Dial C for Cipher

Le chiffrement était presque parfait*

Thomas Baignères** and Matthieu Finiasz

EPFL
CH-1015 Lausanne – Switzerland
http://lasecwww.epfl.ch

Abstract. We introduce C, a practical provably secure block cipher with a slow key schedule. C is based on the same structure as AES but uses independent random substitution boxes instead of a fixed one. Its key schedule is based on the Blum-Blum-Shub pseudo-random generator, which allows us to prove that all obtained security results are still valid when taking into account the dependencies between the round keys. C is provably secure against several general classes of attacks. Strong evidence is given that it resists an even wider variety of attacks. We also propose a variant of C with simpler substitution boxes which is suitable for most applications, and for which security proofs still hold.
Keywords: Block Cipher, provable security, AES, Blum-Blum-Shub generator, decorrelation

1 Introduction

When designing a public key cryptosystem, proving tight security results often requires to rely on hard problems such as factoring or discrete logarithm computation which, by nature, require to manipulate complex objects. When designing block ciphers, speed requirements do not allow to do so. As a consequence, security arguments often rely on heuristic assumptions which, in some cases, might prove wrong. At SAC 2005, Baignères and Vaudenay [5] showed that replacing the substitution boxes of AES by independent perfectly random permutations is enough to prove that 4 rounds are enough to resist linear and differential cryptanalysis and that 10 rounds are enough to resist any iterated attack of order 1. Here, we use the exact same construction, improve some results, and plug in a key schedule based on a provably secure pseudo-random generator. We propose to use the Blum-Blum-Shub pseudo-random generator [15, 16] as its security is well established (even for practical parameters), although any provably secure generator (like for example QUAD [7] or any fast construction based on Goldreich and Levin's hard-core predicate [26]) could be used, possibly leading to faster implementations. We obtain C, a block cipher with a slow key schedule, but as fast as AES when it comes to encryption/decryption and provably secure against most common attacks: linear and differential cryptanalysis, iterated attacks of

* Refering to the famous movie by Alfred Hitchcock *Dial M for Murder* [28], this is how the title of this article should be translated to French.

** Supported by the Swiss National Science Foundation, 200021-107982/1

order 1, impossible differentials and presumably algebraic attacks, slide attacks, boomerang attack, and, to a certain extent, saturation attacks. Note that all the security results we obtain take into account the key schedule. To the best of our knowledge, all current iterated block cipher constructions consider in their "security proofs" that the round keys are statistically independent, which is not true in practice as they all derive from the same key.

We start this article with a detailed description of C and of its key schedule. Ensues a review of all security results on C, starting with those which are proven, and going on with some results which, though not proven, seem quite reasonable. We then present a way of considerably speeding up the key schedule while preserving all security results and finish with implementation considerations.

2 The Block Cipher C

In this paper, a *perfectly random permutation* denotes a random permutation uniformly distributed among all possible permutations. Consequently, when referring to a *random permutation*, nothing is assumed about its distribution.

2.1 High Overview

The block cipher $\mathsf{C} : \{0,1\}^{128} \rightarrow \{0,1\}^{128}$ is an iterated block cipher. It is made of a succession of *rounds*, all identical in their structure. Each round is parameterized by a *round-key* which is derived from the main 128 bits secret key using a so-called *key schedule algorithm*. The structure of each round is made of a (non-linear) substitution layer followed by a (linear) permutation layer. The non-linear part of the round mixes the key bits with the text bits in order to bring *confusion* (in the sense of [43]). The linear part dissipates the eventual redundancy, bringing *diffusion*. Such an iterated block cipher is often referred to as a *substitution-permutation network* (SPN). Several modern block ciphers (such as AES [21] or SAFER [36]) follow this structure. In what follows, we successively detail the SPN of C and its key schedule algorithm.

2.2 The Substitution-Permutation Network

In a nutshell, C follows the same SPN as AES [21], except that there is no round key addition, that the fixed substitution box is replaced by independent perfectly random permutations, and that the last round of C only includes the non-linear transformation. This construction exactly corresponds to the one studied in [5].

C is made of $r = 10$ *independent* rounds $\mathsf{R}^{(1)}, \ldots, \mathsf{R}^{(r)} : \{0,1\}^{128} \rightarrow \{0,1\}^{128}$, so that $\mathsf{C} = \mathsf{R}^{(r)} \circ \cdots \circ \mathsf{R}^{(1)}$. A r round version of C will either be denoted by $\mathsf{C}[r]$ or simply by C when the number of rounds is clear from the context. Each round considers the 128 bit text input as a four by four array of bytes seen as elements of the finite field $\mathrm{GF}(q)$ where $q = 2^8$. Consequently, if $a \in \{0,1\}^{128}$ denotes some input of the round transformation, we will denote a_ℓ (resp. $a_{i,j}$) the ℓ-th (resp. the $(i+4j)$-th) byte of a for $0 \leq \ell \leq 15$ (resp. $0 \leq i, j \leq 3$) and call such an input a *state*. Except for the last one, each round $\mathsf{R}^{(i)}$ successively applies a non-linear transformation $\mathsf{S}^{(i)}$ followed by a linear transformation L so that $\mathsf{R}^{(i)} = \mathsf{L} \circ \mathsf{S}^{(i)}$ for $i = 1, \ldots, r-1$. The last round $\mathsf{R}^{(r)}$ excludes the linear transformation, i.e., $\mathsf{R}^{(r)} = \mathsf{S}^{(r)}$.

The non-linear transformation $\mathsf{S}^{(i)}$ is a set of 16 *independent* and *perfectly random* permutations[1] of $\mathrm{GF}(q)$. Denoting $\mathsf{S}^{(i)} = \{\mathsf{S}_0^{(i)}, \ldots, \mathsf{S}_{15}^{(i)}\}$ the 16 permutations of round i and $a, b \in \{0,1\}^{128}$ the input and the output of $\mathsf{S}^{(i)}$ respectively, we have $b = \mathsf{S}^{(i)}(a) \ \Leftrightarrow \ b_\ell = \mathsf{S}_\ell^{(i)}(a_\ell)$ for $0 \leq \ell \leq 15$. Depending on the level of security/performance one wants to achieve, the round permutations can be *de-randomized* (see Section 5).

The linear transformation L does not depend on the round number. It first applies a rotation to the left on each row of the input state (considered as a four by four array), over four different offsets. A linear transformation is then applied to each column of the resulting state. More precisely, if a, b denote the input and the output of L respectively, we have (considering indices modulo 4)

$$\begin{pmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{pmatrix} = \begin{pmatrix} \mathtt{02} & \mathtt{03} & \mathtt{01} & \mathtt{01} \\ \mathtt{01} & \mathtt{02} & \mathtt{03} & \mathtt{01} \\ \mathtt{01} & \mathtt{01} & \mathtt{02} & \mathtt{03} \\ \mathtt{03} & \mathtt{01} & \mathtt{01} & \mathtt{02} \end{pmatrix} \times \begin{pmatrix} a_{0,j} \\ a_{1,j+1} \\ a_{2,j+2} \\ a_{3,j+3} \end{pmatrix}$$

2.3 The Key-Schedule Algorithm

Generating a perfectly random permutation of $\{0,1\}^8$. As there are $2^8!$ possible permutations of $\{0,1\}^8$, it is possible to define a one to one mapping between $[0\,; 2^8! - 1]$ and the set of permutations of $\{0,1\}^8$. The mapping we choose is described in Table 1. We simply need to derive pseudo-random integers in $[0\,; 2^8! - 1]$ from the 128 bit secret key. As each of the ten rounds involves 16 permutations, we need 160 such integers, representing a total of $160 \cdot \lceil \log_2(2^8!) \rceil = 269440$ pseudo-random bits.

Deriving an extended key from the secret key.

Definition 1. *An extended key of* $\mathsf{C}[r]$ *is a set of* $16 \cdot r$ *integers in* $[0\,; 2^8! - 1]$.

In order to derive an extended key from the 128 secret key, we need to generate $16 \cdot r$ pseudo-random integers of $[0\,; 2^8! - 1]$. We propose to use the Blum-Blum-Shub pseudo-random number generator [16].

