*, Vol. 850, pp. 366–375. Springer-Verlag, 1995.

[19] W. Millan, A. Clark, and E. Dawson, "Smart Hill-Climbing Finds Better Boolean Functions." ICICS'97, *Lecture Notes in Computer Science*, Vol. 1334, pp. 149-158. Springer-Verlag, 1997.

[20] W. Millan, A. Clark, and E. Dawson, "Heuristic Design of Cryptographically Strong Balanced Boolean Functions." EUROCRYPT'98, *Lecture Notes in Computer Science*, Vol. 1403, pp. 489-499. Springer-Verlag, 1998.

[21] W. Millan, "How to Improve the Non-Linearity of Bijective S-boxes." ACISP'98. *Lecture Notes in Computer Science*, Vol. 1438, pp. 181-192. Springer-Verlag, 1998.

[22] W. Millan, L. Burnett, G. Carter, A. Clark, and E. Dawson, "Evolutionary Heuristics for Finding Cryptographically Strong S-boxes." ICICS'99, *Lecture Notes in Computer Science*, Vol. 1403, pp. 489-499. Springer-Verlag, 1998.

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.

TABLE V

ALGEBRAIC NORMAL FORM OF THE BEST APPROXIMATIONS FOUND TO 2 AND 4 ROUNDS

R	$f(x_0, \ldots, x_7)$	$g(y_0, \ldots, y_7)$	Bias
2	$x_0 \oplus x_2 \oplus x_0x_2 \oplus x_0x_1x_2 \oplus x_0x_3 \oplus x_0x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_0x_4 \oplus x_2x_4 \oplus x_0x_2x_4 \oplus x_3x_4 \oplus x_1x_3x_4 \oplus x_0x_2x_3x_4 \oplus x_5 \oplus x_1x_5 \oplus x_1x_2x_5 \oplus x_3x_5 \oplus x_1x_3x_5 \oplus x_0x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_0x_2x_3x_5 \oplus x_1x_2x_3x_5 \oplus x_0x_4x_5 \oplus x_1x_4x_5 \oplus x_0x_1x_4x_5 \oplus x_1x_2x_4x_5 \oplus x_0x_1x_2x_4x_5 \oplus x_0x_1x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_0x_2x_3x_4x_5 \oplus x_0x_1x_2x_3x_4x_5 \oplus x_6 \oplus x_0x_6 \oplus x_2x_6 \oplus x_0x_2x_6 \oplus x_3x_6 \oplus x_0x_3x_6 \oplus x_0x_2x_3x_6 \oplus x_1x_2x_3x_6 \oplus x_0x_1x_2x_3x_6 \oplus x_1x_4x_6 \oplus x_0x_1x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_1x_3x_4x_6 \oplus x_1x_2x_3x_4x_6 \oplus x_5x_6 \oplus x_0x_1x_5x_6 \oplus x_0x_2x_5x_6 \oplus x_0x_1x_2x_5x_6 \oplus x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_0x_2x_3x_5x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_0x_1x_2x_4x_5x_6 \oplus x_1x_3x_4x_5x_6 \oplus x_2x_3x_4x_5x_6 \oplus x_1x_2x_3x_4x_5x_6 \oplus x_7 \oplus x_0x_7 \oplus x_0x_1x_7 \oplus x_0x_2x_7 \oplus x_1x_2x_7 \oplus x_0x_1x_2x_7 \oplus x_0x_1x_3x_7 \oplus x_0x_4x_7 \oplus x_0x_1x_4x_7 \oplus x_0x_2x_4x_7 \oplus x_0x_1x_2x_4x_7 \oplus x_3x_4x_7 \oplus x_0x_1x_3x_4x_7 \oplus x_1x_2x_3x_4x_7 \oplus x_0x_1x_2x_3x_4x_7 \oplus x_5x_7 \oplus x_0x_5x_7 \oplus x_1x_5x_7 \oplus x_0x_1x_5x_7 \oplus x_0x_2x_5x_7 \oplus x_1x_2x_5x_7 \oplus x_3x_5x_7 \oplus x_0x_3x_5x_7 \oplus x_1x_3x_5x_7 \oplus x_2x_3x_5x_7 \oplus x_1x_2x_3x_5x_7 \oplus x_4x_5x_7 \oplus x_1x_4x_5x_7 \oplus x_2x_4x_5x_7 \oplus x_1x_2x_4x_5x_7 \oplus x_0x_1x_2x_4x_5x_7 \oplus x_3x_4x_5x_7 \oplus x_0x_1x_3x_4x_5x_7 \oplus x_2x_3x_4x_5x_7 \oplus x_6x_7 \oplus x_1x_6x_7 \oplus x_2x_6x_7 \oplus x_0x_3x_6x_7 \oplus x_1x_3x_6x_7 \oplus x_0x_1x_3x_6x_7 \oplus x_2x_3x_6x_7 \oplus x_0x_2x_3x_6x_7 \oplus x_0x_4x_6x_7 \oplus