The COSvd Ciphers

Eric Filiol*[1] Caroline Fontaine[2] Sébastien Josse[1]

[1] Laboratoire de virologie et de cryptologie
ESAT/SSI - B.P. 18
35998 Rennes - France
efiliol@esat.terre.defense.gouv.fr
[2] CNRS-LIFL, Bâtiment M3
59655 Villeneuve d'Ascq cedex – France
Caroline.Fontaine@lifl.fr

Abstract. This paper presents the strengthened version of COS stream cipher family denoted COSvd. It is a vectorized stream cipher, hence producing blocks of keystream. We particularly focus on the member of this family producing 128-bit blocks mainly from a 256-bit key. The design is based on the *crossing over* technique presented in [17] which allows to vectorize stream ciphering by using nonlinear shift registers. A highly nonlinear layer combined with a auto-controlled chaotic module yields an extremely high cryptographic security and prevents any existing attack. This cipher design offers a much higher speed encryption than any existing stream ciphers or block ciphers while insuring a high level of security. A 500$ rewarded cryptanalysis challenge is proposed.

Keywords: stream cipher, nonlinear feedback shift register, vectorized cipher, high speed encryption, Boolean functions, chaotic layer.

1 Introduction

We are interested, in this paper, in the design of secret key cipher schemes. There are two families for such schemes: stream ciphers, and block ciphers. Both have advantages and drawbacks, and our purpose is here to present the design of an "hybrid" cipher, which has advantages of both, without their drawbacks. Hence, two approaches are possible: to simulate stream ciphers with block ciphers, or to simulate block ciphers with stream ciphers. As you will see below, the second one seems more interesting, and the cipher we will present is mainly a vectorized stream cipher, which acts as a block cipher.

Composing cryptographic primitives is a well-known manner to build some other primitives. One famous example is that of block ciphers which can simulate stream ciphers through output feedback mode [7, 18]. The main advantage of this approach is to obtain provable security. For example, attack on such simulated stream ciphers would, indeed, be equivalent to cryptanalyze the underlying block

* *Corresponding author*

cipher. Such an approach is described in [5] for the ANSI message authentication code, built from block cipher. However there are few exceptions to this security assumption: as an example, CBC mode revealed vulnerabilities independently from the quality of the underlying block cipher. However, building stream ciphers from block ciphers or hash functions represents a major drawback: block ciphers are usually much slower than real, true stream ciphers. Moreover they are not as suitable as stream ciphers, for VLSI implementations. The explosion of today's need for secure communications asks for very high encryption speed that block ciphers achieve with some difficulty. By now, fastest block cipher software implementations run at nearly 260 Mbits/s [4].

On the other hand, security of most frequent and common variants of stream ciphers (based on linear feedback shift registers and suitable Boolean functions) becomes little by little challenged and questioned. In particular, the security of synchronized stream ciphers (that is to say, stream ciphers whose output bits may be easily described by an equation of low degree) has been weakened by recent new powerful attacks [8, 10, 14], proving that effective, real-life cryptanalysis becomes more and more reachable.

One approach which has not been very often considered so far, is to consider and take the best of both worlds: to simulate block cipher by vectorizing stream ciphers. It nevertheless would combine the advantage of both sides without their respective drawbacks. First known attempts have been presented in [28, 29] with keyed hash functions built from stream ciphers and in [2] with block ciphers built from SHA-1 and SEAL.

In this paper, we present an improved and more secure version of a new approach in cipher design, COS, we published in 2001 [17]. By using only the known cryptographically strongest Boolean functions and Non Linear Feedback Shift Register (NLFSR), we build a family of vectorized binary additive stream ciphers (namely where output bits are produced as a block of L bits in a row) called *COSvd ciphers* (COS version for defense), with arbitrary output block size. It used the same basic design, called *crossing over* COS cipher family was based on. This design allows to consider the internal state at instant t as constituent of the ciphering output blocks or vectors. These latter are bitwise xored to blocks of plaintext, and produce blocks of ciphertext. But whilst COS and COSvd are based on the same concept of crossing over, COSvd distinguish itself by a highly nonlinear layer combined with a new kind of chaotic module greatly, which increase the overall security. Thus, no relation between output bits and the key bits can be even formulated and exploited by any existing cryptanalysis technique. In particular, the only significant attacks on one member of the COS family [37] is no longer applicable for COSvd family.

In terms of security, we primarily rely on the cryptographic strength of the constituent primitives. The crossing over mechanism allows then to combine them in a very secure way. The key size can be 128, 192 and 256 bits long with an additional key represented by two pairs of real values. These aspects already insure a high level of security while the additional highly nonlinear, chaotic layer offers suitable security for data protection requiring a relatively high clearance.

COSvd has been chosen as the core of the file system encryption software TURENNE for the security of sensitive data of restricted clearance. Given fast software implementation of Boolean functions and shift registers, our scheme reveals itself far better than previous ones in practical applications requiring high encryption speed. With 128-bit block size, software experiments have reached very high speed encryption: about 300 Mbits/s with 128-bit block size and still nearly 1Gbits/s in a 512-bit block size setting.