[1] Note that a random 8 bit permutation is usually more biased than the substitution box of AES [42, 52]. However this bias is key-dependent and thus does not represent a threat. Biases on the AES box are independent of the key and thus can help to distinguish (reduced rounds of) AES from the perfect cipher when the key is unknown. Exploiting the strong bias of the substitution boxes of C requires to know the location of this bias, which is impossible without the knowledge of the permutation that was used (i.e., of the key). For instance the maximum ELP of the transformation made of a random key addition followed by the AES substitution box is 2^{-6} whereas the perfectly random substitution boxes we use have a maximum ELP of $1/(q-1) \approx 2^{-8}$. Intuitively, a cipher cannot become weaker when replacing an (arbitrary) random permutation by a perfectly random permutation.

Table 1. Defining a one to one mapping from integers between 0 and $2^8!$ onto the set of permutations of $\{0,1\}^8$.

Input: An integer $0 \leq \kappa < 2^8!$
Output: A table π of size 256 such that $\pi[0],\dots,\pi[255] \in \{0,\dots,255\}$ is a permutation of $\{0,1\}^8$ uniquely defined by κ
External Procedure: `EucDiv(a,b)` returns the quotient and remainder of the Euclidean division of a by b.

```
0: q ← κ,   π[0] ← 0,   π[1] ← 1 , . . . ,   π[255] ← 255
1: for m = 256 down to 1
2:    (q, r) ← EucDiv(q, m)
3:    Swap the values of π at positions r and m
4: end for
```

Definition 2. *A prime p is a strong-prime if $(p-1)/2$ is prime. A prime p is a strong-strong-prime if both p and $(p-1)/2$ are strong-primes.*

Let p and q be two (fixed) 1024-bit strong-strong-prime numbers[2], and let $n = p \cdot q$. Considering the secret key k as a 128 bit integer, let $\{x_i \in \mathbf{Z}_n^* \ : \ i = -1,0,1,2,\dots\}$ be the sequence defined by

$$\begin{cases} x_{-1} = k \cdot 2^{894} + 2^{1023} & \text{and} \\ x_i \ \ = x_{i-1}^2 \bmod n & \text{for } i \geq 0. \end{cases}$$

Let BBS $= a_1 b_1 a_2 b_2 \dots$ be the pseudo-random bit string where $a_i, b_i \in \{0,1\}$ respectively denote the least and most significant[3] bits of x_i. We will use BBS to generate the 160 integers we need.

Dividing the BBS sequence into $\lceil \log_2(2^8!) \rceil$-bit substrings, we can obtain pseudo-random integers in $[0\,;2^{\lceil \log_2(2^8!)\rceil}-1]$, thus sometimes larger than $2^8!$. A naive approach to deal with those too large integers is to discard the substrings leading to such integers, thus having to generate $\lceil \log_2(2^8!) \rceil$ more bits each time this happens. This strategy requires the generation of $160 \cdot 2^{\lceil \log_2(2^8!)\rceil}/2^8! \approx 270\,134$ pseudo-random bits in average. More efficient approaches exits (e.g., discarding only a few bits instead of a whole block), but the improvement in terms of efficiency is not worth the loss in terms of clarity.

3 Security Results: What is Known for Sure

3.1 C is resistant to Linear and Differential Cryptanalysis

Linear Cryptanalysis (LC) [37, 38, 45], aims at uncovering correlations between linear combinations of plaintext and ciphertext bits. It is known that the data

[2] Note that strong-strong-primes are always congruent to 3 modulo 4, i.e., are Blum integers. We use strong-strong primes to ensure that the generator will have a long period. See sections 3.5 and 6 for more details.

[3] the most significant bit corresponds to being larger or smaller than $(n-1)/2$.

Table 2. Exact value of $\max_{a\neq 0,b} \mathrm{ELP}^{\mathsf{C}}(a,b)$ (and $\max_{a\neq 0,b} \mathrm{EDP}^{\mathsf{C}}(a,b)$) for various number of rounds.

2 rounds	3 rounds	4 rounds	5 rounds	6 rounds	7 rounds	8 rounds	9 rounds
$2^{-33.98}$	$2^{-55.96}$	$2^{-127.91}$	$2^{-127.91}$	$2^{-127.99}$	$2^{-127.99}$	$2^{-128.00}$	$2^{-128.00}$

complexity of LC is inversely proportional to the linear probability (LP) [17,39]. For given input/output masks $a, b \in \{0,1\}^{128}$ on C, $\mathrm{LP}^{\mathsf{C}}(a,b) = \big(2\Pr[a \bullet X = b \bullet \mathsf{C}(X)] - 1\big)^2$, where the probability is taken over the uniformly distributed input $X \in \{0,1\}^{128}$ and where $\bullet$ denotes a scalar product. $\mathrm{LP}^{\mathsf{C}}(a,b)$ is a function of the random variable C (the randomness coming from the key). The block cipher is considered to be provably secure against LC if, for all input/output masks $a, b \in \{0,1\}^{128}$, the *expected value* $\mathrm{ELP}^{\mathsf{C}}(a,b)$ of the linear probability $\mathrm{LP}^{\mathsf{C}}(a,b)$ (the mean being taken over all possible instances of the block cipher, that is, over all possible keys) is close to the one of the perfect cipher C^*, i.e., close to $1/(q^{16}-1)$ in our case.

Nyberg showed in [41] that the ELP of an iterated block cipher can be expressed as a sum of *linear characteristics*, which means in our case that for any input/output masks $c_0, c_r \in \{0,1\}^{128}$,

$$\mathrm{ELP}^{\mathsf{C}[r]}(c_0, c_r) = \sum_{c_1,..,c_{r-1}} \prod_{i=1}^{r} \mathrm{ELP}^{\mathsf{R}_i}(c_{i-1}, c_i) \geq \max_{c_1,..,c_{r-1}} \prod_{i=1}^{r} \mathrm{ELP}^{\mathsf{R}_i}(c_{i-1}, c_i) \quad (1)$$

where the sum is taken over all possible input/output masks on the individual rounds of C. Choosing a specific characteristic (i.e., a sequence of masks $c_1, \ldots, c_{r-1}$), it is possible to lower bound the value of $\mathrm{ELP}^{\mathsf{C}[r]}(a,b)$. This is usually enough to attack the block cipher: a lower bound on the ELP gives an upper bound on the number of samples needed to perform the attack. However, such a bound is not enough to prove the security of the system, as the cumulative effect of linear hulls (the set of all intermediate masks for given input/output masks) may lead to an attack much more efficient than expected[4]. In security proofs of block ciphers, it is often considered without any formal justification that one characteristic is overwhelming, so that the sum in (1) is of same order than the max. In our case, for any input/output masks $a, b \in \{0,1\}^{128}$, the *exact* value of $\mathrm{ELP}^{\mathsf{C}}(a,b)$ can be made arbitrarily close to the ELP of the perfect cipher by taking a sufficient number of rounds. It is also possible to compute the exact value of the ELP (see Table 2), and thus determine the exact minimal number of rounds required to resist LC. See [5] for a proof of these results.

Differential Cryptanalysis (DC) [11, 12] looks for input/output difference pairs occurring with non-negligible probability for the block cipher. The efficiency of DC is inversely proportional to the *differential probability* defined by $\mathrm{DP}^{\mathsf{C}}(a,b) = \Pr[\mathsf{C}(X \oplus a) = \mathsf{C}(X) \oplus b]$ for any input/output differences $a, b \in \{0,1\}^{128}$ and uniformly distributed $X \in \{0,1\}^{128}$. Similarly to LC, the

[4] Most block cipher designers choose to compensate for possible hull effects by adding an arbitrary number of rounds. This is the case for AES [21], Camellia [2], CAST256 [1], Crypton [34], CS-Cipher [44], FOX [30] and many others.

block cipher is considered to be secure against DC if for all (a,b) pairs, the expected value $\mathrm{EDP}^{\mathsf{C}}(a,b)$ of $\mathrm{DP}^{\mathsf{C}}(a,b)$ is close to that of the perfect cipher C^*, i.e., to $1/(q^{16}-1)$ in our case.

The development we propose for LC applies similarly for DC. Indeed, in the case of Markov ciphers [32], an equation identical to (1) can be written for the EDP coefficients. The concept of linear hulls translates into the one of differentials. Again, security proofs tend to approximate the differentials using a single differential characteristic. In our case, the EDP can be made arbitrarily close to the optimal value. It is possible to compute the exact value of the EDP (see Table 2), and thus to determine the exact minimal number of rounds to resist DC. See [5] for a proof of these results.