x_0x_2x_4x_6x_7 \oplus x_1x_2x_4x_6x_7 \oplus x_3x_4x_6x_7 \oplus x_0x_1x_3x_4x_6x_7 \oplus x_1x_5x_6x_7 \oplus x_0x_1x_5x_6x_7 \oplus x_3x_5x_6x_7 \oplus x_0x_3x_5x_6x_7 \oplus x_1x_3x_5x_6x_7 \oplus x_2x_3x_5x_6x_7 \oplus x_0x_1x_2x_3x_5x_6x_7 \oplus x_4x_5x_6x_7 \oplus x_0x_4x_5x_6x_7 \oplus x_1x_4x_5x_6x_7 \oplus x_0x_1x_4x_5x_6x_7 \oplus x_2x_4x_5x_6x_7 \oplus x_0x_2x_4x_5x_6x_7 \oplus x_1x_2x_4x_5x_6x_7 \oplus x_0x_1x_2x_4x_5x_6x_7 \oplus x_1x_3x_4x_5x_6x_7 \oplus x_0x_2x_3x_4x_5x_6x_7 \oplus x_0x_1x_2x_3x_4x_5x_6x_7$	y_0	0.293579
4	$x_0x_1 \oplus x_1x_2 \oplus x_0x_1x_2 \oplus x_0x_3 \oplus x_1x_3 \oplus x_0x_1x_3 \oplus x_2x_3 \oplus x_1x_2x_3 \oplus x_4 \oplus x_0x_2x_4 \oplus x_1x_2x_4 \oplus x_3x_4 \oplus x_0x_3x_4 \oplus x_2x_3x_4 \oplus x_5 \oplus x_0x_5 \oplus x_0x_1x_5 \oplus x_2x_5 \oplus x_0x_2x_5 \oplus x_1x_2x_5 \oplus x_0x_1x_2x_5 \oplus x_0x_3x_5 \oplus x_0x_1x_3x_5 \oplus x_2x_3x_5 \oplus x_0x_2x_3x_5 \oplus x_1x_4x_5 \oplus x_0x_2x_4x_5 \oplus x_3x_4x_5 \oplus x_2x_3x_4x_5 \oplus x_6 \oplus x_0x_1x_6 \oplus x_0x_2x_6 \oplus x_0x_3x_6 \oplus x_1x_3x_6 \oplus x_2x_3x_6 \oplus x_0x_1x_2x_3x_6 \oplus x_4x_6 \oplus x_0x_4x_6 \oplus x_1x_4x_6 \oplus x_0x_1x_4x_6 \oplus x_1x_2x_4x_6 \oplus x_0x_1x_2x_4x_6 \oplus x_3x_4x_6 \oplus x_0x_3x_4x_6 \oplus x_0x_5x_6 \oplus x_1x_2x_5x_6 \oplus x_0x_1x_2x_5x_6 \oplus x_0x_3x_5x_6 \oplus x_1x_3x_5x_6 \oplus x_0x_1x_3x_5x_6 \oplus x_2x_3x_5x_6 \oplus x_0x_2x_3x_5x_6 \oplus x_1x_2x_3x_5x_6 \oplus x_4x_5x_6 \oplus x_1x_4x_5x_6 \oplus x_2x_4x_5x_6 \oplus x_0x_1x_2x_4x_5x_6 \oplus x_3x_4x_5x_6 \oplus x_0x_3x_4x_5x_6 \oplus x_0x_2x_3x_4x_5x_6 \oplus x_1x_2x_3x_4x_5x_6 \oplus x_0x_1x_2x_3x_4x_5x_6 \oplus x_7 \oplus x_1x_7 \oplus x_0x_2x_7 \oplus x_3x_7 \oplus x_1x_3x_7 \oplus x_0x_1x_3x_7 \oplus x_1x_2x_3x_7 \oplus x_0x_4x_7 \oplus x_1x_4x_7 \oplus x_2x_4x_7 \oplus x_0x_1x_2x_4x_7 \oplus x_3x_4x_7 \oplus x_0x_3x_4x_7 \oplus x_1x_3x_4x_7 \oplus x_0x_1x_2x_3x_4x_7 \oplus x_5x_7 \oplus x_0x_5x_7 \oplus x_0x_1x_5x_7 \oplus x_2x_5x_7 \oplus x_1x_2x_5x_7 \oplus x_0x_1x_2x_5x_7 \oplus x_3x_5x_7 \oplus x_1x_3x_5x_7 \oplus x_0x_2x_3x_5x_7 \oplus x_0x_1x_2x_3x_5x_7 \oplus x_4x_5x_7 \oplus x_1x_4x_5x_7 \oplus x_2x_4x_5x_7 \oplus x_1x_2x_4x_5x_7 \oplus x_0x_3x_4x_5x_7 \oplus x_1x_3x_4x_5x_7 \oplus x_0x_1x_3x_4x_5x_7 \oplus x_2x_3x_4x_5x_7 \oplus x_1x_2x_3x_4x_5x_7 \oplus x_0x_1x_2x_3x_4x_5x_7 \oplus x_0x_1x_6x_7 \oplus x_0x_2x_6x_7 \oplus x_0x_1x_2x_6x_7 \oplus x_3x_6x_7 \oplus x_0x_1x_3x_6x_7 \oplus x_1x_2x_3x_6x_7 \oplus x_4x_6x_7 \oplus x_0x_4x_6x_7 \oplus x_1x_4x_6x_7 \oplus x_0x_1x_4x_6x_7 \oplus x_1x_2x_4x_6x_7 \oplus x_0x_3x_4x_6x_7 \oplus x_1x_2x_3x_4x_6x_7 \oplus x_0x_5x_6x_7 \oplus x_1x_5x_6x_7 \oplus x_0x_1x_5x_6x_7 \oplus x_2x_5x_6x_7 \oplus x_0x_2x_5x_6x_7 \oplus x_2x_3x_5x_6x_7 \oplus x_0x_2x_3x_5x_6x_7 \oplus x_4x_5x_6x_7 \oplus x_0x_1x_2x_4x_5x_6x_7 \oplus x_3x_4x_5x_6x_7 \oplus x_1x_3x_4x_5x_6x_7 \oplus x_0x_1x_3x_4x_5x_6x_7 \oplus x_2x_3x_4x_5x_6x_7 \oplus x_0x_1x_2x_3x_4x_5x_6x_7$	$y_0 \oplus y_1 \oplus y_5 \oplus y_6$	0.078247