This paper is organized as follows. We first present in Section 2 the COSvd cipher design itself, and particularly the 128-bit block size version. Section 3 will discuss some security aspects. Section 4 presents the performance of some of our implementations (encryption speed and code size). Section 5 finally presents the rewarded cryptanalysis challenge we propose.

More materials and data on this cipher (C implementation, test vectors, ...) will be soon available on corresponding author's web page.

2 The COSvd Ciphers

COSvd ciphers is a stream cipher family which is an improvement of the COS one, published in [17]. In some steps, it is quite similar (key setup, crossing over mechanism), but in other ones really different (only one operating mode is considered, and a nonlinear module has been added), and stronger.

2.1 General Description

The COSvd cipher's design exclusively centers on $n+1$ nonlinear feedback shift registers (NLFSR): $M, L_1, L2, \ldots, L_n$.

Consider the $2L$-bit block size version. The M register is $4L$ bits long, and is devoted to the key setup, that is to say, to the generation of the initial states of registers $L_1, L_2, \ldots, L_n$. Registers L_i generate the output ciphering blocks through the crossing over mechanism. They each are $2L$ bits long and are irregularly clocked.

The feedback Boolean functions of these n registers have been chosen according to their cryptographic properties. They are optimally *balanced*, *highly nonlinear* and have high *correlation-immunity order*. Moreover, their *Algebraic Normal Forms* (ANF), that is to say, their representations as multivariate polynomials, have to present some particular structure to meet important security requirements, as defined in [24]. We have considered the strongest known functions meeting all these criteria. After numerous simulations, a careful choice has been done among the best Boolean functions recently proposed in literature. Optimal functions have been taken: an eleven-variables function for the M register feedback and nine-variables functions for the L_i register feedback.

The central point of COSvd design lies in the crossing over technique inspired by the same mechanism as in chromosome genetic differentiation (hence the name COS standing for Crossing Over System). Let us Consider n registers $L_1, L_2, \ldots, L_n$ of length $2L$. If $MSB(L_i)$ and $LSB(L_i)$ denote respectively the

L most significant bits and the L least significant bits of L_i, then one output block generation step can be summarized as follows:

1. Clock L_i at least L times.
2. Generate the $2(n-1)$ output L-bit blocks B_k ($k = 1, \ldots, 2(n-1)$) as follows, for ($i \neq j$)
$$MSB(B_k) = MSB(L_i) \oplus LSB(L_j)$$
$$LSB(B_k) = LSB(L_i) \oplus MSB(L_j)$$
3. Clock L_j ($j \neq i$) irregularly.
4. $i = i + 1 \bmod n$

Before being xored to plaintext blocks, L-bit output blocks are filtered through a Highly Non-Linear Layer (HNLL module for short). This module combines an optimal substitution primitive with a chaotic function based on Hénon map.

For more details on the security provided by all these modules, see Section 3. In terms of performance, we see that n $2L$-bit registers produce, in each step, $2(n-1)$ L-bit blocks with only slightly $2L$ register clockings. Performance evaluation will be exposed in Section 4.

¿From a general point of view, we then will speak of a $(n, 2L)$ COSvd cipher to describe one particular member of the COSvd family. We now present more deeply the $(2, 128)$ cipher. Details of implementation and specifications will be found soon on corresponding author's web page.

2.2 Key Setup

Whatever may be the value of n and L, any $(n, 2L)$ COSvd cipher works with an internal key K of 256 bits and a message key MK of 32 bits. So the first step in key setup expands shorter keys of either 128 or 192 bits.

Let us represent the M register (256 bits) by eight 32-bit words $M[0], M[1], \ldots, M[7]$. In the same way, we will represent the key by $K[0], \ldots, K[s]$ where $s = 3, 5, 7$ according to the user's key size, and $L_1[0], \ldots, L_1[3], L_2[0], \ldots, L_2[3]$ as the L_1 and L_2 registers.

We first fill up the M register with the user's key:

$$M[i] = K[i] \qquad i = 0, \ldots, s$$

The expansion step then takes the user's key to produce lacking bits to fill up remaining part of the M register $M[s+1], \ldots, M[7]$. A look-up table T will give 8-bit blocks from 8-bit blocks of the user's key. Due to lack of space, this table will be found on corresponding author's web page. To be more precise if K_i denotes the 8-bit block of the user's key starting at bit i, we have:

– $s = 3$ (user's key has 128 bits)

$$M[4] = 2^{24} * T[K_{248}] + 2^{16} * T[K_{240}] + 2^8 * T[K_{232}] + T[K_{224}]$$
$$M[5] = 2^{24} * T[K_{216}] + 2^{16} * T[K_{208}] + 2^8 * T[K_{200}] + T[K_{192}]$$
$$M[6] = 2^{24} * T[K_{184}] + 2^{16} * T[K_{176}] + 2^8 * T[K_{168}] + T[K_{160}]$$
$$M[7] = 2^{24} * T[K_{152}] + 2^{16} * T[K_{144}] + 2^8 * T[K_{136}] + T[K_{128}]$$