Theorem 3. *Considering* C *on* r *rounds and any non-zero* $a,b \in \{0,1\}^{128}$ *(either considered as input/output masks or as input/output differences), we have*

$$\mathrm{ELP}^{\mathsf{C}[r]}(a,b) \xrightarrow[r\to\infty]{} \mathrm{ELP}^{\mathsf{C}^*}(a,b) \quad \textit{and} \quad \mathrm{EDP}^{\mathsf{C}[r]}(a,b) \xrightarrow[r\to\infty]{} \mathrm{EDP}^{\mathsf{C}^*}(a,b),$$

where $\mathrm{ELP}^{\mathsf{C}^*}(a,b) = \mathrm{EDP}^{\mathsf{C}^*}(a,b) = (q^{16}-1)^{-1}$. *Moreover, four rounds of* C *are enough to prove its security against linear (resp. differential) cryptanalysis as* $\max_{a\neq 0,b} \mathrm{ELP}^{\mathsf{C}[4]}(a,b) = \max_{a\neq 0,b} \mathrm{EDP}^{\mathsf{C}[4]}(a,b) = 2^{-127.91}$.

3.2 C is resistant to Impossible Differentials

Impossible Differentials [8] attacks are a variation of DC. They consist in finding pairs of input/output differences such that for any instance c of C we have $\mathrm{DP}^{\mathsf{c}}(a,b) = 0$. In other words, an input difference of a can never (i.e., for any input and any key) lead to an output difference of b. In the case of C we can prove that five rounds are enough to have no impossible differential[5], i.e., given any input/output masks a and b, there exists an instance c of $\mathsf{C}[5]$ (i.e., a key defining 80 permutations) such that $\mathrm{DP}^{\mathsf{c}}(a,b) \neq 0$.

Definition 4. *Let* $a \in \{0,1\}^{128}$ *be an arbitrary state. The* support *of* a *is a four by four binary array with 1's at the non-zero positions of* a *and 0 elsewhere. It is denoted* $\textsc{supp}(a)$. *The* weight *of the support is denoted* $w(\textsc{supp}(a))$ *or simply* $w(a)$, *and is the Hamming weight of the support. A state is said to be of* full support *when its weight is equal to 16.*

Lemma 5. *Let* $a,b \in \{0,1\}^{128}$ *be any two differences of full support. One substitution layer* S *is enough to ensure that there exists an instance* s *of* S *such that* $\mathrm{DP}^{\mathsf{s}}(a,b) \neq 0$.

Proof. Considering the two plaintexts 0 and a, we can define the 16 substitution boxes $\mathsf{s}_0,\ldots,\mathsf{s}_{15}$ of one round such that $\mathsf{s}_i(0) = 0$ and $\mathsf{s}_i(a_i) = b_i$. As both a_i and b_i are non-zero (a and b are of full support), both conditions can be verified without being inconsistent with the fact that s_i is a permutation. □

[5] There exists an impossible differential on 4 rounds of AES leading to an attack on 6 rounds [18].

Lemma 6. *Let $a \in \{0,1\}^{128}$ be a non-zero difference of arbitrary support. Considering two full rounds of C (i.e., $\mathsf{C} = \mathsf{L}^{(2)} \circ \mathsf{S}^{(2)} \circ \mathsf{L}^{(1)} \circ \mathsf{S}^{(1)}$), there exists a difference $b \in \{0,1\}^{128}$ of full support and an instance c of C such that $\mathrm{DP}^{\mathsf{c}}(a,b) \neq 0$.*

Proof (sketch). For simplicity reasons, we restrict ourselves to the case where the support of a is of weight 1. Without loss of generality, assume $a_0 \neq 0$ while $a_i = 0$ for $i = 1, \ldots, 15$. We consider the two plaintexts to be 0 and a. Letting $\mathsf{S}_i^{(1)}(0) = 0$ for all i, we have $\mathsf{L}^{(1)} \circ \mathsf{S}^{(1)}(0) = 0$. By carefully choosing $\mathsf{S}_0^{(1)}(a_0)$, we can make sure that $\mathsf{L}^{(1)} \circ \mathsf{S}^{(1)}(a)$ has a support of weight 4 (on the first columns of the four by four array). Proceeding in the same manner in the second round, we can make sure that $\mathsf{C}(0) = 0$ and $b = \mathsf{C}(a)$ is of full support. $\square$

Consider any two differences $a, b \in \{0,1\}^{128}$ and a five round version of $\mathsf{C} = \mathsf{S}^{(5)} \circ \mathsf{L}^{(4)} \circ \mathsf{S}^{(4)} \circ \mathsf{L}^{(3)} \circ \mathsf{S}^{(3)} \circ \mathsf{L}^{(2)} \circ \mathsf{S}^{(2)} \circ \mathsf{L}^{(1)} \circ \mathsf{S}^{(1)}$. From Lemma 6, there exists an instance $\mathsf{c}_{\mathrm{start}}$ of the first two rounds $\mathsf{L}^{(2)} \circ \mathsf{S}^{(2)} \circ \mathsf{L}^{(1)} \circ \mathsf{S}^{(1)}$ and a difference d of full support such that $\mathrm{DP}^{\mathsf{c}_{\mathrm{start}}}(a,d) \neq 0$. Starting from the end, there exists an instance $\mathsf{c}_{\mathrm{end}}$ of $\mathsf{S}^{(5)} \circ \mathsf{L}^{(4)} \circ \mathsf{S}^{(4)} \circ \mathsf{L}^{(3)}$ and a difference e of full support such that $\mathrm{DP}^{\mathsf{c}_{\mathrm{end}}^{-1}}(b,e) \neq 0$, so that $\mathrm{DP}^{\mathsf{c}_{\mathrm{end}}}(e,b) \neq 0$. From Lemma 5, there exists an instance $\mathsf{c}_{\mathrm{mid}}$ of $\mathsf{S}^{(3)}$ such that $\mathrm{DP}^{\mathsf{c}_{\mathrm{mid}}}(d,e) \neq 0$. Consequently, $\mathrm{DP}^{\mathsf{c}_{\mathrm{end}} \circ \mathsf{c}_{\mathrm{mid}} \circ \mathsf{c}_{\mathrm{start}}}(a,b) \neq 0$.

Property 7 (Provable security of C against Impossible Differentials). *Five rounds of C are enough to ensure that no impossible differential exists.*

3.3 C is resistant to 2-Limited Adaptive Distinguishers

In the Luby-Rackoff model [35], an adversary has an unbounded computational power and is only limited by its number of queries to an oracle $\mathcal{O}$ implementing a random permutation. Let $\mathcal{A}$ be an adversary in this model. The goal of $\mathcal{A}$ is to guess whether $\mathcal{O}$ is implementing an instance drawn uniformly among the permutations defined by the block cipher C or among all possible permutations, knowing that these two events are equiprobable and that one of them is eventually true. Denoting C^* a perfectly random permutation on $\{0,1\}^{128}$ (i.e., C^* is the perfect cipher), the ability of the adversary to succeed is measured by means of its *advantage*.

Definition 8. *The* advantage *of an adversary $\mathcal{A}$ of distinguishing two random permutations P_0 and P_1 is defined by*

$$\mathrm{Adv}_{\mathcal{A}}(P_0, P_1) = \Pr\left[\mathcal{A}(P_0) = 0\right] - \Pr\left[\mathcal{A}(P_1) = 0\right].$$

In this model, the most powerful adversary performs a d-limited adaptive attack, where d denotes the number of oracle queries. Theorem 14 in [5] gives a loose bound against 2-limited adaptive distinguishers. Using the decorrelation theory, we manage to obtain the *exact* value of the advantage of the best distinguisher.