[23] W. Millan et al., "Evolutionary Generation of Bent Functions for Cryptography." CEC'03. IEEE Press, 2003.

[24] National Bureau of Standards (NBS). *Data Encryption Standard*. U.S. Department of Commerce, FIPS Publication 46. January, 1977.

[25] National Institute of Standards and Technology (NIST). *Data Encryption Standard*. FIPS Publication 46-2. December 30, 1993.

[26] J.A. Reeds and J.L. Manferdelli, "DES Has no Per Round Linear Factors." CRYPTO'84. *Lecture Notes in Computer Science*, Vol. 196, pp. 377–389. Springer-Verlag, 1985.

[27] P. Stanica, S. Maitra and J.A. Clark, "Results on Rotation Symmetric Bent and Correlation Immune Boolean Functions." FSE'04. *Lecture Notes in Computer Science*, Vol. 3017, pp. 161–177. Springer-Verlag, 2004.

[28] M.S. Turan, A. Doganaksoy, and C. Calik, "Statistical Analysis of Synchronous Stream Ciphers." ECRYPT Stream Cipher Project. Available at *http://www.ecrypt.eu.org/stream/papersdir/2006/012.pdf*, March 2007.

[29] M.S. Turan, A. Doganaksoy, and C. Calik, "Detailed Statistical Analysis of Synchronous Stream Ciphers." ECRYPT Stream Cipher Project. Available at *http://www.ecrypt.eu.org/stream/papersdir/2006/043.pdf*, March 2007.

[30] Y. Tsunoo, T. Saito, H. Kubo, T. Suzaki, and H. Nakashima, "Differential Cryptanalysis of Salsa20/8." SASC'06. Available at *www.ecrypt.eu.org/stream/papersdir/2007/010.pdf*, March 2007.

Authorized licensed use limited to: University of York. Downloaded on September 11, 2009 at 05:21 from IEEE Xplore. Restrictions apply.