– $s = 5$ (user's key has 192 bits)

$$\begin{aligned} M[6] &= 2^{24} * T[K_{248} \oplus K_{216} \oplus K_{184}] + 2^{16} * T[K_{240} \oplus K_{208} \oplus K_{176}] \\ &\quad +2^{8} * T[K_{232} \oplus K_{200} \oplus K_{168}] + T[K_{224} \oplus K_{192} \oplus K_{160}] \\ M[7] &= 2^{24} * T[K_{152} \oplus K_{120} \oplus K_{80}] + 2^{16} * T[K_{144} \oplus K_{112} \oplus K_{80}] \\ &\quad +2^{8} * T[K_{136} \oplus K_{104} \oplus K_{72}] + T[K_{128} \oplus K_{96} \oplus K_{64}] \end{aligned}$$

The 32-bit message key MK is then combined with initial state of register M:

$$M[0] = M[0] \oplus MK$$

This key MK is to be changed for every different message. Since the intrinsic security of the cipher does not rely on MK, it can be transmitted with the message. Its role is to prevent different messages to be sent with the same base key K.

The M register is then clocked 256 times. The eleven-variables feedback function *f11* take bits $2, 31, 57, 87, 115, 150, 163, 171, 201, 227$ and 255 of M as input bits and output one feedback bit (see Figure 1). Function *f11* having a too

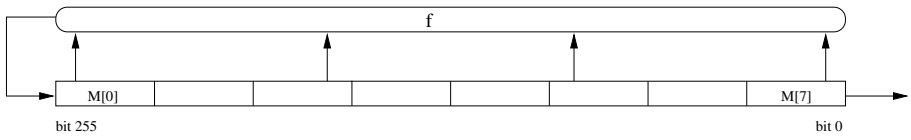

Fig. 1. Clocking of the M register

big representation cannot be given here due to lack of space. It will be found on corresponding author's web page. Its properties are recalled in Section 3.1.

After these 256 clockings, M register internal state provides initialization of L_1 register in this way:

$$L_1[i] = M[i+4] \qquad i = 0, \ldots, 3$$

M is then clocked 128 times and L_2 is initialized as follows:

$$L_2[i] = M[i] \qquad i = 0, \ldots, 3$$

2.3 Encryption - Decryption

The output blocks of the cipher are xored with either plaintext blocks (encipherment) or ciphertext blocks (decipherment) we just need to describe how these 128-bit blocks are generated. Each step S_i outputs one 128-bit block and decomposes as follows:

1. Compute clocking values d and d'.
   (a) Compute $clk = 2 * lsb(L_2) + lsb(L_1)$.
   (b) $d = C[clk]$ where $C[0, \ldots, 3] = \{64, 65, 66, 64\}$.
   (c) Compute $clk = 2 * msb(L_1) + msb(L_2)$.
   (d) $d' = C'[clk]$ where $C'[0, \ldots, 3] = \{41, 43, 47, 51\}$.
2. Clock L_1 if i even or L_2 if i odd, d times.
3. Clock L_2 if i even or L_1 if i odd, d' times.
4. Produce a 128-bit block B_i in this way (see Figure 2):

$$B_i = (L_1[1] \oplus L_2[3]) + 2^{32} * (L_1[0] \oplus L_2[2]) \\ +2^{64} * (L_1[3] \oplus L_2[1]) + 2^{96} * (L_1[2] \oplus L_2[0])$$

5. Output blocks are finally filtered bytewise by the HNLL module before being xored to the plaintext. Detailed description of this module is given in Section 2.4.

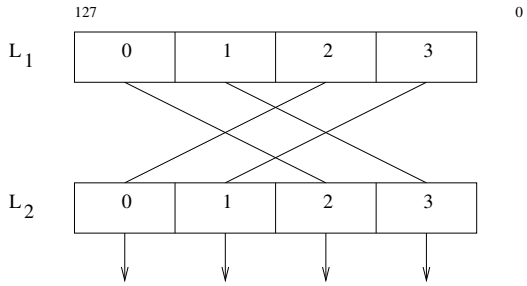

Fig. 2. Generation of Output block

Feedback Boolean functions are in 9 variables (*f9a* and *f9b*) and both use bits $2, 5, 8, 15, 26, 38, 44, 47$ and 57 of L_1 and L_2. Interested readers will find soon the complete functions on corresponding author's web page.