A Dash of Decorrelation Theory. We briefly recall the results from the decorrelation theory on which our proofs are based. For the sake of simplicity, we restrict to block ciphers defined on $\{0,1\}^{128}$. Given a block cipher B, the d-wise distribution matrix $[B]^d$ is a $2^{128d} \times 2^{128d}$ matrix defined by $[B]^d_{(x_1,\ldots,x_d),(y_1,\ldots,y_d)} =$

$\Pr_B[B(x_1) = y_1, \ldots, B(x_d) = y_d]$. Theorem 10 in [47] tells us that the advantage of the best d-limited *non-adaptive* distinguisher is given by

$$\begin{aligned}\mathrm{Adv}_{\mathcal{A}_{\mathrm{na}}}(B, \mathsf{C}^*) &= \frac{1}{2}|||[B]^d - [\mathsf{C}^*]^d|||_\infty \\ &= \frac{1}{2}\max_{x_1}\cdots\max_{x_d}\sum_{y_1}\cdots\sum_{y_d}\left|[B]^d_{(x_1,\ldots,x_d),(y_1,\ldots,y_d)} - [\mathsf{C}^*]^d_{(x_1,\ldots,x_d),(y_1,\ldots,y_d)}\right|.\end{aligned}$$

Similarly, Theorem 11 in [47] gives the advantage of the best d-limited *adaptive* distinguisher

$$\begin{aligned}\mathrm{Adv}_{\mathcal{A}}(B, \mathsf{C}^*) &= \frac{1}{2}\|[B]^d - [\mathsf{C}^*]^d\|_a \\ &= \frac{1}{2}\max_{x_1}\sum_{y_1}\cdots\max_{x_d}\sum_{y_d}\left|[B]^d_{(x_1,\ldots,x_d),(y_1,\ldots,y_d)} - [\mathsf{C}^*]^d_{(x_1,\ldots,x_d),(y_1,\ldots,y_d)}\right|.\end{aligned}$$

Finally, if A and B are two independent random permutations, $[A \circ B]^d = [A]^d \times [B]^d$. For an iterated block cipher with r independent rounds, it is thus enough to compute the distribution matrix of one round and to raise it to the power r.

Computing $[\mathsf{C}]^2$. C is built as a succession of independent substitution and linear layers $\mathsf{S}^{(r)} \circ \mathsf{L} \circ \mathsf{S}^{(r-1)} \circ \cdots \circ \mathsf{L} \circ \mathsf{S}^{(1)}$. Therefore, as all the substitution layers have the same distribution matrix $[\mathsf{S}]^2$, the distribution matrix of C is given by $[\mathsf{C}]^2 = [\mathsf{S}]^2 \times [\mathsf{L}]^2 \times [\mathsf{S}]^2 \times \cdots \times [\mathsf{L}]^2 \times [\mathsf{S}]^2$.

Let $q = 2^8$ be the size of the field. For a perfectly random substitution box S we have $\Pr[S(u) = v \cap S(u') = v'] = q^{-1}$ if $u = u'$ and $v = v'$, $\Pr[S(u) = v \cap S(u') = v'] = q^{-1}(q-1)^{-1}$ if $u \neq u'$ and $v \neq v'$, and 0 otherwise. As the 16 substitution boxes of S are independent, we obtain

$$[\mathsf{S}]^2_{(x,x'),(y,y')} = \mathbf{1}_{\mathrm{SUPP}(x\oplus x')=\mathrm{SUPP}(y\oplus y')}\, q^{-16}(q-1)^{-w(x\oplus x')},$$

where we recall that $w(x \oplus x')$ denotes the Hamming weight of the support of $x \oplus x'$. We note that $[\mathsf{S}]^2$ only depends on the respective supports of the input and output differences. We will use this property to dramatically reduce the size of the matrices we have to manipulate. Denoting SP the $2^{256} \times 2^{16}$ matrix such that $SP_{(u,u'),\gamma} = \mathbf{1}_{\mathrm{SUPP}(u\oplus u')=\gamma}$ and PS the $2^{16} \times 2^{256}$ matrix such that $PS_{\gamma,(u,u')} = \mathbf{1}_{\mathrm{SUPP}(u\oplus u')=\gamma}\, q^{-16}(q-1)^{-w(\gamma)}$, we obtain $PS \times SP = Id$ and $SP \times PS = [\mathsf{S}]^2$. As the last round of C misses the linear operation, we deduce that $[\mathsf{C}]^2 = SP \times \overline{\mathsf{L}}^{r-1} \times PS$, where $\overline{\mathsf{L}} = PS \times [\mathsf{L}]^2 \times SP$ is a $2^{16} \times 2^{16}$ matrix indexed by supports. Noting that $[\mathsf{L}]^2_{(x,x'),(y,y')} = \mathbf{1}_{\mathsf{L}(x)=y}\mathbf{1}_{\mathsf{L}(x')=y'}$ and using the fact that L is linear, it is possible to expand the expression of $\overline{\mathsf{L}}$ and obtain $\overline{\mathsf{L}}_{\gamma,\gamma'} = (q-1)^{-w(\gamma)}\sum_u \mathbf{1}_{\mathrm{SUPP}(u)=\gamma}\mathbf{1}_{\mathrm{SUPP}(\mathsf{L}(u))=\gamma'}$. The matrix $\overline{\mathsf{L}}$ happens to be *precisely* the one used in the expression of the expected linear probability of C given in Theorem 6 in [5]. With our notations, the theorem states that for all support γ, γ' and any states u, u' of respective support γ and γ', we can write

$(\overline{\mathsf{L}}^{r-1})_{\gamma,\gamma'} = (q-1)^{w(\gamma')}\mathrm{ELP}^{\mathsf{C}[r]}(u,u')$, where $\mathrm{ELP}^{\mathsf{C}[r]}(u,u')$ is the expected linear probability on r rounds of C given an input (resp. output) mask u (resp. u'). Because $\mathrm{ELP}^{\mathsf{C}}$ obviously only depends on the supports γ and γ' of u and u', we will denote it from now on $\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma')$. From this, we easily obtain the following property.

Property 9. *Let $q = 2^8$ and let $\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma')$ be the expected linear probability of $r > 1$ rounds of C given an input (resp. output) mask of support γ (resp. γ'). The 2-wise distribution matrix of r rounds of C is such that $[\mathsf{C}]^2_{(x,x'),(y,y')} = q^{-16}\,\mathrm{ELP}^{\mathsf{C}}(\textsc{supp}(x\oplus x'),\textsc{supp}(y\oplus y'))$.*

Computing $\mathbf{Adv}_{\mathcal{A}}$ and $\mathbf{Adv}_{\mathcal{A}_{na}}$. The expression we just obtained for $[\mathsf{C}]^2$ leads to the following expression for $\|[\mathsf{C}]^2 - [\mathsf{C}^*]^2\|_a$:

$$\max_x \sum_y \max_{x'} \sum_{y'} \left| q^{-16}\mathrm{ELP}^{\mathsf{C}}(\textsc{supp}(x\oplus x'),\textsc{supp}(y\oplus y')) - [\mathsf{C}^*]^2_{(x,x'),(y,y')} \right|.$$

In the case where $x = x'$, the inner sum of the previous equation is 0 as $q^{-16}\mathrm{ELP}^{\mathsf{C}}(0,0) = [\mathsf{C}^*]^2_{(x,x),(y,y)} = q^{-16}$ and as $\mathrm{ELP}^{\mathsf{C}}(0,\gamma') = [\mathsf{C}^*]^2_{(x,x),(y,y')} = 0$ when $\gamma' = \textsc{supp}(y\oplus y')$ and $y' \neq y$. We obtain

$$\begin{aligned}\|[\mathsf{C}]^2 - [\mathsf{C}^*]^2\|_a &= \tfrac{1}{q^{16}} \max_x \sum_y \max_{\gamma\neq 0} \sum_{\gamma'\neq 0} \left|\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma') - \tfrac{1}{q^{16}-1}\right| \sum_{y'\neq y} \mathbf{1}_{\textsc{supp}(y\oplus y')=\gamma'} \\ &= \max_{\gamma\neq 0} \sum_{\gamma'\neq 0} \left|\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma') - \tfrac{1}{q^{16}-1}\right| (q-1)^{w(\gamma')}.\end{aligned}$$

Using similar techniques, one can derive the exact same expression for $|||[\mathsf{C}]^2 - [\mathsf{C}^*]^2|||_\infty$. This implies that, when limited to two queries, an adaptive distinguisher against C is not more powerful than a non-adaptive one. This is not surprising as the first query does not leak any information. A single substitution layer S is enough to have such a result.