2.4 The HNLL Module

This module aims, together with the crossing-over scheme, at providing a high level of security. Moreover, it forbids all specific attacks against the $(2, 128)$-COS cipher [37]. The overall setting is described in Figure 3. Output blocks B_i are considered bytewise. For each byte $b_7, b_6, \ldots, b_0$, the chaotic module produces 10 bits $h_9, h_8, \ldots, h_0$. Thus, the HNLL module outputs one byte denoted $c_7, c_6, \ldots, c_0$ and computed as follows:

$$(c_7, c_6, \ldots, c_0) = F[(b_7 \oplus h_7, b_6 \oplus h_6, \ldots, b_0 \oplus h_0)][(h_8, h_9)]$$

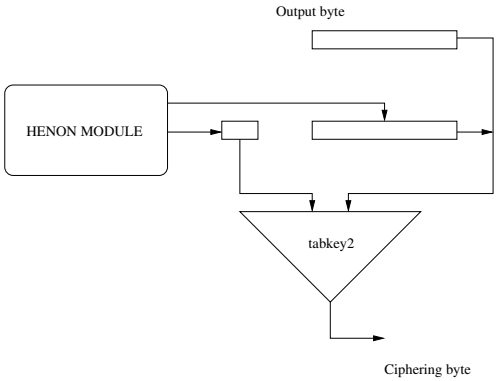

Fig. 3. The HNLL module

Function F is an optimally nonlinear substitution S-box, mapping $\mathbb{F}_2^8 \times \mathbb{F}_2^2$ to $\mathbb{F}_2^8$ and denoted by TABKEY2. Due to lack of space, this table will be found on corresponding author's web page. Its cryptographic properties are discussed in Section 3.3.

The chaotic module is constructed from the Hénon map [19, 26]. This map is defined as follows. Given a pair of points (x_0, y_0) in $\mathbb{R}^2$, let us consider the following recursion equations:

$$\begin{cases} x_{n+1} = 1 + y_n - 1.4x_n^2 \\ y_{n+1} = 0.3x_n \end{cases}$$

At each time instant, output bit z_n is produced as follows:

$$\begin{cases} z_n = 0 \text{ if } x_n \leq 0.39912 \\ z_n = 1 \text{ if } x_n > 0.39912 \end{cases}$$

The initial value (x_0, y_0) is an additional part of the secret key and may be any pair of values chosen inside the geometric area defined in $\mathbb{R}^2$ by the following four points (convergence intervals or domain):

$$(-1.33; 0.42)\ ,\ (1.32; 0.133)\ ,\ (1.245; -0.14) \text{ and } (-1.06; -0.5) \qquad (1)$$

We know that if (x_0, y_0) belongs to this area, then $[x_n, y_n)$ will also belong to it for any $n \geq 0$ [19, 26]. Moreover, the slightest variation of initial values x_0 and y_0 will result in a different sequence (chaotic effect). Hénon map has been proposed by Forre [20] as a constituent primitive for stream cipher. Unfortunately, this map is too weak, as is, for cryptographic purposes as shown by Erdman [13], since the 4-bit subsequence 1100 never appears.

In order to bypass this weakness, two Hénon maps have been considered in parallel, each of them producing respectively sequences $(z_n^1)_{n\geq 0}$ and $(z_n^2)_{n\geq 0}$. We then consider in COSvd the sequence $(z_n^1 \oplus z_n^2)_{n\geq 0}$, which presents suitable

statistical properties. The additional secret key is denoted $\sigma_1 = (x_0, y_0)$ and $\sigma_2 = (x'_0, y'_0)$. In order to prevent user errors in key generation (real values chosen outside the convergence domain defined by (1), this additional key may be any pairs of real values, then transformed by simple operations to produce values in convergence intervals.

In order to prevent compatibility problems between different processors, we use the multiprecision package GMP to implement the chaotic module.

3 Security Evaluation

The security of COSvd cipher lies primarily on the feedback Boolean functions, on the crossing over mechanism and on the HNLL module.

Some aspects of security have already been discussed in [17] concerning the elements which were already present in COS ciphers. This concerns the key setup and the feedback Boolean functions. We just recall them briefly. The key size is large enough to resist exhaustive key search. Moreover the additional key (σ_1, σ_2) greatly increases the security since there exists quite an infinity of pairs of values. In such kind of design with NLFSR, correlation attacks [8, 10, 14, 33], linear syndrome attacks [38, 39], and algebraic attacks [12] are no longer working and even have no longer significance. Indeed they can be used only when register feedback is linear. Known attack on COS proposed by Bao and Wu [37] is no longer effective with the HNLL layer. Thus, concerning the security of COSvd ciphers, mainly three aspects have to be considered and controlled: randomness, presence of cycles and security of the HNLL module.

We first discuss security aspects related to the feedback functions and the crossing over mechanism, and their impact on the presence of cycles. Then, we discuss the statistical aspects, showing that our pseudo-random generator is good. Finally, we discuss the security aspects of the HNLL module, which notably increase the security and make Bao-Wu's cryptanalysis ineffective.

3.1 Boolean Functions and Crossing-over Mechanism

Since these are the same as for the COS family, we just recall the main properties of these two primitives. Definitions of the security criteria on Boolean functions can be found in [15].