Theorem 10. *The respective advantages of the best 2-limited* non-adaptive *distinguisher $\mathcal{A}_{\mathrm{na}}$ and of the best 2-limited* adaptive *distinguisher $\mathcal{A}$ against $r > 1$ rounds of C are such that $\mathrm{Adv}_{\mathcal{A}}(\mathsf{C},\mathsf{C}^*) = \mathrm{Adv}_{\mathcal{A}_{\mathrm{na}}}(\mathsf{C},\mathsf{C}^*)$ and (taking the sum over all non-zero supports)*

$$\mathrm{Adv}_{\mathcal{A}}(\mathsf{C},\mathsf{C}^*) = \tfrac{1}{2}\max_{\gamma} \sum_{\gamma'\neq 0} \left|\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma') - \mathrm{ELP}^{\mathsf{C}^*}(\gamma,\gamma')\right|(q-1)^{w(\gamma')}.$$

Practical computations can take into account the fact that $\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma')$ actually only depends on the 4 weights of the diagonals of γ and on those of the columns of γ' (from Theorem 12 in [5], see Appendix A). Results of our practical computations are reported in Table 3 (together with the corresponding upper bounds obtained in [5]). Finally, we can obtain the following corollary from Theorem 3 and Theorem 10.

Corollary 11. *The advantage of the best 2-limited adaptative distinguisher $\mathcal{A}$ against $\mathsf{C}[r]$ tends towards 0 as r increases, i.e., $\mathrm{Adv}_{\mathcal{A}}(\mathsf{C}[r],\mathsf{C}^*) \xrightarrow[r\to\infty]{} 0$.*

Table 3. Exact values of the advantage of the best 2-limited adaptive distinguisher for several number of rounds r compared to the bounds given in [5].

r	2	3	4	5	6	7	8	9	10	11	12
Bound	$2^{94.0}$	$2^{72.0}$	$2^{-4.0}$	$2^{-4.0}$	$2^{-24.2}$	$2^{-46.7}$	$2^{-72.4}$	$2^{-95.9}$	$2^{-142.8}$	-	-
Exact	1	$2^{-4.0}$	$2^{-23.4}$	$2^{-45.8}$	$2^{-71.0}$	$2^{-126.3}$	$2^{-141.3}$	$2^{-163.1}$	$2^{-185.5}$	$2^{-210.8}$	$2^{-238.9}$

3.4 C is resistant to Iterated Attacks of Order 1

Iterated attacks of order 1 [46, 47] are very similar to LC except that the bit of information retrieved from each plaintext/cipher pair does not necessarily have to be derived in a linear way. Such attacks have proven to be sometimes much more powerful than linear cryptanalysis[6]. According to Theorem 18 in [47], proving resistance against $2d$-limited adaptive distinguishers is enough to prove resistance to iterated attacks of order d. We can deduce that C is immune to any iterated attack of order 1.

Property 12 (Provable Security of C against iterated attacks of order 1). *Seven rounds of* C *are sufficient to obtain provable security against iterated attacks of order 1.*

3.5 All Substitution Boxes of C are Indistinguishable from Independent Perfectly Random Permutations

A pseudo-random bit generator is said to be cryptographically secure if no polynomial-time statistical test can distinguish an output sequence of this generator from a perfectly random bit string with a significant advantage [51]. Such a generator can always be distinguished if the length of the bit string is longer than the generator's period. We need to prove that the Blum-Blum-Shub generator we use has a period long enough to generate a complete extended key.

We know from the original paper [15] that the period of the x_i's sequence of the BBS generator divides $\lambda(\lambda(n))$ (where λ denotes the Carmichael function) if both p and q are strong-primes and both p and q are Blum integers. Obviously, the period of the bit string output by BBS divides the period of the x_i's. By making sure that $\lambda(\lambda(n))$ does not contain small factors, we can prove that this length will be large enough. This can be done by choosing strong-strong-primes p and q. In such a case we can write $p = 2p_1 + 1 = 4p_2 + 3$ and $q = 2q_1 + 1 = 4q_2 + 3$, and obtain $\lambda(\lambda(n)) = \lambda(\text{lcm}(2\,p_1, 2\,q_1)) = \lambda(2\,p_1\,q_1) = \text{lcm}(2\,p_2, 2\,q_2) = 2\,p_2\,q_2$. Therefore, if the period of the bit string is not 2, it is necessarily long enough to generate a complete extended key as $\min(p_2, q_2) \gg 300\,000$.

It is known that the original Blum-Blum-Shub pseudo-random bit generator is cryptographically secure [15,16]. Vazirani and Vazirani showed that outputting both the least and most significant bits of the quadratic residues produced by the generator is also cryptographically secure [48, 49].

[6] see for example [4, pg. 9], where an example of a biased source is given. Although impossible to distinguish from a true random source with a linear distinguisher, this source is easily broken by a non-linear distinguisher

Definition 13. *Let s_0 and s_1 be two bit strings, such that s_0 is obtained using the BBS pseudo-random generator and s_1 is perfectly random. The advantage of an adversary $\mathcal{A}$ trying to distinguish s_0 from s_1 is given by*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{BBS}} = \Pr\left[\mathcal{A}(s_0) = 0\right] - \Pr\left[\mathcal{A}(s_1) = 0\right].$$

Assuming that the problem of deciding the quadratic residuosity modulo n is hard (an assumption we will refer to as the *quadratic residuosity assumption* [27]), we know that $\mathrm{Adv}_{\mathcal{A}}^{\mathsf{BBS}}$ can be made arbitrarily small by increasing the value of n. The key schedule of C relies on the BBS generator and makes sure that the mapping from the set of 2^{128} keys to the set of possible seeds of the pseudo-random generator is injective. Therefore, the pseudo-random sequence produced by the key schedule of C is indistinguishable from a perfectly random binary sequence of the same length. The method we use to convert this binary sequence into substitution boxes makes sure that for an unbiased sequence one obtains an unbiased set of substitution boxes. By choosing a suitable n, the substitution boxes of C can thus be made indistinguishable from independent perfectly random permutations.

3.6 The Keyed C is Not Less Secure than C

Definition 14. *Let k_0 and k_1 be two extended keys of C, such that k_0 is obtained through the key schedule seeded by a perfectly random 128 bit key and k_1 is perfectly random. The advantage of an adversary $\mathcal{A}$ trying to distinguish k_0 from k_1 is given by*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{key}} = \Pr\left[\mathcal{A}(k_0) = 0\right] - \Pr\left[\mathcal{A}(k_1) = 0\right].$$

Property 15. *Let k_0 and k_1 be two extended keys as in Definition 14 and s_0 and s_1 be two bit strings as in Definition 13. An adversary $\mathcal{A}$ able to distinguish k_0 from k_1 with probability p can distinguish s_0 from s_1 with probability $p' \geq p$, i.e.,* $\mathrm{Adv}_{\mathcal{A}}^{\mathsf{key}} \leq \mathrm{Adv}_{\mathcal{A}}^{\mathsf{BBS}}$.

Proof. Given s_b ($b \in \{0,1\}$), the adversary can derive an acceptable extended key k_b. From this, the adversary has an advantage $\mathrm{Adv}_{\mathcal{A}}^{\mathsf{key}}$ of guessing the correct value of b and thus obtains a distinguisher on BBS with advantage $\mathrm{Adv}_{\mathcal{A}}^{\mathsf{key}}$. □

The strongest notion of security for a block cipher is its indistinguishability from a perfectly random permutation C^*. Proving the security of C against a distinguishing attack performed by $\mathcal{A}$ consists in upper bounding $\mathrm{Adv}_{\mathcal{A}}(\mathsf{C}, \mathsf{C}^*)$.

Let k_0 and k_1 be two random extended keys of C picked as in Definition 14, defining two random instances of C denoted $\mathsf{C}_{\mathsf{key}}$ and $\mathsf{C}_{\mathsf{rand}}$ respectively. Obviously, distinguishing $\mathsf{C}_{\mathsf{key}}$ from $\mathsf{C}_{\mathsf{rand}}$ is harder than distinguishing k_0 from k_1, so that $\mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{key}}, \mathsf{C}_{\mathsf{rand}}) \leq \mathrm{Adv}_{\mathcal{A}}^{\mathsf{key}}$.

Assume there exists a distinguishing attack on $\mathsf{C}_{\mathsf{key}}$ that does not work on $\mathsf{C}_{\mathsf{rand}}$ such that, for an adversary $\mathcal{A}$ using it, $\mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{key}}, \mathsf{C}^*) \geq 2 \cdot \mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{rand}}, \mathsf{C}^*)$. From the triangular inequality we have $\mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{key}}, \mathsf{C}^*) - \mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{rand}}, \mathsf{C}^*) \leq \mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{key}}, \mathsf{C}_{\mathsf{rand}})$ so that $\mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{key}}, \mathsf{C}^*) \leq 2 \cdot \mathrm{Adv}_{\mathcal{A}}(\mathsf{C}_{\mathsf{key}}, \mathsf{C}_{\mathsf{rand}}) \leq 2 \cdot \mathrm{Adv}_{\mathcal{A}}^{\mathsf{key}}$.