- Three Boolean functions have been chosen for the registers feedback: *f11* for the Register M and *f9a*, *f9b* for Registers L_1 and L_2. Function *f11* is common to all members of the COSvd family. Some other functions for the feedback of Registers L_i have been chosen, for cases $n > 2$, or another L.
  All these functions are presenting the best cryptographic properties trade-offs (balancedness, correlation-immunity, nonlinearity). Different simulations have been conducted to choose the most suitable functions. Table 1 summarizes their main characteristics for the $(2, 128)$ COSvd cipher.
  The strong trade-off between correlation-immunity and nonlinearity suppresses any exploitable correlation first, between K and the initializations

of L_1 and L_2, and secondly, between K and the output blocks. Thus, the chosen functions highly contribute to the excellent statistical behavior (see Section 3.2).

	Balanced	Correlation Immunity Order	Non Linearity	Degree
f11	yes	CI(2)	960	7
f9a	yes	CI(2)	224	6
f9b	yes	CI(2)	224	6

Table 1. Characteristics of Feedback Functions of the $(2, 128)$ COSvd Cipher

- The structure of their Algebraic Normal Forms complies to the Golomb's theoretical results [24, chap. VI and VII, page 115]), thus preventing degeneration in the output sequence of the FSR.
- The crossing over scheme aims at strengthening the overall scheme security. First, probability to have cycles in the output sequence is greatly reduced. Forecasting presence of cycles of length strictly less than 2^{L-1} where L is the register length, still remains a general open problem for NLFSR [25]. Only statistical testing (when tractable) can be envisaged up to now to give insight on possible cycles.
  Suppose however that registers L_1 and L_2 present cycles of length respectively l_1 and l_2 for some initializations. The crossing over mechanism will then obviously produce a cycle of length $lcm(l_1, l_2)$ which is confidently supposed to be extremely large. In the general case of a $(n, 2L)$ cipher, cycles we be of length $lcm(l_1, \ldots, l_n)$.
- One other important role of the crossing over mechanism is to suppress any exploitable correlation between output blocks B and internal states of the L_1 and L_2 registers at any time. Since register contents are cross-xored together and since respective bit probability of each register internal bit is exactly $\frac{1}{2}$, it is impossible to guess any internal bit of both L_1 and L_2, by knowing bits of B. In other words, if L_j^i denotes the i-th bit of register L_j and B^k the k-th bit of B then:

$$P[L_1^i = a | B^k = b] = P[L_2^l = a \oplus b | B^k = b] = \frac{1}{2}$$

  where $B^k = L_1^i \oplus L_2^l$. Moreover irregular clocking make such guess far more difficult.

3.2 Statistical Tests

COSvd has been tested with the NIST statistical tests suite (FIPS 140 and STS package [35]). We did not notice any significant bias. Randomness results are rather very good. Output bits behave as coin-tossing experiment, independently from their neighbours.

3.3 Facing cryptanalysis

We will now focuses on the primitives that are strengthening COS scheme: the HNLL module. Thus, we will now consider the existing attacks and will show that none of them is likely to succeed, due to this module.

Attacks on FSRs We will begin with *(fast) correlation attacks*, that are usually designed for breaking stream ciphers based on the combination or filtering of LFSRs [8, 32, 9, 27]. These attacks are not applicable since COSvd consider FSR and not LFSRs, and since no significant correlation may be exhibited due to the S-Box (see later on) and the use of an unpredictable chaotic primitive. This latter forbids the writing of simple linear equations that could be used for (fast) correlation attacks.

Two usual weaknesses of filtered FSRs are *Anderson leakage* [1] and *inversion attack* [23]. They are both based on the fact that if the FSR is of length n, hence the n sequences given by its n cells are all equal, up to the shift operator. Thus, the inputs of the filtering function f are obtained from the same sequence, with small shifts. This results in information leakage from the input to the output, even if f is resilient [1]. But nowadays, no general algorithm is known to exploit this leakage. It is shown in [23] that the chosen tap positions (for filter) must form a proper difference set, or else the system is then vulnerable to an inversion attack; this comes from the fact that the relative shift in the sequences extracted from two tap positions is equal to the distance between them. An existing information leakage, if any, will imply the S-Box output relatively to a noisy version of its input (by the chaotic function) with probability $\frac{1}{2}$. Moreover the irregularly clocking of both the registers make these attacks untractable.

Attacks on the S-box Since our cipher uses an S-box, we also have to mention the usual attacks on such primitives. The best known are the *differential cryptanalysis* [6] and the *linear cryptanalysis* [31]. Each of them is related to a measure of the capacity for an S-box $S : \mathbb{F}_2^m \to \mathbb{F}_2^n$ to resist cryptanalysis; we applied both measures to TABKEY2, as considered as an S-box with $m = 10$ and $n = 8$ (the entries corresponding the the random data are entries number 9 and 10).

Firstly, we computed the number

$$D_S[a][b] = \#\{x \in \mathbb{F}_2^m, S(x + a) + S(x) = b\}$$

for all $a \in \mathbb{F}_2^m$, $a \neq 0$, and $b \in \mathbb{F}_2^n$. Here, $\#\{\}$ denotes the cardinal of the set. The lower is $D_S = \max_{a \neq 0, b} D_S[a][b]$, the better S resists differential cryptanalysis. Here, we get $D_{\text{TABKEY2}} = 38$. We also applied this measure on the $2 \times m$ functions obtained from TABKEY2 by fixing one entry to 1 or 0. The values of the corresponding D are given in Figure 4. We can see that these values are really good.