In conclusion, using Property 15, any distinguishing attack twice as efficient on $\mathsf{C}_{\mathsf{key}}$ than on $\mathsf{C}_{\mathsf{rand}}$ gives an advantage which is bounded by $2 \cdot \mathrm{Adv}_{\mathcal{A}}^{\mathsf{BBS}}$. Under the quadratic residuosity assumption, such an attack cannot be efficient.

Although the quadratic residuosity problem is not equivalent to the problem of factoring $p \cdot q$, the best known attacks require it. The exact cost of this factorization is not obvious. For a given symmetric key size, there are several estimates for an equivalent asymmetric key size [31]. According to the NIST recommendations, a 2048 bit modulus is equivalent to a 112 bit symmetric key [24].

Property 16 (Provable security of $\mathsf{C}_{\mathsf{key}}$). *Under the quadratic residuosity assumption,* C *used with the key schedule described in Section 2.3 is as secure as* C *used with independent perfectly random substitution boxes.*

3.7 The Keyed C has no Equivalent Keys

Two block cipher keys are said to be *equivalent* when they define the same permutation. It is easy to build equivalent *extended* keys for C (when *not* using the key schedule). Consider an extended key k_1 defining a set of 160 substitution boxes such that the first 32 are the identity. We consider a second extended key k_2 defining another set of substitution boxes such that the last 128 are identical to that defined by k_1 and such that the first 16 boxes simply xor a constant $a \in \{0,1\}^{128}$ to the plaintext, the remaining boxes (in the second layer) correcting the influence of a by xoring $\mathsf{L}(a)$ to its input. Although they are different, k_1 and k_2 define the same permutation. Such a property could be a threat to the security of C. If too many such extended keys were equivalent, it could be possible to find equivalent 128 bit keys for $\mathsf{C}_{\mathsf{key}}$. We can prove that the probability that two 128 bit equivalent keys exist is negligible.

The probability that two equivalent 128 bit keys exist depends on the number of equivalence classes among the extended keys. Considering a one round version of C, it can be seen that no equivalent extended keys exist. Consequently, there are at least $(2^8!)^{16} \approx 2^{26944}$ equivalence classes. Adding rounds (thus increasing the extended key size) cannot decrease this number of classes. Assuming that the key schedule based on BBS uniformly distributes the extended keys obtained from the 128 bit keys among these classes, the probability that two keys fall into the same class can be upper bounded by

$$1 - e^{-(2^{128})^2/(2*2^{26944})} \approx 2^{-26689}.$$

Property 17 ($\mathsf{C}_{\mathsf{key}}$ has no Equivalent Keys). *The probability that two 128 bit keys lead to the same instance of* C *is upper bounded by* 2^{-26689}.

4 Security Results: What we Believe to be True

4.1 C is (not that) Resistant to Saturation Attacks

Saturation attacks [20] are chosen-plaintext attacks on byte-oriented ciphers. An attack on four rounds of AES can be performed [22] by choosing a set of 2^8

plaintexts equal on all but one byte. After 3 rounds of AES, the xor of all the corresponding ciphertexts is 0. This makes it easy to guess the key of the fourth round, as all round key bytes can be guessed independently.

In our case, the property on the third round output still holds. Nevertheless, it only allows to exclude 255 out of 256 keys for each substitution box. This was enough for AES, but in our case an adversary would still be left with 255! valid substitution boxes, so that a more subtle approach is needed.

In [13], Biryukov and Shamir present an attack on SASAS, a generic construction with three rounds of random key-dependent substitution boxes linked by random key-dependent affine layers. Following their approach, the saturation attacks on the AES can be adapted to C but with a non-negligible cost. In this approach, an exhaustive search on 8 bits (as necessary with the AES) is replaced by a linear algebra step which requires 2^{24} operations. The additional workload is thus of the order of 2^{16}. This overhead implies that any attack with a complexity higher than 2^{112} becomes infeasible. In particular the saturation attacks on 7 rounds of the AES [23] should not apply to C.

We believe that saturation-like attacks are the biggest threat for reduced rounds versions of C. Chances that such attacks apply to 10 rounds are however very low.

4.2 C is Resistant to a Wide Variety of Attacks

Algebraic attacks consist in rewriting the whole block cipher as a system of algebraic equations. The solutions of this system correspond to valid plaintext, ciphertext, and key triples. Algebraic attack attempts on AES take advantage of the simple algebraic structure of the substitution box [19]. In our case, substitution boxes can by no means be described by simple algebraic forms, and thus, algebraic attacks will necessarily be much more complex against C than against AES. We do believe that they will be more expensive than exhaustive key search.

Slide attacks [14] exploit a correlation between the different round keys of a cipher. These attacks apply for example against ciphers with weak key schedules or against block ciphers with key-dependent substitution boxes and periodic key schedules. C uses independent perfectly random substitution boxes, so that all rounds are independent from each other. Slide attacks cannot apply here.

The boomerang attack [50] is a special type of differential cryptanalysis. It needs to find a differential characteristic on half the rounds of the cipher. Four rounds of C being sufficient to be provably secure against DC, 10 rounds are necessarily sufficient to resist the boomerang attack. Similarly, neither differential-linear cryptanalysis [10, 33] nor the rectangle attack [9] apply to C.

5 Reducing the Extended Key Size

The main drawback in the design of C is the huge amount of pseudo-random bits required for the key schedule. Having to generate hundreds of thousands of bits with the Blum-Blum-Shub generator is unacceptable for many applications. We propose here an adaptation of C, enjoying the same security proofs, but requiring much less pseudo-random bits.

Using Order 2 Decorrelated Substitutions Boxes. As stated in [5], the bounds on the LP and DP obtained when replacing the substitution boxes of the AES by independent perfectly random permutations remain exactly the same if one uses independent order 2 decorrelated substitution boxes instead. This is also the case concerning resistance against 2-limited adaptive distinguishers and, as a consequence, resistance against iterated attacks of order 1.

Suppose we have a family $\mathcal{D}_2$ of order 2 decorrelated substitution boxes. Using the Blum-Blum-Shub generator and the same method as for the standard C key schedule, we can generate a set of 160 substitution boxes from $\mathcal{D}_2$ indistinguishable from 160 randomly chosen $\mathcal{D}_2$ boxes. Again, it is possible to prove that any attack on a keyed C using substitution boxes in $\mathcal{D}_2$ requires to be able to distinguish the output of the Blum-Blum-Shub generator from a perfectly random binary stream.

Hence, apart from the resistance to impossible differentials, all proven security arguments of C remain untouched when using boxes of $\mathcal{D}_2$. However, each time the key schedule required $\log_2 256!$ bits from the Blum-Blum-Shub generator, it only requires $\log_2 |\mathcal{D}_2|$ now.

$A \oplus \frac{B}{X}$: a Good Family of Order 2 Decorrelated Substitution Boxes. From what we have just seen, whatever the family $\mathcal{D}_2$ we use, security results will still hold. For optimal efficiency, we need to select the smallest possible such family. It was shown in [3] that any family of the form $\mathcal{D}_2 = \{X \mapsto A \oplus B \cdot S(X); A, B \in \{0,1\}^8, B \neq 0\}$ where S is any *fixed* permutation of $\mathrm{GF}(2^8)$ (and where $\cdot$ represents a product in $\mathrm{GF}(2^8)$) is decorrelated at order 2.

We propose to use the family $\mathcal{D}_2 = \{X \mapsto A \oplus \frac{B}{X}; A, B \in \{0,1\}^8, B \neq 0\}$. This family contains 2^{16} elements and the substitution boxes can be chosen uniformly in $\mathcal{D}_2$ from 16 bits of the Blum-Blum-Shub generator. The first 8 bits define A, the last 8 define B. So, the whole key schedule for ten rounds of C only requires 2 560 pseudo-random bits and should be about 100 times faster than an unmodified C with perfectly random permutations. One may believe that this construction is very similar to that of the AES (assuming that the round keys are independent and perfectly random). Nevertheless, deriving the AES construction from ours requires to set $B = 1$. The family obtained in this case is no longer decorrelated at order 2, so that, unfortunately, none of the security results we obtained for C directly applies to the AES.