Secondly, we computed the number

$$N_S[a][b] = \#\{x \in \mathbb{F}_2^m, x \cdot a = S(c) \cdot b\} - 2^{m-1}$$

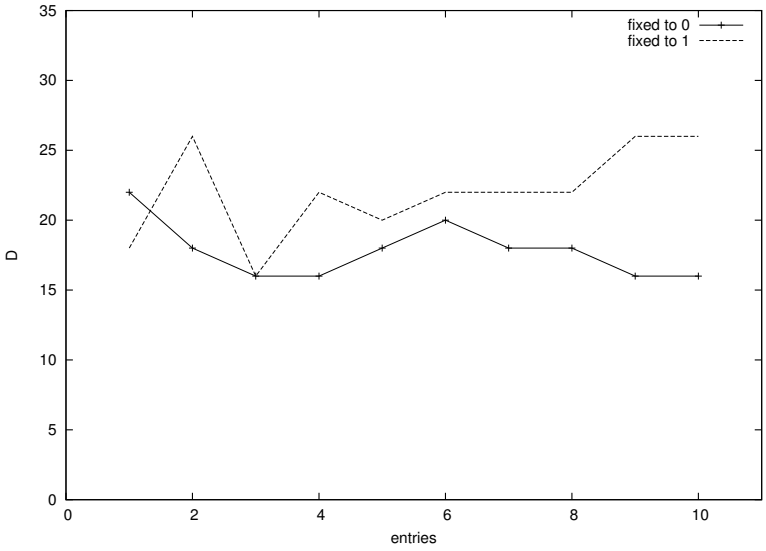

Fig. 4. Differential analysis of TABKEY2, with 1 fixed entry

for all $a \in \mathbb{F}_2^m$, $a \neq 0$, and $b \in \mathbb{F}_2^n$, $b \neq 0$. Here, $\cdot$ denotes the usual scalar product. The lower is $N_S = \max_{a \neq 0, b \neq 0} N_S[a][b]$, the better S resists linear cryptanalysis. Here, we get $N_{\text{TABKEY2}} = 76$. We also applied this measure on the $2 \times m$ functions obtained from TABKEY2by fixing one entry to 1 or 0. The values of the corresponding N are given in Figure 5. We can see that these values are really good.

Previous attacks on COS The only efficient attack published on the original COS cipher is [37]. This attack does not work any more on this new version of COS, because of the use of the S-box and the chaotic module. Mainly, this attack is based on the ability to find particular configurations in the two FSR internal states by guessing the correct clocking. Moreover, it requires the direct knowledge of the output of the crossing over module (modulo 2 addition of the internal state of the FSRs). Due to the chaotic module, the S-box and the irregular clocking of both FSR, such configurations no longer exist. Wu-Bao's attack is no longer effective.

Algebraic Attacks These attacks rely on the existence of equations of low degree expressing, linking secret key bits to the output [3, 12]. Once relinearized, it might be possible to solve the resulting linear system in order to find part of the secret key. Such equations of low degree do not exist with COSvd. By using the technique presented in [16] and based on Möbius transform, each output bits can only be described by a Boolean function of very high degree. By computing all the monomials up to degree 4, we find that the average number of monomials of degree d is $\frac{1}{2}\binom{|K|}{d}$, where $|K|$ is the number of the secret bits. Subsequently, it is not possible to realize here an effective algebraic attack.

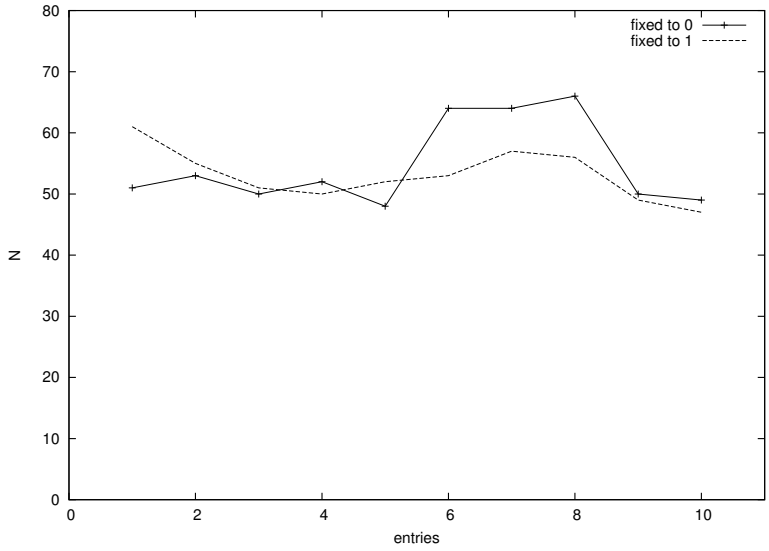

Fig. 5. Linear analysis of TABKEY2, with 1 fixed entry

4 Performance Analysis

A $(n, 2L)$ version of COSvd cipher will produce a $2(n-1)L$-bit block at each iteration, with only L register shifts. Hence, COSvd ciphers are extremely fast ciphers. Performance analysis is presented in Table 2. It has been conducted on a

	Mean Encryption speed
$(2, 128)$	158 Mbits/sec.
$(3, 128)$	283 Mbits/sec.
$(4, 128)$	390 Mbits/sec.
$(3, 256)$	570 Mbits/sec.
$(2, 512)$	600 Mbits/sec.
$(3, 512)$	1100 Mbits/sec.

Table 2. Approximative Encryption Speed of some COSvd ciphers

Pentium IV with 2.4 GHz CPU, 2 Gbytes RAM, under Linux. Compiler was gcc version 3.3 20030226 on a SuSe Linux 8.2. Approximative mean encryption speed (in Mbits/sec.) is given. Performance analysis has been conducted on a relatively optimized C implementation. Loading time of the plaintext into memory has not been

taken into account. Code size after complete compilation is less than 20 Kbytes.

5 Conclusion and Challenge

COSvd ciphers are new, ultrafast, (vectorized) stream ciphers, acting as block cipher, built up entirely with non linear feedback shift registers. They offer a very high speed encryption and a very high cryptographic strength. They particularly exhibit extremely good randomness properties.

However, since public evaluation only can confirm the very high quality of this cipher, we propose a 500 euros cryptanalysis challenge. Conditions and details will be soon available on the corresponding author's web page.

These ciphers are placed under the GNU General Public License.

References

1. R.J. Anderson. Searching for the optimum correlation attack. In *Fast Software Encryption 94*, number 1008 in Lecture Notes in Computer Science, pp 137–143. Springer-Verlag, 1995.
2. R. Anderson, E. Biham, Two Practical and Provable Secure Block Ciphers: BEAR and LION, In *Fast Software Encryption 96*, number 1039 in Lecture Notes in Computer Science, pp 113–120, Springer Verlag, 1997.
3. F. Armknecht, Improving Fast Algebraic Attacks, FSE 2004.
4. K. Aoki, H. Lipmaa, Fast Implementation of AES Candidates, In *Third AES Conference*, April 13-14th, 2000, New York.
5. M. Bellare, J. Killian, P. Rogaway, The security of Cipher Block Chaining, In *Advances in Cryptology - CRYPTO'94*, number 839 in Lecture Notes in Computer Science, pp 341–358, Springer Verlag.
6. E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. In A.J. Menezes and S.A. Vanstone, editors, *Advances in Cryptology - CRYPTO'90*, number 537 in Lecture Notes in Computer Science, pp 2–21. Springer-Verlag, 1991.
7. C.M. Campbell, Design and Specification of Cryptographic Capabilities, *IEEE Computer Society Magazine*, Vol. 16, Nr. 6, pp 15–19, November 1979.
8. A. Canteaut and M. Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In B. Preneel, editor, *Advances in Cryptology - EUROCRYPT'2000*, number 1807 in Lecture Notes in Computer Science, pp 573–588, 2000.
9. V. Chepyzhov and B. Smeets. On a fast correlation attack on certain stream ciphers. In D.W. Davis, editor, *Advances in Cryptology - EUROCRYPT'91*, number 547 in Lecture Notes in Computer Science, pp 176–185. Springer-Verlag, 1991.
10. V. Chepyzhov, T. Johansson, B. Smeets. A Simple Algorithm for Fast Correlation Attacks on Stream Ciphers. In *Fast Software Encryption 2000*, Lecture Notes in Computer Science 1978, Springer Verlag, 2001.
11. D. Coppersmith, S. Halevi, and C. Jutla. Cryptanalysis of stream ciphers with linear masking. In *Advances in Cryptology - CRYPTO 2002*, number 2442 in Lecture Notes in Computer Science, pp 515–532. Springer-Verlag, 2002.
12. N. Courtois, W. Meier, Algebraic Attacks on Stream Cipher with Linear Feedback, In *Advances in Cryptology – EUROCRYPT'03*, number 2656 in Lecture Notes in Computer Science, pp 345–359, Springer.
13. D. Erdmann, S. Murphy Hénon Stream Cipher, *Electronic Letters*, vol. 28, no 9, pp 893–895, 1992.