Security Considerations. Even if this might not be the case for any order 2 decorrelated family of substitution boxes, it is interesting to note that C built on the family $\mathcal{D}_2$ we chose is also resistant to impossible differentials. As for perfectly random permutations, lemmas 5 and 6 can both be proven for boxes of the form $A \oplus \frac{B}{X}$.

None of the security results we obtained requires using perfectly random permutations and substitution boxes of the form $A \oplus \frac{B}{X}$ are enough. We believe that achieving the same security level with perfectly random permutations is possible with fewer rounds. More precisely, it may be possible to obtain a trade-off between the number of rounds and the level of decorrelation of the random substitution boxes. Fewer rounds lead to fast encryption/decryption procedures but require a higher level of decorrelation. In this case, more pseudo-random bits are necessary to generate each substitution box, and this may lead to a (very) slow key schedule. The best choice depends on the application.

6 Implementation and Performances

Implementation. As seen in Section 2.3, before being able to use the Blum-Blum-Shub generator, one needs to generate two strong-strong-primes p and q, which is not an easy operation: it has a complexity of $O((\log p)^6)$. For primes of length 1024, this takes one million times more operations than generating a prime of the same size. Some optimizations exist to improve the constant factor in the prime number generation [29] and can become very useful for strong-strong-prime numbers.

When implementing C, the same optimizations as for AES are possible. In particular, one round of C can be turned into 16 table look-ups and 12 xors. Basically, the output can be split in four 32 bits blocks, each of which only depends on four bytes of the input. However, all the tables of C are different from each other. This is the only reason why encrypting/decrypting with C could be slower than with AES. Considering standard 32-bits computers, this has little influence in practice as the 160 tables still fit in the cache of the CPU. The required memory is $160 \cdot 256 \cdot 4 = 160$kBytes. This however becomes an issue when implementing C on a smartcard (but who wants to implement Blum-Blum-Shub on a smartcard anyway?) or on a CPU with 128 kBytes of cache.

We programmed C in C using GMP [25] for the key schedule operations. On a 3.0 GHz Pentium D, we obtain encryption/decryption speeds of 500 Mbits/s. Generating the 160 substitution boxes from the 128 bit secret key takes 2.5s when using perfectly random permutations and 25ms when using the $A \oplus \frac{B}{X}$ construction. Note that to decrypt, it is also necessary to invert the substitution boxes. This takes a negligible time compared to the generation of the extended key, which is the most expensive step of the key schedule.

Applications. Given the timings we obtained, it appears that using C for encryption purpose is practical, in particular with the shortened key schedule. Of course, a key schedule of 25ms is much slower than most existing key schedules but is still acceptable in a large majority of applications. This can become negligible when the amount of data to encrypt becomes large.

The 2.5s obtained for the “most secure” version using perfectly random substitution boxes is suitable for only a few very specific applications. However, we believe that in the case where a very high security level is required, this price is not that high. This might not be an issue in certain cases when the key schedule is run in parallel with some other slow operation, like for hard disk drive encryption (for which the key schedule is performed only once during a boot sequence which already takes several seconds).

In some other circumstances however, C is not usable at all. For example, when using it as a compression function in a Merkle-Damgård construction, as one key schedule has to be performed for each block (hashing a 1 MByte message would take more than one day).

Further Improvements. It is known that outputting $\alpha(n) = O(\log \log n)$ bits at each iteration of the Blum-Blum-Shub generator is cryptographically secure [49]. However, for a modulus n of given bit length, no explicit range for $\alpha(n)$ was ever given in the literature [40]. Finding such a constant could considerably improve the speed of the key schedule of C.

Another possible improvement to the key schedule would be to rely on some other cryptographically secure pseudo-random generator. The pseudo-random generator on which the stream cipher QUAD [6, 7] is based may be a good candidate: it offers provable security results and achieves speeds up to 5.7Mbits/s. Using such a construction would certainly improve the key schedule time by an important factor, so that the "most secure" version of C might compare to the current version using derandomized substitution boxes.

7 Conclusion

We have introduced C, a block cipher provably secure against a wide range of attacks. It is as fast as AES for encryption on a standard workstation. Provable security requires a cryptographically secure key schedule. Consequently, the key schedule of C is too slow for some applications.

As far as we know, C is the first practical block cipher to provide tight security proofs that do take into account the key schedule. It is proven that C resists: linear cryptanalysis (taking into account the possible cumulative effects of a linear hull), differential cryptanalysis (similarly considering cumulative effects of differentials), 2-limited adaptive distinguishers, iterated attacks of order 1, and impossible differentials. We also give strong evidence that it also resists: algebraic attacks, slide attacks, the boomerang attack, the rectangle attack, differential-linear cryptanalysis, and, to some extent, saturation attacks. From our point of view, the most significant improvement that could be made on C would be to give a bound on the advantage of the best d-limited adversary for $d > 2$.

"Mind you, even I didn't think of that one... extraordinary."
Chief Insp. Hubbard

References

1. C. Adams, H.M. Heys, S.E. Tavares, and M. Wiener. CAST256: a submission for the advanced encryption standard, 1998. First AES Candidate Conference (AES1).

2. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita. Camellia: a 128-bit block cipher suitable for multiple platforms - design and analysis. In D.R. Stinson and S.E. Tavares, editors, *Selected Areas in Cryptography,* SAC *'00*, volume 2012 of *LNCS*, pages 39–56. Springer-Verlag, 2001.
3. K. Aoki and S. Vaudenay. On the use of GF-inversion as a cryptographic primitive. In M. Matsui and R. Zuccherato, editors, *Selected Areas in Cryptography,* SAC *'03*, volume 3006 of *LNCS*, pages 234–247. Springer-Verlag, 2004.
4. T. Baignères, P. Junod, and S. Vaudenay. How far can we go beyond linear cryptanalysis? In P.J. Lee, editor, *Advances in Cryptology* - Asiacrypt *'04*, volume 3329 of *LNCS*, pages 432–450. Springer-Verlag, 2004.
5. T. Baignères and S. Vaudenay. Proving the security of AES substitution-permutation network. In B. Preneel and S.E. Tavares, editors, *Selected Areas in Cryptography,* SAC *'05*, volume 3897 of *LNCS*, pages 65–81. Springer-Verlag, 2006.
6. C. Berbain, O. Billet, and H. Gilbert. Efficient implementations of multivariate quadratic systems. In E. Biham and A.M. Youssef, editors, *Selected Areas in Cryptography,* SAC *'06*, LNCS. Springer-Verlag, To appear.
7. C. Berbain, H. Gilbert, and J. Patarin. QUAD: a practical stream cipher with provable security. In S. Vaudenay, editor, *Advances in Cryptology* - Eurocrypt *'06*, volume 4004 of *LNCS*, pages 109–128. Springer-Verlag, 2006.
8. E. Biham, A. Biryukov, and A. Shamir. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In J. Stern, editor, *Advances in Cryptology* - Eurocrypt *'99*, volume 1592 of *LNCS*, pages 12–23. Springer-Verlag, 1999.
9. E. Biham, O. Dunkelman, and N. Keller. The rectangle attack - rectangling the Serpent. In B. Pfitzmann, editor, *Advances in Cryptology* - Eurocrypt *'01*, volume 2045 of *LNCS*, pages 340–357. Springer-Verlag, 2001.
10. E. Biham, O. Dunkelman, and N. Keller. Enhancing differential-linear cryptanalysis. In Y. Zheng, editor, *Advances in Cryptology* - Asiacrypt *'02*, volume 2501 of *LNCS*, pages 254–266. Springer-Verlag, 2002.
11. E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4:3–72, 1991.
12. E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems (extended abstract). In A. Menezes and S. Vanstone, editors, *Advances in Cryptology* - Crypto *'90*, volume 537 of *LNCS*, pages 2–21. Springer-Verlag, 1991.
13. A. Biryukov and A. Shamir. Structural cryptanalysis of SASAS. In B. Pfitzmann, editor, *Advances in Cryptology* - Eurocrypt *'01*, volume 2045 of *LNCS*, pages 394–405. Springer-Verlag, 2001.
14. A. Biryukov and D. Wagner. Slide attacks. In L. Knudsen, editor, *Fast Software Encryption* - FSE *'99*, volume 1636 of *LNCS*, pages 245–259. Springer-Verlag, 1999.
15. L. Blum, M. Blum, and M. Shub. Comparison of two pseudo-random number generators. In D. Chaum, R.L. Rivest, and A. Sherman, editors, *Advances in Cryptology* - Crypto *'82*, pages 61–78. Plemum, 1983.
16. L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15(2):364–383, May 1986.
17. F. Chabaud and S. Vaudenay. Links between differential and linear cryptanalysis. In A. De Santis, editor, *Advances in Cryptology* - Eurocrypt *'94*, volume 950 of *LNCS*, pages 356–365. Springer-Verlag, 1995.
18. J.H. Cheon, M.J. Kim, K. Kim, J.-Y. Lee, and S.W. Kang. Improved impossible differential cryptanalysis of Rijndael and Crypton. In *Information Security and Cryptology ICISC'01*, volume 2288 of *LNCS*, pages 39–49. Springer, 2002.