14. E. Filiol, Decimation Attack of Stream Ciphers, In *Progress in Cryptology - INDOCRYPT'2000*, number 1977 in Lecture Notes in Computer Science, pp 31–42, Springer Verlag, 2000.
15. E. Filiol, C. Fontaine, Highly Nonlinear Balanced Boolean Functions with a Good Correlation-Immunity, In *Advances in Cryptology - EUROCRYPT'98*, number 1403 in Lecture Notes in Computer Science, pp 475–488, Springer Verlag, 1998.
16. E. Filiol, A New Statistical Testing for Symmetric Ciphers and Hash Functions. In : *Proceedings of the 4th International Conference on Information and Communication Security 2002*, Lecture Notes in Computer Science, Springer, 2002.
17. E. Filiol and C. Fontaine. A new Ultrafast Stream Ciphers Design: COS Ciphers. In: *Proceedings of the 8th IMA Conference on Cryptography and Coding*, Lecture Notes in Computer Science 2260, pp. 85–98, Springer Verlag, 2001.
18. National Bureau of Standards, NBS FIPS PUB 81, DES Modes of Operation, U.S. Department of Commerce, Dec 1980.
19. A.P. Fontana, On a proposed symbolic dynamics for the Hénon map, Thesis, Naval postgraduate school, June 1993.
20. R. Forré, The Hénon Attractor as Key Stream Generator, Advances in Cryptology - Eurocrypt'91, LNCS, Springer Verlag, 1991.
21. R. Forré. A fast correlation attack on nonlinearly feedforward filtered shift-register sequences. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology - EUROCRYPT'89*, number 434 in Lecture Notes in Computer Science, pp 586–596. Springer-Verlag, 1990.
22. J. Golić and M. Mihaljevic. A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance. *Journal of Cryptology*, (3):201–212, 1991.
23. J.D. Golić, A. Clark, and E. Dawson. Generalized inversion attack on nonlinear filter generators. *IEEE Transactions on Computers*, 49(10):1100–1109, 2000.
24. S.W. Golomb, Shift Register Sequences, Agean Park Press, 1982.
25. S.W. Golomb, On the Cryptanalysis of Nonlinear Sequences, In *7th IMA Conference on Cryptography and Coding*, number 1746 in Lecture Notes in Computer Science, pp 236–242 Springer Verlag, 1999.
26. M. Hénon, A two-dimensional mapping with a strange attractor. *Communications in Mathematical Physics*, 1976, vol. 50, pp. 69-77.
27. T. Johansson and F. Jnsson. Fast correlation attacks through reconstruction of linear polynomials. In *Advances in Cryptology – CRYPTO'00*, number 1880 in Lecture Notes in Computer Science, pages 300–315, 2000.
28. H. Krawczyk, LFSR-based Hashing and Authentication, In *Advances in Cryptology - CRYPTO'94*, number 839 in Lecture Notes in Computer Science, pp 129–139 Springer Verlag.
29. X.J. Lai, R.A. Rueppel, J. Woolven, A fast Cryptographic Checksum Algorithm based on Stream Ciphers, In *Advances in Cryptology - AUSCRYPT'92*, number 718 in Lecture Notes in Computer Science, pp 338–348 Springer Verlag.
30. J.L. Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, IT-15:122–127, 1969.
31. M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology - EUROCRYPT'93*, number 765 in Lecture Notes in Computer Science, pages 386–397. Springer-Verlag, 1994.
32. W. Meier and O. Staffelbach. Fast correlation attacks on stream ciphers. In C.G. Günther, editor, *Advances in Cryptology - EUROCRYPT'88*, number 330 in Lecture Notes in Computer Science, pages 301–314. Springer-Verlag, 1988.

33. M. Mihaljevic, J.D. Golic, A Fast Iterative Algorithm for a Shift-Register Initial State Reconstruction given the Noisy Output Sequence, In *Advances in Cryptology - AUSCRYPT'90*, number 453 in Lecture Notes in Computer Science, pp 165–175, Springer Verlag, 1990.
34. M.J. Mihaljevic, M.P.C. Fossorier, and H. Imai. Fast correlation attack algorithm with the list decoding and an application. In *Fast Software Encryption 2001*, number 2355 in Lecture Notes in Computer Science, pages 196–210, 2002.
35. Revised NIST Special Publication 800-22, A Statistical Test Suite for the Validation of Random Number Generators and Pseudo-random Numbers Generators for Cryptographic Applications. `http://csrc.nist.gov/rng/rng2.html`
36. T. Siegenthaler. Correlation-immunity of nonlinear combining fonctions for cryptographic applications. *IEEE Transactions on Information Theory*, IT-30(5):776–780, 1984.
37. H. Wu and F. Bao. Cryptanalysis of stream cipher COS $(2, 128)$ mode I. In *Australian Conference on Information Security and Privacy, ACISP 2002*, number 2384 in Lecture Notes in Computer Science, pages 154–158. Springer-Verlag, 2002.
38. K. Zheng, M. Huang, On the Linear Syndrome Method in Cryptanalysis, In *Advances in Cryptology - CRYPTO'88*, number 405 in Lecture Notes in Computer Science, pp 469–478, Springer Verlag, 1990.
39. K. Zeng, C.H. Yang, T.R. Rao, An Improved Linear Syndrome Algorithm in Cryptanalysis with Applications, In *Advances in Cryptology - CRYPTO'90*, number 537 in Lecture Notes in Computer Science, pp 34–47, Springer Verlag, 1991.