19. N. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In Y. Zheng, editor, *Advances in Cryptology* - Asiacrypt *'02*, volume 2501 of *LNCS*, pages 267–287. Springer-Verlag, 2002.
20. J. Daemen, L. Knudsen, and V. Rijmen. The block cipher SQUARE. In E. Biham, editor, *Fast Software Encryption* - FSE *'97*, volume 1267 of *LNCS*, pages 149–165. Springer-Verlag, 1997.
21. J. Daemen and V. Rijmen. AES proposal: Rijndael. NIST AES Proposal, 1998.
22. J. Daemen and V. Rijmen. *The Design of Rijndael*. Information Security and Cryptography. Springer-Verlag, 2002.
23. N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting. Improved cryptanalysis of Rijndael. In B. Schneier, editor, *Fast Software Encryption* - FSE *'00*, volume 1978 of *LNCS*, pages 213–230. Springer-Verlag, 2001.
24. C. Gehrmann and M. Näslund. Ecrypt yearly report on algorithms and keysizes (2005). Technical report, Ecrypt, 2006.
25. GMP. GNU Multiple Precision arithmetic library. http://www.swox.com/gmp.
26. O. Goldreich and L.A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing - STOC'89*, pages 25–32. ACM Press, 1989.
27. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
28. A. Hitchcock. Dial M for Murder, 1954. http://www.imdb.com/title/tt0046912.
29. M. Joye, P. Paillier, and S. Vaudenay. Efficient generation of prime numbers. In C.K. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems* - Ches *'00*, volume 1965 of *LNCS*, pages 340–354. Springer-Verlag, 2000.
30. P. Junod and S. Vaudenay. FOX: a new family of block ciphers. In H. Handschuh and A. Hasan, editors, *Selected Areas in Cryptography,* SAC *'04*, volume 3357 of *LNCS*, pages 114–129. Springer-Verlag, 2005.
31. Keylength.com. http://www.keylength.com.
32. X. Lai, J. Massey, and S. Murphy. Markov ciphers and differential cryptanalysis. In D.W. Davies, editor, *Advances in Cryptology* - Eurocrypt *'91*, volume 547 of *LNCS*, pages 17–38. Springer-Verlag, 1991.
33. S.K. Langford and M.E. Hellman. Differential-linear cryptanalysis. In Y.G. Desmedt, editor, *Advances in Cryptology* - Crypto *'94*, volume 839 of *LNCS*, pages 17–25. Springer-Verlag, 1994.
34. C.H. Lim. A revised version of CRYPTON: CRYPTON V1.0. In L. Knudsen, editor, *Fast Software Encryption* - FSE *'99*, volume 1636 of *LNCS*, pages 31–45. Springer-Verlag, 1999.
35. M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.
36. J. Massey. SAFER-K64: a byte-oriented block-ciphering algorithm. In R.J. Anderson, editor, *Fast Software Encryption* - FSE *'93*, volume 809 of *LNCS*, pages 1–17. Springer-Verlag, 1994.
37. M. Matsui. The first experimental cryptanalysis of the Data Encryption Standard. In Y.G. Desmedt, editor, *Advances in Cryptology* - Crypto *'94*, volume 839 of *LNCS*, pages 1–11. Springer-Verlag, 1994.
38. M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology* - Eurocrypt *'93*, volume 765 of *LNCS*, pages 386–397. Springer-Verlag, 1994.
39. M. Matsui. New structure of block ciphers with provable security against differential and linear cryptanalysis. In D. Gollmann, editor, *Fast Software Encryption* - FSE *'96*, volume 1039 of *LNCS*, pages 205–218. Springer-Verlag, 1996.

40. A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of applied cryptography.* The CRC Press series on discrete mathematics and its applications. CRC-Press, 1997.
41. K. Nyberg. Linear approximation of block ciphers. In A. De Santis, editor, *Advances in Cryptology* - EUROCRYPT *'94*, volume 950 of *LNCS*, pages 439–444. Springer-Verlag, 1995.
42. L. O'Connor. Properties of linear approximation tables. In B. Preneel, editor, *Fast Software Encryption* - FSE *'94*, volume 1008 of *LNCS*, pages 131–136. Springer-Verlag, 1995.
43. C.E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, October 1949. Re-edited in *Claude Elwood Shannon - Collected Papers*. IEEE Press, New York, 1993.
44. J. Stern and S. Vaudenay. CS-Cipher. In S. Vaudenay, editor, *Fast Software Encryption* - FSE *'98*, volume 1372 of *LNCS*, pages 189–204. Springer-Verlag, 1998.
45. A. Tardy-Corfdir and H. Gilbert. A known plaintext attack of FEAL-4 and FEAL-6. In J. Feigenbaum, editor, *Advances in Cryptology* - CRYPTO *'91*, volume 576 of *LNCS*, pages 172–182. Springer-Verlag, 1992.
46. S. Vaudenay. Resistance against general iterated attacks. In J. Stern, editor, *Advances in Cryptology* - EUROCRYPT *'99*, volume 1592 of *LNCS*, pages 255–271. Springer-Verlag, 1999.
47. S. Vaudenay. Decorrelation: a theory for block cipher security. *Journal of Cryptology*, 16(4):249–286, 2003.
48. U. Vazirani and V. Vazirani. Efficient and secure pseudo-random number generation. In G. Blakley and D. Chaum, editors, *Advances in Cryptology* - CRYPTO *'84*, volume 196 of *LNCS*, pages 193–202. Springer-Verlag, 1985.
49. U. Vazirani and V. Vazirani. Efficient and secure pseudo-random number generation (extended abstract). In *Proceedings of FOCS'84*, pages 458–463. IEEE, 1985.
50. D. Wagner. The boomerang attack. In L. Knudsen, editor, *Fast Software Encryption* - FSE *'99*, volume 1636 of *LNCS*, pages 156–170. Springer-Verlag, 1999.
51. A.C. Yao. Theory and applications of trapdoor functions (extended abstract). In *Proceedings of FOCS'82*, pages 80–91, 1982.
52. A.M. Youssef and S.E. Tavares. Resistance of balanced S-boxes to linear and differential cryptanalysis. *Information Processing Letters*, 56:249–252, 1995.

A Further Reducing the Matrix Size

From Theorem 12 in [5], we know that $\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma')$ actually only depends on the weights of the diagonals of γ and of the columns of γ'. Respectively denoting $\nu=(\nu_0,\nu_1,\nu_2,\nu_3)$ and $\mu=(\mu_0,\mu_1,\mu_2,\mu_3)$ those two sets of 4 weights, we obtain from Theorem 10 that

$$\begin{aligned}2\,\mathrm{Adv}_{\mathcal{A}} &= \max_{\gamma}\sum_{\gamma'\neq 0}\left|\mathrm{ELP}^{\mathsf{C}}(\gamma,\gamma')-\mathrm{ELP}^{\mathsf{C}^*}(\gamma,\gamma')\right|(q-1)^{w(\gamma')}\\ &= \max_{\nu}\sum_{\mu\neq 0}\left|\mathrm{ELP}^{\mathsf{C}}(\nu,\mu)-\mathrm{ELP}^{\mathsf{C}^*}(\nu,\mu)\right|(q-1)^{w(\mu)}B[\mu],\end{aligned}$$

where $B[\mu]=\binom{4}{\mu_0}\binom{4}{\mu_1}\binom{4}{\mu_2}\binom{4}{\mu_3}$ denotes the number of distinct supports having a column weight pattern equal to μ. Consequently, the final computation can be reduced to computations on 625×625 matrices.